

IBM Rational Service Tester for SOA Quality
10.5.2 Documentation
March 2023

Special notice

Before using this information and the product it supports, read the information in [Notices on page cmvii](#).

Contents

Special notice.....	ii	Installation of the product by using IBM® Installation Manager.....	51
Chapter 1. Release Notes.....	12	Uninstalling the product by using IBM® Installation Manager.....	62
Product description.....	12	License management.....	63
What's new.....	12	License descriptions.....	63
Installing the product.....	14	Runtime license examples.....	65
Known issues.....	14	Purchasing licenses.....	65
Contacting IBM Rational Software Support.....	14	License enablement.....	66
Chapter 2. System Requirements.....	16	Viewing license information for installed packages.....	67
Hardware.....	17	Product upgrade and migration.....	67
Operating systems.....	20	Migrating test assets to new version of the product.....	67
Host prerequisites.....	22	Integration with other products.....	68
Recording support.....	24	Integration plugin compatibility matrix.....	68
Supported software.....	25	Running tests by using Ant.....	69
Chapter 3. Getting Started.....	29	Integration with Azure DevOps.....	82
Rational® Service Tester for SOA Quality overview.....	29	Integration with Apache JMeter.....	97
Service testing overview.....	29	EGit integration.....	104
Generic service client overview.....	31	Integration with Engineering Test Management.....	113
Socket API performance testing overview.....	33	Integration with IBM® Engineering Workflow Management.....	124
TN3270 performance testing overview.....	33	Integration with IBM® Rational® Integration Tester.....	127
IBM® Engineering Test Management overview.....	34	Integration with IBM® UrbanCode™ Deploy.....	132
Streamlined Eclipse and full Eclipse overview.....	36	Integration of Jaeger with the product.....	155
Starting the product in full Eclipse mode.....	36	Integration with Jenkins.....	157
Starting the product in streamlined Eclipse mode.....	37	Testing with Maven.....	185
Chapter 4. Tutorials.....	38	Integration with Micro Focus ALM.....	188
Test an SOA application.....	38	Chapter 6. Test Author Guide.....	204
Introduction: Testing services.....	38	Creating tests.....	204
Module 1: Recording service calls and creating tests.....	39	Creating a project.....	204
Module 2: Editing service tests.....	40	Recording service tests.....	204
Module 3: Running service tests.....	42	Digital certificates overview.....	229
Module 4: Simulating services.....	43	Annotating a test during recording.....	238
Summary.....	44	Recording sensitive session data.....	239
Chapter 5. Administrator Guide.....	46	Splitting a test during recording.....	240
Installation of Rational® Performance Tester.....	46	Generating a new test from a recorded session.....	240
Installation requirements.....	46	Organizing test assets by type.....	241
Installation conventions and terminology.....	47	Editing tests.....	242
Capacity Planning.....	47	Editing service tests.....	242
Installation Manager overview.....	48	Searching within tests.....	276
Installation locations.....	48	Exporting a test.....	279
Offering coexistence considerations.....	49	Copying test assets with dependencies.....	279
Eclipse instance overview.....	49		
Increasing the number of file handles on Linux™ workstations.....	50		
Starting the launchpad.....	51		

Disabling portions of a test.....	281	Reducing the performance impact of custom code.....	407
Running test elements in random order.....	283	Custom code examples.....	408
Renaming test assets.....	284	Migrating custom code from previous versions.....	436
Deleting test assets.....	286	Chapter 8. Test Manager Guide.....	437
Debugging custom code for tests and compound tests.....	287	Evaluating results in web analytic reports.....	437
Providing tests with variable data (datasets).....	288	Comparing results among runs.....	437
Test variables.....	308	Comparing schedule stages.....	437
Correlating response and request data.....	315	Comparing results from various regions or agent locations.....	438
Simulating services with stubs.....	333	Generating functional reports.....	439
Service stub overview.....	333	Publishing functional reports to the server.....	441
Creating a service stub.....	334	Publishing test results to the server.....	442
Editing a service stub.....	335	Publishing specific results to the server.....	445
Deploying service stubs.....	336	Customizing reports.....	446
Recording service stub activity in a log file.....	336	Export test results.....	455
Setting log level for service stubs.....	337	Viewing response time breakdown.....	463
Sending service requests with the generic service client.....	338	Comparing service test response contents.....	465
Creating transport protocol configurations.....	338	Logs overview.....	465
Sending service requests with WSDL files.....	350	Viewing test logs.....	467
Sending HTTP endpoint requests.....	352	Viewing reports after a run.....	467
Sending a JMS endpoint request.....	354	Accessing reports remotely.....	468
Sending a WebSphere® MQ endpoint request.....	355	Exporting test logs.....	469
Sending OData endpoint batch requests.....	356	Exporting event log.....	469
Sending WebSphere Java MQ endpoint requests.....	357	Exporting event console output.....	469
Testing all operations in a WSDL file.....	361	Viewing resource monitoring data.....	470
Viewing message content.....	362	Reports and counters.....	471
Synchronizing a remote WSDL file.....	363	Requirements report.....	471
Synchronizing a local WSDL file with GSC.....	364	Synchronization Point report.....	473
Adding static XML headers to a service request.....	364	Loops report.....	473
Opening file attachments.....	365	Agents Health Report.....	473
Chapter 7. Test Execution Specialist Guide.....	366	Rate Runner report.....	474
Running schedules with performance testing.....	366	Transaction report.....	475
Running a local schedule or test.....	366	Transaction Percentile report.....	477
Setting a launch configuration.....	367	Transaction Net Server Time Percentile report.....	478
Running a configured schedule.....	369	Rate Generator report.....	479
Configuring multiple host names for a location.....	369	HTTP performance test reports.....	480
Automating tests from command line.....	370	SAP performance test reports.....	503
Controlling caches size.....	400	Citrix performance test reports.....	507
Increasing memory allocation.....	401	Web service performance test reports.....	512
Extending test execution with custom code.....	402	Socket performance test reports.....	520
Creating custom Java™ code.....	402	HTTP counters.....	522
Test execution services interfaces and classes.....	405	SAP counters.....	541
		Citrix counters.....	548
		Service counters.....	558
		Socket counters.....	579

Chapter 9. Troubleshooting Guide.....	583	RPHE0125W.....	622
Troubleshooting performance testing.....	583	RPIB0007E.....	623
Performance testing error messages.....	587	RPIH0014E.....	623
PRXE0101W.....	588	RPKG0090E.....	624
PRXE4943W.....	588	RPKG0100E.....	624
PRXE4951I.....	588	RPKG0101E.....	625
RMSE0003W.....	588	RPKG0110E.....	625
RMSE0004W.....	589	RPSE0014W.....	626
RMSE0005W.....	589	RPSF0114E.....	626
RMSE0006W.....	590	RPSF0172E.....	627
RPAC0001W.....	590	RPSF0195E.....	627
RPHD1032E.....	591	RPSF0197E.....	628
RPHD1034E.....	593	RPSF0198E.....	628
RPHE0001E.....	594	RPSF0199E.....	629
RPHE0116E.....	594	RPTA0000W.....	629
RPHE0010W.....	595	RPTA0001I.....	629
RPHE0011W.....	595	RPTA0002E.....	629
RPHE0012W.....	595	RPTA0003E.....	629
RPHE0013W.....	596	RPTA0004E.....	630
RPHE0014W.....	596	RPTA0009E.....	631
RPHE0100W.....	597	RPTA0010E.....	632
RPHE0101W.....	598	RPTA0011E.....	633
RPHE0102W.....	599	RPTA0012E.....	634
RPHE0103W.....	600	RPTA0013E.....	635
RPHE0104W.....	600	RPTA0014E.....	636
RPHE0105W.....	601	RPTA0015E.....	637
RPHE0106W.....	602	RPTA0016E.....	638
RPHE0107W.....	604	RPTA0017E.....	639
RPHE0108W.....	606	RPTA0018E.....	639
RPHE0109W.....	607	RPTA0019E.....	639
RPHE0110W.....	608	RPTA0020E.....	640
RPHE0111W.....	609	RPTA0021E.....	640
RPHE0112W.....	610	RPTA0022E.....	641
RPHE0113E.....	611	RPTA0023E.....	643
RPHE0113W.....	612	RPTA0024E.....	644
RPHE0114E.....	612	RPTA0025E.....	645
RPHE0114W.....	613	RPTA0026E.....	646
RPHE0115E.....	613	RPTA0025I.....	647
RPHE0115W.....	614	RPTA0026I.....	647
RPHE0117W.....	615	RPTA0027I.....	647
RPHE0118W.....	616	RPTA0031E.....	647
RPHE0119E.....	617	RPTA0032I.....	648
RPHE0120E.....	618	RPTA0033I.....	648
RPHE0121E.....	619	RPTA0034E.....	648
RPHE0122W.....	620	RPTA0035E.....	649
RPHE0123W.....	621	RPTA0036E.....	649
RPHE0124W.....	622	RPTA0037E.....	649

RPTA0038E.....	650	RPTI0075E.....	668
RPTA0039E.....	650	RPTI0110I.....	668
RPTA0040E.....	650	RPTI0111I.....	668
RPTA0041E.....	651	RPTI0112I.....	669
RPTA0042E.....	651	RPTI0113I.....	669
RPTA0043E.....	651	RPTI0141E.....	669
RPTA0100W.....	652	RPTI0142E.....	670
RPTA0518E.....	652	RPTI0143E.....	670
RPTA1050E.....	653	RPTI0144W.....	671
RPTC0003E.....	654	RPTI0145E.....	672
RPTC0004E.....	654	RPTI0146E.....	673
RPTC0005E.....	655	RPTJ0063E.....	673
RPTC0006E.....	655	RPTJ0075E.....	674
RPTC0007E.....	656	RPTJ1002E.....	674
RPTC0008I.....	656	RPTJ1003E.....	675
RPTC00020E.....	657	RPTJ1004E.....	675
RPTC1001W.....	657	RPTJ1005E.....	676
RPTC1002W.....	657	RPTJ1006E.....	677
RPTC1009I.....	657	RPTJ1007E.....	678
RPTC1011I.....	658	RPTJ1008E.....	680
RPTC1012I.....	658	RPTJ1009E.....	681
RPTC1013I.....	658	RPTJ1010E.....	681
RPTC1014I.....	658	RPTJ1011E.....	682
RPTC1015I.....	659	RPTJ1012E.....	683
RPTC1016I.....	659	RPTJ1013E.....	684
RPTC1017I.....	659	RPTJ1014E.....	685
RPTC1018I.....	659	RPTJ1015E.....	687
RPTC1019I.....	659	RPTJ1016E.....	688
RPTC1020I.....	659	RPTJ1017E.....	689
RPTC1021I.....	660	RPTJ1018E.....	690
RPTC1030E.....	660	RPTJ1019E.....	691
RPTC1031E.....	660	RPTJ1020E.....	692
RPTC1032E.....	661	RPTJ1021E.....	693
RPTD0011W.....	661	RPTJ0121I.....	694
RPTE0005W.....	662	RPTJ1022E.....	694
RPTE0011W.....	663	RPTJ1023E.....	695
RPTE0147E.....	664	RPTJ1024E.....	696
RPTE0150E.....	665	RPTJ1025I.....	696
RPTH0130I.....	665	RPTJ1026I.....	696
RPTH049E.....	666	RPTJ1030E.....	696
RPTI0069E.....	666	RPTJ1040E.....	697
RPTI0070E.....	666	RPTJ1041E.....	698
RPTI0071I.....	667	RPTJ1042E.....	698
RPTI0072E.....	667	RPTJ1043E.....	699
RPTI0072I.....	667	RPTJ1044E.....	700
RPTI0073E.....	667	RPTJ1100I.....	701
RPTI0074E.....	668	RPTJ1101E.....	701

RPTJ1102W.....	702	RPTS1001I.....	722
RPTJ1103W.....	703	RPTS1004I.....	723
RPTJ1104E.....	703	RPTX0001E.....	724
RPTJ1141E.....	704	RPTX0002E.....	725
RPTJ1142E.....	704	RPTX0003E.....	725
RPTJ1200W.....	704	RPTX0004E.....	726
RPTJ1220E.....	705	RPTX0005E.....	726
RPTJ1221E.....	705	RPTX0006E.....	727
RPTJ1240E.....	706	RPTX0007E.....	727
RPTJ1241E.....	707	RPTX0008E.....	728
RPTJ1242E.....	707	RPTX0009E.....	728
RPTJ1244E.....	708	RPTX0010E.....	728
RPTJ1245E.....	709	RPXD0022W.....	729
RPTJ1261E.....	710	RPXE0061I.....	729
RPTJ1270E.....	710	RPXE5502E.....	730
RPTJ1271E.....	710	RPTX1010I.....	730
RPTJ1280E.....	711	RPTX1011I.....	730
RPTJ1400I.....	711	RPTX1012I.....	731
RPTK0000I.....	711	RPTX1017I.....	731
RPTK1001E.....	712	RPTX1018I.....	731
RPTK1016E.....	713	RPTX1019I.....	731
RPTK1019E.....	713	RPTX1081E.....	732
RPTK1020E.....	714	RPTX1082E.....	732
RPTK1021E.....	714	RPTX2001E.....	733
RPTK1022E.....	714	RPTX2002E.....	734
RPTK1023E.....	715	RPTX2003E.....	734
RPTL0001W.....	715	RPTX2004E.....	735
RPTL0002W.....	715	RPTX2005E.....	735
RPTL0003W.....	716	RPTX2006W.....	736
RPTL0004W.....	716	RPTX2007I.....	736
RPTL0005W.....	716	RPTX2008I.....	737
RPTL0006W.....	717	RPTX2009I.....	737
RPTL0007W.....	717	RPTX2010I.....	737
RPTL0008E.....	717	RPTX2011E.....	738
RPTL0009I.....	718	RPTX2012E.....	739
RPTL0010E.....	718	RPTX2013E.....	739
RPTL0011E.....	718	RPTX2014E.....	740
RPTR0000W.....	718	RPTX2015E.....	740
RPTR0001W.....	718	RPTX2016I.....	741
RPTR0002W.....	719	RPTX2017E.....	741
RPTR0003W.....	719	RPTX2018W.....	741
RPTR0004W.....	719	RPTX2019I.....	741
RPTR2001E.....	719	RPTX2020I.....	741
RPTR2003W.....	720	RPTX2021E.....	741
RPTS1000E.....	720	RPTX2022E.....	742
RPTS1002E.....	721	RPTX2023W.....	742
RPTS1510E.....	722	RPTX2024E.....	743

RPTX2025E.....	743	RPWF0101E.....	768
RPTX2026E.....	744	RPWF0102E.....	769
RPTX2027W.....	744	RPWF0103E.....	769
RPTX2029W.....	745	RPWF0104E.....	769
RPTX2030I.....	745	RPWF0111E.....	769
RPTX2031I.....	746	RPWF0112E.....	770
RPTX2032I.....	746	RPWF0121W.....	770
RPTX2033E.....	747	RPWF0122W.....	770
RPTX2034E.....	747	RPWF0123W.....	770
RPTX2035E.....	748	RPWF0124W.....	771
RPTX2036E.....	748	RPWF0130W.....	771
RPTX2037E.....	749	RPWF0131W.....	771
RPTX2050E.....	749	RPWF0132E.....	771
RPTX2051E.....	750	RPWF0140E.....	772
RPTX2055E.....	750	RPWH0007W.....	772
RPTX2056E.....	751	RPWH0009W.....	772
RPTX2057E.....	751	RPWH0010W.....	772
RPTX2058E.....	752	RPWH0012E.....	772
RPTX2060E.....	753	RPWH0014E.....	772
RPTX2061W.....	754	RPWH0015E.....	773
RPTX2062W.....	755	RPWH0016E.....	773
RPTX2063W.....	756	RPWH0017E.....	773
RPTX2070E.....	757	RPWS0001E.....	774
RPTX2071E.....	758	RPWS0002E.....	774
RPTX2072E.....	758	RPWS0003E.....	774
RPTX2073E.....	759	RPWS0004E.....	775
RPTX2074E.....	760	RPWS0005E.....	775
RPTX2075E.....	761	RPWS0006E.....	775
RPTX2077E.....	762	RPWS0007E.....	776
RPWF0011E.....	763	RPWS0008E.....	776
RPWF0012E.....	764	RPWY0002E.....	776
RPWF0021E.....	764	RPWY0003I.....	777
RPWF0032E.....	764	RPWY0004W.....	777
RPWF0051E.....	764	RPWY0005E.....	777
RPWF0052E.....	765	RPWY0006E.....	777
RPWF0056E.....	765	RPWY0007E.....	777
RPWF0066E.....	765	RPWZI0002E.....	778
RPWF0071E.....	766	RPXD0001E.....	778
RPWF0072E.....	766	RPXD0002E.....	778
RPWF0074E.....	766	RPXD0003E.....	778
RPWF0075E.....	766	RPXD0004E.....	779
RPWF0076W.....	767	RPXD0005E.....	779
RPWF0081W.....	767	RPXD0006E.....	779
RPWF0082W.....	767	RPXD0007F.....	779
RPWF0083E.....	768	RPXD0017W.....	780
RPWF0084E.....	768	RPXD0018E.....	780
RPWF0085E.....	768	RPXD0019E.....	781

RPXD0020E.....	783	RPXE0058E.....	795
RPXD0021E.....	784	RPXE0059E.....	795
RPXD0021W.....	784	RPXE0060E.....	796
RPXE0001W.....	784	RPXE0100W.....	796
RPXE0010W.....	785	RPXE0102W.....	796
RPXE0011W.....	785	RPXE0103W.....	797
RPXE0012W.....	785	RPXE0104W.....	797
RPXE0013W.....	785	RPXE2501E.....	798
RPXE0014W.....	785	RPXE2550E.....	799
RPXE0015W.....	786	RPXE2552I.....	799
RPXE0016W.....	786	RPXE2900E.....	799
RPXE0017W.....	786	RPXE2901W.....	800
RPXE0018W.....	787	RPXE4000W.....	801
RPXE0019W.....	787	RPXE4001E.....	801
RPXE0021W.....	787	RPXE4002E.....	801
RPXE0023W.....	787	RPXE4003E.....	801
RPXE0024W.....	787	RPXE4004E.....	802
RPXE0025W.....	788	RPXE4005E.....	802
RPXE0027W.....	788	RPXE4006E.....	802
RPXE0028W.....	788	RPXE4007E.....	803
RPXE0029W.....	788	RPXE4008E.....	803
RPXE0030W.....	788	RPXE4008I.....	804
RPXE0031W.....	788	RPXE4009I.....	804
RPXE0033W.....	789	RPXE4010I.....	805
RPXE0035W.....	789	RPXE4011E.....	805
RPXE0036W.....	789	RPXE4013I.....	805
RPXE0037W.....	789	RPXE4014E.....	806
RPXE0038W.....	790	RPXE4015E.....	806
RPXE0039W.....	790	RPXE4016E.....	806
RPXE0040W.....	790	RPXE4017I.....	807
RPXE0041W.....	790	RPXE4018E.....	808
RPXE0042I.....	791	RPXE4019E.....	808
RPXE0043I.....	791	RPXE4020E.....	808
RPXE0044W.....	791	RPXE4021E.....	809
RPXE0045W.....	791	RPXE4022E.....	809
RPXE0046W.....	792	RPXE4023E.....	809
RPXE0047E.....	792	RPXE4024E.....	810
RPXE0048W.....	792	RPXE4025E.....	810
RPXE0049W.....	793	RPXE4026E.....	810
RPXE0050W.....	793	RPXE4027E.....	810
RPXE0051W.....	793	RPXE4028E.....	811
RPXE0052W.....	794	RPXE4029E.....	811
RPXE0053W.....	794	RPXE4050I.....	812
RPXE0054W.....	794	RPXE4100W.....	812
RPXE0055W.....	794	RPXE4101E.....	812
RPXE0056W.....	795	RPXE4102E.....	813
RPXE0057E.....	795	RPXE4103E.....	813

RPXE4104E.....	813	RPXE4912I.....	829
RPXE4105E.....	814	RPXE4913I.....	829
RPXE4106E.....	814	RPXE4914I.....	829
RPXE4107E.....	814	RPXE4915I.....	830
RPXE4108E.....	815	RPXE4916I.....	830
RPXE4109E.....	815	RPXE4917I.....	830
RPXE4110E.....	815	RPXE4918I.....	830
RPXE4111W.....	816	RPXE4920I.....	831
RPXE4112W.....	816	RPXE4921I.....	832
RPXE4120E.....	817	RPXE4930I.....	833
RPXE4150E.....	817	RPXE4931I.....	833
RPXE4151E.....	817	RPXE4932I.....	833
RPXE4152E.....	817	RPXE4940I.....	834
RPXE4153E.....	818	RPXE4941I.....	834
RPXE4200W.....	818	RPXE4942I.....	834
RPXE4201W.....	818	RPXE4944W.....	834
RPXE4202E.....	819	RPXE4945W.....	835
RPXE4203E.....	819	RPXE4948W.....	835
RPXE4204W.....	819	RPXE4950I.....	835
RPXE4205E.....	820	RPXE4952E.....	835
RPXE4208E.....	820	RPXE5301E.....	836
RPXE4209I.....	820	RPXE5305E.....	836
RPXE4210E.....	821	RPXE5330E.....	836
RPXE4211E.....	821	RPXE5500W.....	837
RPXE4212E.....	821	RPXE5501W.....	838
RPXE4213E.....	822	RRIT0001E.....	839
RPXE4214W.....	822	RRIT0002E.....	839
RPXE4215E.....	822	RRIT0003E.....	840
RPXE4215I.....	823	RRIT0004E.....	840
RPXE4216E.....	823	RRIT0005E.....	840
RPXE4217E.....	824	RRIT0006E.....	841
RPXE4218E.....	825	RRIT0007E.....	841
RPXE4219E.....	825	RRITUI1002W.....	841
RPXE4220E.....	826	DCRC0001E.....	842
RPXE4221E.....	826	DCRC0002E.....	842
RPXE4900I.....	827	DCRC0003E.....	842
RPXE4901I.....	827	DCRC0008W.....	842
RPXE4902I.....	827	DCRC0009W.....	843
RPXE4903I.....	827	DCRC0010E.....	843
RPXE4904I.....	827	DCUI0001E.....	843
RPXE4905I.....	828	DCUI0003E.....	844
RPXE4906I.....	828	DCUI0004E.....	844
RPXE4907I.....	828	DCUI0006E.....	844
RPXE4908I.....	828	DCUI0007W.....	844
RPXE4909I.....	828	DCUI0008W.....	844
RPXE4910I.....	828	DCUI0009E.....	845
RPXE4911I.....	829	DCUI0010E.....	845

DCUI0011E.....	845
DCUI0012E.....	846
DCUI0013E.....	846
DCUI0014E.....	846
DCUI0015E.....	846
DCUI0016E.....	846
DCUI0017E.....	847
DCUI0998E.....	847
Additional error messages.....	847
Address already in use.....	847
Browser profile in use.....	848
Cannot open test.....	849
Connection closed.....	850
Dataset accessed using different modes.....	850
Error binding to port.....	851
No local agent controller.....	851
Performance Test Errors were found in the project.....	852
Test run aborted.....	853
Test run aborted due to error.....	853
Testgen completed with warnings.....	854
Variable not initialized.....	855
Chapter 10. Reference Guide.....	856
Accessibility features.....	856
Keyboard shortcuts for performance and service testing.....	856
General reference for performance testing.....	858
Data correlation rules.....	858
Error conditions.....	860
Resource monitoring data sources.....	861
Response time breakdown data sources.....	863
UI preferences.....	863
VU Schedule editor reference.....	863
Generic service client references.....	871
WSDL security editor reference.....	900
Security Considerations.....	cmvi
Notices.....	cmvii
Index.....	911

Chapter 1. Release notes for IBM® Rational® Service Tester for SOA Quality

This document contains information about what's new, installation instructions, known issues in Rational® Service Tester for SOA Quality, and contact information of IBM Customer Support.

Contents

- [Product description on page 12](#)
- [What's new on page 12](#)
- [Deprecated capabilities on page 13](#)
- [Removed capabilities on page 14](#)
- [Installing the product on page 14](#)
- [Known issues on page 14](#)
- [Contacting IBM Rational Software Support on page 14](#)

Product description

You can find the description of Rational® Service Tester for SOA Quality.

The service testing capabilities of Rational® Service Tester for SOA Quality automate the creation, execution, and analysis of functional, regression and performance tests for SOAP-based web services. The SOAP-based web services includes support for Java Messaging Service (JMS), WebSphere MQ, WebSphere Java MQ, and Microsoft .NET Windows Communication Foundation (WCF), and any service that produces XML, plain text, or binary data.

See [Rational Service Tester for SOA Quality overview on page 29](#).

What's new

You can find information about the features introduced in this release of Rational® Service Tester for SOA Quality.

• **Error reports for dropped connections**

Previously, during test runs or schedules, the system under test occasionally dropped the connection, however, the error was reported as a server timeout. Rational® Service Tester for SOA Quality now reports the error condition for a dropped connection separately. See [Error conditions on page 860](#).

• **Support to view the type of correlation for Test References**

You can now view the type of correlation that is configured for Test References from the Type column on the **Test Editor -Test References page** for HTTP tests. You can also view the reference type in the **Details** tab on

the **Test Element properties** page. (**Data Correlation > Show References > Test Editor- Test References > Properties > Test Element Properties > Details**).

The types of correlation are Auto, User, or Rule.

See Viewing references.

- **Changed the assets.xml to assets.yaml**

Rational® Service Tester for SOA Quality now writes details of the assets to the assets.yaml file, which was previously written to the assets.xml file. You can view the assets.xml file is deleted and new assets.yaml file is added. You can now easily check in and merge the assets.yaml file when there is a conflict.

- **Support for Instana integration**

You can now integrate Instana with Rational® Service Tester for SOA Quality to enhance data collection in performance tests. When you select the Enable Instana option, Rational Performance Tester filters the HTTP headers and adds them to the request in tests to collate data about the pages and transactions.

See Using Application Performance Management in a schedule.

- **Enhancement to an IF condition comparison**

Rational® Service Tester for SOA Quality is now enhanced to perform a digit comparison when you apply an IF condition to a test or a schedule. If the digit comparison in the IF condition is successful, the result is displayed as true. Otherwise, it is displayed as false. See Adding a conditional logic.

- **Enhancement to the command line option**

Previously, to run all the test assets from the specified folder path, you created a batch file and ran the test from the command line. The command line option is now enhanced so that you can run multiple test assets at a time by using the wildcard character * enclosed in quotes. You can use * in different combinations to specify any or all test assets that are in a folder. See [Running multiple tests from the command line on page 397](#).

- **Support for TLS 1.3**

Rational® Service Tester for SOA Quality now supports TLS 1.3. You can record websites that support TLS 1.3 and play back the recorded test.

- **Bug fixes**

Fixed the customer-reported and internally found defects.

Deprecated capabilities

You can find information about the capabilities that are deprecated in this release and are planned for a removal in a future release.



Note: You can contact the IBM Client Support team if you have any concerns.

- No capabilities were deprecated in this release.

Removed capabilities

No capabilities were removed in this release.

Installing the product

You can find information about the installation and upgrading information for Rational® Service Tester for SOA Quality.

Installing Rational® Service Tester for SOA Quality

For installation instructions, see [Installing IBM Rational Service Tester for SOA Quality on page 52](#).

Upgrading

- You can upgrade to Rational® Service Tester for SOA Quality 10.5.2 from 10.5.0 only.
- If you installed Rational® Service Tester for SOA Quality that is earlier than 10.5.0, then you cannot upgrade the product to the latest version. You must uninstall the existing version of the product before you install the latest version of the product.
- After you install Rational® Service Tester for SOA Quality 10.5.2, at any point in time if you want to use the previous version of the product, you cannot roll back to the previous version. If you want to use the previous version of the product, you must uninstall the existing version, and then install the required version of the product.

Known issues

You can find information about the known issues identified in this release of Rational® Service Tester for SOA Quality.

Table 1. Release documents - Fix list and known issues

Product	Download document	Knowledge Base
Rational® Service Tester for SOA Quality	Release document	Knowledge articles

Known problems are documented in the download document for each product and in the form of individual technotes in the Support Knowledge Base:

The knowledge base is continually updated as problems are discovered and resolved. By searching the knowledge base, you can quickly find workarounds or solutions to problems.

Contacting IBM Rational Software Support

You can find information about IBM technical support assistance for Rational® Service Tester for SOA Quality.

- For contact information and guidelines or reference materials that you might need when you require support, read the [IBM Support Guide](#).
- For personalized support that includes notifications of significant upgrades, subscribe to [Product notification](#).
- Before you contact IBM Rational Software Support, you must gather the background information that you might need to describe your problem. When you describe a problem to an IBM software support specialist, be as specific as possible and include all relevant background information so that the specialist can help you solve the problem efficiently. To save time, know the answers to these questions:
 - What software versions were you running when the problem occurred?
 - Do you have logs, traces, or messages that are related to the problem?
 - Can you reproduce the problem? If so, what steps do you take to reproduce it?
 - Is there a workaround for the problem? If so, be prepared to describe the workaround.

Chapter 2. System Requirements

This document includes information about hardware and software requirements for IBM® Rational® Service Tester for SOA Quality.

Contents

- [Hardware on page 17](#)
 - [AIX on page 17](#)
 - [Linux on page 18](#)
 - [Mac on page 19](#)
 - [Windows on page 19](#)
- [Operating Systems on page 20](#)
 - [AIX on page 21](#)
 - [Linux on page 21](#)
 - [Mac on page 22](#)
 - [Windows on page 22](#)
- [Host prerequisites on page 22](#)
 - [Licensing on page 23](#)
 - [Terminal services on page 23](#)
 - [Virtualization Management on page 23](#)
 - [Web Browsers on page 24](#)
- [Recording support on page 24](#)
- [Supported software on page 25](#)
 - [Application servers on page 25](#)
 - [Business process management on page 26](#)
 - [Development tools on page 26](#)
 - [DevOps tools on page 27](#)
 - [Message Oriented Middleware on page 28](#)

Disclaimers

This report is subject to the Terms of Use (<https://www.ibm.com/legal/us/en/>) and the following disclaimers:

The information contained in this report is provided for informational purposes only. While efforts were made to verify the completeness and accuracy of the information contained in this publication, it is provided AS IS without warranty of any kind, express or implied, including but not limited to the implied warranties of merchantability, non-infringement, and fitness for a particular purpose. In addition, this information is based on IBM's current product plans and strategy, which are subject to change by IBM without notice. IBM shall not be responsible for any direct, indirect, incidental, consequential, special or other damages arising out of the use of, or otherwise related to, this report or any other materials. Nothing contained in this publication is intended to, nor shall have the effect of, creating any warranties or representations from IBM or its suppliers or licensors, or altering the terms and conditions of the applicable license agreement governing the use of IBM software.

References in this report to IBM products, programs, or services do not imply that they will be available in all countries in which IBM operates. Product release dates and/or capabilities referenced in this presentation may change at any time at IBM's sole discretion based on market opportunities or other factors, and are not intended to be a commitment to future product or feature availability in any way. The underlying database used to support these reports is refreshed on a weekly basis. Discrepancies found between reports generated using this web tool and other IBM documentation sources may or may not be attributed to different publish and refresh cycles for this tool and other sources. Nothing contained in this report is intended to, nor shall have the effect of, stating or implying that any activities undertaken by you will result in any specific sales, revenue growth, savings or other results. You assume sole responsibility for any results you obtain or decisions you make as a result of this report.

Notwithstanding the Terms of Use (<https://www.ibm.com/legal/us/en/>), users of this site are permitted to copy and save the reports generated from this tool for such users own internal business purpose. No other use shall be permitted.

Hardware

You can find information about the hardware requirements for IBM® Rational® Service Tester for SOA Quality

Contents

- [AIX on page 17](#)
- [Linux on page 18](#)
- [Mac on page 19](#)
- [Windows on page 19](#)

AIX

Hardware	Components	Requirement	Notes
Disk space	Rational® Performance Tester Agent	10 GB	<ul style="list-style-type: none"> • Disk space requirements can be reduced or increased depending on the features that you install. • Large test runs can store several gigabytes of data. Make sure that you have adequate disk space before attempting a large test run.

Hardware	Components	Requirement	Notes
Memory	Rational® Performance Tester Agent	8 GB	<ul style="list-style-type: none"> • For best results with large test runs, use 16 GB of RAM. • For best results with large test runs, use a modern processor with 2.30 GHz or higher that supports the supported Operating system.

Linux

Hardware	Components	Requirement	Notes
Disk space	Rational® Service Tester for SOA Quality	10 GB	<ul style="list-style-type: none"> • Disk space requirements can be reduced or increased depending on the features that you install. • An additional 500 MB of disk space is required in the /tmp directory.
	Rational® Performance Tester Agent		
Memory	Rational® Service Tester for SOA Quality	8 GB	<ul style="list-style-type: none"> • For best results with large test runs, use 16 GB of RAM. • For best results with large test runs, use a modern processor with 2.30 GHz or higher that supports the supported Operating system
	Rational® Performance Tester Agent		

Mac

Hardware	Components	Requirement	Notes
Disk space	Rational® Service Tester for SOA Quality	10 GB	<ul style="list-style-type: none"> • Disk space requirements can be reduced or increased depending on the features that you install. • Large test runs can store several gigabytes of data. Make sure that you have adequate disk space before attempting a large test run.
	Rational® Performance Tester Agent		
Memory	Rational® Service Tester for SOA Quality	8 GB	<ul style="list-style-type: none"> • For best results with large test runs, use 16 GB of RAM. • For best results with large test runs, use a modern processor with 2.30 GHz or higher that supports the supported Operating system
	Rational® Performance Tester Agent		

Windows

Hardware	Components	Requirement	Notes
Disk space	Rational® Service Tester for SOA Quality	10 GB	<ul style="list-style-type: none"> • Disk space requirements can be reduced or increased depending on the features that you install. • Large test runs can store several gigabytes of data. Make sure that you have adequate disk space before attempting a large test run. • Additional disk space is required if you use FAT32 instead of NTFS. An additional 500 MB of disk
	Rational® Performance Tester Agent		

Hardware	Components	Requirement	Notes
			space is required in the directory that you specify in the environment variable TEMP.
Memory	Rational® Service Tester for SOA Quality	8 GB	<ul style="list-style-type: none"> • For best results with large test runs, use 16 GB of RAM. • For best results with large test runs, use a modern processor with 2.30 GHz or higher that supports the supported Operating system.
	Rational® Performance Tester Agent		

Operating systems

You can find the operating systems that are supported, organized by operating system family for IBM® Rational® Service Tester for SOA Quality.

Contents

- [AIX on page 21](#)
- [Linux on page 21](#)
- [Mac on page 22](#)
- [Windows on page 22](#)

Bit version support

Different parts of a product might run on the same operating system but support different application bitness. For example, one part of the product might run only in 32-bit mode, whereas another might support 64-bit tolerate mode.

Bitness	Description
32	The product or part of the product runs as a 32-bit application in the 32-bit platforms listed as supported.
64-Tolerate	The product or part of the product runs as a 32-bit application in the 64-bit platforms listed as supported.
64-Exploit	The product or part of the product runs as a 64-bit application in the 64-bit platforms listed as supported.

AIX

Operating system	Hardware	Bitness	Components	
			Desktop	Agent
AIX 7.2 TL5	POWER System - Big Endian	32	✗	✓
AIX 7.1 TL5	POWER System - Big Endian	32	✗	✓

Linux

Operating system	Hardware	Bitness	Components	
			Desktop	Agent
Red Hat Enterprise Linux (RHEL) 9.0	x86-64	64-Exploit	✓	✓
Red Hat Enterprise Linux (RHEL) 8.6	x86-64	64-Exploit	✓	✓
Red Hat Enterprise Linux (RHEL) 8.4	x86-64	64-Exploit	✓	✓
Red Hat Enterprise Linux (RHEL) 7.9	x86-64	64-Exploit	✓	✓
Ubuntu 22.04.1 LTS	x86-64	64-Exploit	✓	✓
Ubuntu 20.04.1 LTS	x86-64	64-Exploit	✓	✓
Ubuntu 18.04 LTS	x86-64	64-Exploit	✓	✓

Mac

Operating system	Hardware	Bitness	Components	
			Desktop	Agent
macOS Monterey 12	x86-64	64-Exploit	✓	✓
macOS BigSur 11.6	x86-64	64-Exploit	✓	✓

Windows

Operating system	Hardware	Bitness	Components	
			Desktop	Agent
Windows 11	x86-64	32, 64-Exploit	✓	✓
Windows 10 Enterprise	x86-64	32, 64-Exploit	✓	✓
Windows 10 Pro	x86-64	32, 64-Exploit	✓	✓
Windows Server 2022	x86-64	32, 64-Exploit	✓	✓
Windows Server 2019	x86-64	32, 64-Exploit	✓	✓
Windows Server 2016	x86-64	32, 64-Exploit	✓	✓

Host prerequisites

You can find the host prerequisites that support the operating capabilities for IBM® Rational® Service Tester for SOA Quality

Contents

- [Licensing on page 23](#)
- [Terminal services on page 23](#)

- [Virtualization Management on page 23](#)
- [Web Browsers on page 24](#)

Licensing

License Server	Ver- sion	Components	
		Desk- top	Agent
IBM Common License Key Server	9.0	✓	Not applicable

Terminal services

Supported software	Ver- sion	Components		Notes
		Desk- top	Agent	
Citrix Receiver	4.9	✓	Not applicable	For remote terminal ac- cess
Citrix XenApp	6.5	✓	Not applicable	

Virtualization Management

Containers	Ver- sion	Components	
		Desk- top	Agent
Docker Community Edition (CE)	20.10	✓	✓
Docker Compose	2.15.0	✓	✓
	2.12	✓	✓

Containers	Version	Components	
		Desktop	Agent
	2.10	✓	✓
	2.6	✓	✓

Web Browsers

The following versions of web browsers support the viewing of performance reports and datasets. See [Recording support on page 24](#) to know the browsers that are supported to record the HTTP tests.

Browsers	Version	Components	
		Desktop	Agent
Apple Safari	15 or later	✓	Not applicable
Google Chrome	101 or later	✓	Not applicable
Microsoft Edge	101 or later	✓	Not applicable
Mozilla Firefox (including ESR versions)	100 or later	✓	Not applicable

Recording support

You can find information about the web browsers that support recording capability of HTTP tests for IBM® Rational® Service Tester for SOA Quality

Web browsers

The following versions of web browsers support the recording of HTTP tests. See [Web Browsers on page 24](#) to know the browsers that are supported to view the performance reports.

Supported Browsers	Version	Desktop
Apple Safari	15 to 16	✓
Google Chrome	101 to 111	✓
Microsoft Edge	101 to 110	✓
Mozilla Firefox	100 to 110	✓
Mozilla Firefox ESR	102	✓
	91	✓

Supported software

You can find the additional software that is supported for IBM® Rational® Service Tester for SOA Quality.

Contents

- [Application servers on page 25](#)
- [Business process management on page 26](#)
- [Development tools on page 26](#)
- [DevOps tools on page 27](#)
- [Message Oriented Middleware on page 28](#)

Application servers

Support for the following application servers is in reference only to the HTTP Response Time Break Down capability:

Supported software	Version	Components		Notes
		Desk-top	Agent	
IBM® WebSphere Application Server	9.0	✓	Not applicable	To collect response time breakdown data
	8.5.5	✓	Not applicable	

Supported software	Version	Components		Notes
		Desk-top	Agent	
IBM® WebSphere Liberty	17.0.0.1	✓	Not applicable	

Business process management

Supported software	Version	Components		Notes
		Desk-top	Agent	
SAP GUI	7.7	✓	✓	To record and playback tests of SAP applications built with the SAP GUI client
	7.6	✓	✓	

Development tools

Supported software	Version	Components		Note
		Desk-top	Agent	
eGit	5.13	✓	Not applicable	To store your test assets in the remote Git repository and use eGit for version control operations.
IBM® Engineering Test Management	7.0.2	✓	Not applicable	To initiate the test runs from Engineering Test Management
	7.0.1	✓		
IBM® Engineering Workflow Management	7.0.2	✓	Not applicable	To perform integrations with Engineering Workflow Management
	7.0.1	✓		

Supported software	Version	Components		Note
		Desktop	Agent	
IBM® Rational® Functional Tester	10.5.2	✓	Not applicable	Eclipse shell sharing and to run WebUI integrations
Rational® Test Workbench	10.5.2	✓	Not applicable	To integrate and run Rational® Integration Tester tests
JMeter	5.4.3	✓	Not applicable	To integrate and run JMeter tests

DevOps tools

Supported software	Version	Components		Notes
		Desktop	Agent	
Apache Ant	1.9 or later	✓	Not applicable	To initiate the test runs from ANT
Azure DevOps	Latest	✓	Not applicable	To initiate the test runs from Azure DevOps pipeline
HCL Launch	7.3.0	✓	Not applicable	To initiate the test runs from HCL Launch
	7.2.3.0	✓	Not applicable	
	7.2.1.0	✓	Not applicable	
IBM® UrbanCode Deploy	7.3.0	✓	Not applicable	To initiate the test runs from UrbanCode Deploy
	7.2.3.0	✓	Not applicable	To initiate the test runs from UrbanCode Deploy

Supported software	Version	Components		Notes
		Desk-top	Agent	
	7.2.1.0	✓	Not applicable	
Jenkins	2.39	✓	Not applicable	To initiate the test runs from Jenkins
	2.375.2	✓	Not applicable	
	2.361	✓	Not applicable	
	2.346.2	✓	Not applicable	
	2.339	✓	Not applicable	
Maven	3.5 or later	✓	Not applicable	To initiate the test runs from Maven
Microfocus ALM	12.6	✓	Not applicable	To initiate the test runs from Microfocus ALM
Github Actions	Latest	✓	Not applicable	To initiate the test runs from Github Actions

Message Oriented Middleware

Supported software	Version	Components	
		Desk-top	Agent
IBM® MQ	9.2.1	✓	✓
	9.1	✓	✓
	9.0.5	✓	✓

Chapter 3. Getting Started

This guide provides an overview to get you started with Rational® Service Tester for SOA Quality. This guide is intended for new users.

You can find the following information:

- [Rational Service Tester for SOA Quality overview on page 29](#)
- [Streamlined Eclipse and full Eclipse overview on page 36](#)

Rational® Service Tester for SOA Quality overview

You can gain the conceptual understanding of Rational® Service Tester for SOA Quality and its test extensions with these topics.

You can find the following information:

- [Service testing overview on page 29](#)
- [Generic service client overview on page 31](#)
- [Socket API performance testing overview on page 33](#)
- [TN3270 performance testing overview on page 33](#)
- [IBM Engineering Test Management overview on page 34](#)

Service testing overview

The service testing capabilities of IBM® Rational® Performance Tester or IBM® Rational® Service Tester for SOA Quality automate the creation, execution and analysis of functional, regression and performance tests for SOAP-based web services, including support for Java Message Service (JMS), Websphere MQ, WebSphere Java MQ, and Microsoft .NET Windows Communication Foundation (WCF), as well as any service that produces XML, plain text, or binary data.



Informative test results rely upon sound test development. Each of the following stages contributes to generating meaningful test results:

- **Preparation.** Set up your test environment with the libraries and configuration files required for SOAP-based web services or custom security algorithms. You can import Web Service Description Language (WSDL) definition files and digital certificates that are required by the web services to automatically generate your tests. You can create SOAP security profiles with security algorithms for the web service calls and message returns.
- **Test creation:** Create your test by recording the service requests and responses either with the **generic service client**, or with an existing client or a web browser through a recording proxy. When you start the recording, you interact with the service by performing service requests and receiving responses. You can also create service tests manually or from a synchronous Business Process Execution Language (BPEL) model.

- **Test editing:** After recording, you can edit the requests and responses in the test. You can use XML Schema Description (XSD) documents to facilitate XML edition. You can replace recorded test values with variable test data, or add dynamic data to the test.
- **Functional testing:** You can run the test to ensure that service matches the expected behavior defined in *verification points*. During the run, each verification point is checked and receives a *pass*, *fail* or *inconclusive* status.
- **Performance testing:** If you are using IBM® Rational® Performance Tester, you can specify an execution schedule and user groups to emulate a workload that is generated by a large number of virtual users. Then, you can run the schedule, deploying test execution on virtual users that can be hosted on remote computers. Each virtual user runs an instance of the test client. Response times are measured and recorded. Verification points are checked and recorded.
- **Stub simulation:** Service stubs are functional simulations of an existing service. Service stubs are useful for replacing a service that is unavailable or impractical to use in a test environment. They can also be used to input specific data into a service under test or for prototyping. You can deploy stubs onto a stub server, which can replace the actual server in your test or development environment.
- **Evaluation of results:** You evaluate the results that the tests produce through the performance and verification point reports that are generated during execution. You can also design custom reports by manipulating various counters. Functional reports provide a comprehensive view of the behavior of the service under test. Reports can be exported and archived for validation.

Service testing tools

The following tools are available in the product:

- The **generic service client** enables you to manually perform service requests for a wide variety of transport protocols, authentication configurations and security profiles, making it an extremely versatile service client. It effectively replaces a dedicated client and can be used to record service calls or for manual testing and debugging a service during development. To open the generic service client, click the **Generic Service Client**  toolbar button.
- The **WSDL security editor** allows you to set up sophisticated *algorithm stacks* for your service requests and responses. Algorithm stacks contain digital certificate information and the security algorithms that are applied to messages to perform secure communication with a web service. Algorithm stacks are made of blocks, which can be key definitions, encryption, time stamp, or signature operations which can be associated with any operation in the WSDL file. To open the WSDL security editor, right-click a WSDL file in your workspace and select **Edit WSDL Security** or click the **WSDL Security Editor**  button in the generic service client.
- The **test editor** is where you develop your test. After recording, you can modify the test to add data correlation or verification points. You can also add loops and conditions and you can edit every detail of the service requests.

- The **stub editor** enables you to create service stubs. With the stub editor, you can define multiple input conditions, which are similar to verification points. Each condition triggers a predefined simulated response, which is functionally identical to a response from the simulated service.
- In Rational® Performance Tester, the schedule editor lets you deploy multiple virtual users on local and remote computers to generate a heavy load for performance testing. A schedule typically contains multiple tests and multiple virtual users.

Generic service client overview

The purpose of the generic service client is to send requests to any service that uses an HTTP, JMS, WebSphere® MQ, or Microsoft™ .NET transport. The generic service client also displays the response returned by the service.

The generic service client is useful for debugging or testing a service when you do not have access to a dedicated client to send the request. You can set up a large variety of transport and security configurations for the service, edit the parameters of the request and send attachments.

When a request is successfully invoked, its message return is added to the **Request History**. You can use this feature to look back at results that were produced at different times.

If you are using IBM® Rational® Performance Tester or IBM® Rational® Service Tester for SOA Quality, you can select requests in the **Request History** and click **Generate Test** to generate a test that will replay all the selected requests. You can edit the test to replace recorded test values with variable test data, or add dynamic data correlation to the test. You can also set verification points on the contents of the XML documents in the service response.

Supported services

The generic service client enables you to send requests for many types of services that use the following transport protocols:

- HTTP
- Java™ Message Service (JMS), including JBoss and WebSphere® implementations
- WebSphere® MQ
- Microsoft™ .NET Framework Windows™ Communication Foundation (WCF).



Note: If you are using IBM® Security AppScan®, only the HTTP transport protocol is checkmark_perf.jpg.

Encryption and security

The Java™ Runtime Environment (JRE) that the product uses must support the level of encryption required by the digital certificate that you select. For example, you cannot use a digital certificate that requires 256-bit encryption with a JRE that supports only 128-bit encryption. By default, the product is configured with restricted or limited strength ciphers. To use less restricted encryption algorithms, you must download and apply the unlimited jurisdiction policy files (`local_policy.jar` and `US_export_policy.jar`).

For Oracle Java, download the files from this site:<http://www.oracle.com/technetwork/java/javase/downloads/jce8-download-2133166.html>.

Before installing these policy files, back up the existing policy files in case you want to restore the original files later. Then overwrite the files in `/jre/lib/security/` directory with the unlimited jurisdiction policy files.

SSL Authentication

Service tests support simple or double SSL authentication mechanisms:

- Simple authentication (server authentication): In this case, the test client needs to determine whether the service can be trusted. You do not need to setup a key store. If you select the **Always trust** option, you do not need to provide a server certificate key store.

If you want to really authenticate the service, you can configure a certificate trust store, which contains the certificates of trusted services. In this case, the test will expect to receive a valid certificate.

- Double authentication (client and server authentication): In this case, the service needs to authenticate the test client according to its root authority. You must provide the client certificate keystore that needs to be produced to authenticate the test as a certified client.

When recording a service test through a proxy, the recording proxy sits between the service and the client. In this case, you must configure the SSL settings of the recording proxy to authenticate itself as the actual service to the client (for simple authentication), and as the client to the service (for double authentication). This means that you must supply the recording proxy with the adequate certificates.

When using stub services, you can also configure the SSL settings of the stub service to authenticate itself as the actual server. This means that you must supply the service stub with the adequate certificate.

NTLM and Kerberos Authentication

The product supports Microsoft™ NT LAN Manager (NTLMv1 and NTLMv2) and Kerberos authentication. The authentication information is recorded as part of the test during the recording phase.

To enable NTLMv2 support, you must add a third party library to the workbench. For more information, see [Configuring the workbench for NTLMv2 authentication on page 341](#).

Digital certificates

You can test services with digital certificates for both SSL and SOAP security protocol. Digital certificates must be contained in Java™ Key Store (JKS) keystore resources that are accessible in the workspace. When dealing with keystore files, you must set the password required to access the keys both in the security editor and the test editor. For SOAP security you might have to provide an explicit name for the key and provide a password to access the private keys in the keystore.

Limitations

Arrays are not checkmark_perf.jpg.

Because of a lack of specification, attachments are not checkmark_perf.jpg with the Java™ Message Service (JMS) transport. The envelope is directly sent using UTF-8 encoding.

All security algorithms are not always available for every Java™ Runtime Environment (JRE) implementation. If a particular security implementation is not available, add the required libraries to the class path of the JRE that this product uses.

The Microsoft™ .NET transport protocol does not support transactions, scopes, or duplex mode requests such as callbacks or two-way services based on the MS-MQ transport.

Socket API performance testing overview

With IBM® Rational® Performance Tester Extension for *Socket Protocols*, you can test the performance of any application that uses a TCP/IP socket-based protocol.

Informative performance test results rely upon sound test development. Each of the following stages contributes to the generation of meaningful test results:

- **Test creation.** You create your test by recording a session with a client application. Typically, the recorded session starts when you run the client application. You then interact with the application in order to produce relevant network traffic, and the session ends when you close the application or end the recording. The recording is used to generate a performance test that reproduces the behavior of the client application.
- **Test editing.** After recording, you can edit the events that were recorded. You can replace recorded test values with variable test data or add dynamic data to the test.
- **Test validation.** Before deploying the test, you can run the test manually as a single virtual user to make sure that the test runs smoothly and produces the expected results in a nominal environment with minimal server load. You might experience multiple test editing and validation cycles before your test runs as expected.
- **Workload emulation with schedules.** When the test runs repeatedly as anticipated, you specify an execution schedule and user groups to emulate a workload that a large number of virtual users generates.
- **Schedule execution.** You run the schedule, deploying test execution over virtual users that can be hosted on remote hosts. Each virtual user runs an instance of the test. Response time results are collected.
- **Evaluation of results.** You evaluate the results produced by the tests through the various reports that are generated during execution. You can also design custom reports.

TN3270 performance testing overview

With IBM® Rational® Performance Tester Extension for *Socket Protocols*, you can test the performance of TN3270 terminal server applications.

Informative performance test results rely on sound test development. Each of these stages contributes to the generation of meaningful test results:

- **Test creation.** You create a test by recording a session with a client application. Typically, the recorded session starts when you run the TN3270 terminal client. You then interact with the application in order to produce relevant network traffic. The session ends when you close the terminal client or end the recording. The recording is used to generate a performance test that reproduces the behavior of the client application.
- **Test editing.** After recording, you can edit the recorded events. You can replace recorded test values with variable test data or add dynamic data to the test.
- **Test validation.** Before deploying the test, you can run the test manually as a single virtual user to make sure that the test runs smoothly and produces the expected results in a nominal environment with minimal server load. You might complete multiple test editing and validation cycles before your test is robust.
- **Workload emulation with schedules.** When the test runs repeatedly as anticipated, you specify an execution schedule and user groups to emulate a workload that a large number of virtual users generates.
- **Schedule execution.** You run the schedule, deploying test execution over virtual users that can be hosted on remote hosts. Each virtual user runs an instance of the test. Response time results are collected.
- **Evaluation of results.** You evaluate the results that the tests produce through the various reports that are generated during execution. You can also design custom reports.

IBM® Engineering Test Management overview

IBM® Engineering Test Management is a collaborative, web-based, quality management solution that offers comprehensive test planning, manual testing, and integration with other test tools.

Quality Manager is based on the IBM® Rational® Jazz™ platform (<http://jazz.net> and <http://www.ibm.com/software/rational/jazz/>) and inherits many characteristics from that platform. Engineering Test Management is designed to be used by test teams of all sizes and supports a variety of user roles, such as test manager, test architect, test lead, tester, and lab manager, as well as roles outside the test organization.

Comprehensive test planning

A *test plan* that you define in Engineering Test Management drives activity for distributed teams through all phases of the project life cycle. The test plan defines the objectives and scope of the test effort and contains criteria to help teams determine the answer to the question "Are we ready to release?"

The test plan can be configured to meet the needs of your organization. You can use the test plan to do any and all of the following tasks:

- Define business and test objectives
- Establish a review and approval process for the test plan and for individual test cases
- Manage project requirements and test cases and establish the interdependencies between the two
- Estimate the size of the test effort
- Define the schedule for each test iteration and track the dates of other important test activities
- List the various environments to be tested and generate test configurations
- Create a read-only snapshot of the test plan at a particular point in time

- Define quality goals, entrance criteria, and exit criteria
- Create and manage test cases

Test script construction, execution, and reuse

Engineering Test Management provides a full-featured manual test editor. You can also import manual test scripts from IBM® Rational® Manual Tester. You can add reuse and automation capabilities to your manual tests by using keywords.

With Engineering Test Management, you can manage and execute test scripts that are created with tools such as IBM® Rational® Performance Tester, IBM® Rational® Service Tester for SOA Quality, and IBM® Security AppScan® Tester Edition.

You can also import test artifacts from external test management solutions, such as IBM® Rational® ClearQuest® Test Manager and IBM® Rational® Test Manager.

Test analysis and reporting

Engineering Test Management includes several standard test reports to help you evaluate test results. Reports are available during all phases of the test process.

You can use reports to perform these tasks:

- Determine the validity of a test run.
- Check feature coverage against test plans, test inputs, configurations, and so on. This can also be used to measure test progress and to analyze trends.
- Run a gap analysis to measure the resources needed to do your testing versus the resources that are available

Team collaboration

Engineering Test Management makes it easy to share information with other members of your team. With the Jazz-based work-item system, team members can assign tasks and defects to each other and to view everyone's status. Test plan authors and test case designers can distribute their work for review and track the status of each reviewer. New and changed requirements are visible to the team, as are the test cases that are needed to satisfy those requirements. Team members are notified automatically of any changes and milestones that impact their work.

Lab management

With Engineering Test Management lab management capabilities, you can create requests for the test environments that your test plan specifies. You can then work with the lab manager to ensure that lab resources and test environments are available when needed. Lab managers can track all lab resources from a centralized resource repository and fulfill requests from the test team.

Web application security

Engineering Test Management helps IT and security professionals protect against the threat of attacks and security breaches through its integration with IBM® Security AppScan® Tester Edition. Security testing for your web applications can result in higher-quality, more secure applications at a reasonable cost.

Governance

Engineering Test Management helps ensure that your business processes comply with industry, corporate, and departmental standards and regulations. Throughout the testing life cycle, Engineering Test Management provides you with the tools to obtain an up-to-the-minute measurement of software quality and project metrics. With its comprehensive test plan and integration with requirements management and defect tracking tools, Engineering Test Management helps streamline your test strategy and produce reliable records of test results and project history.

Streamlined Eclipse and full Eclipse overview

When you work in the streamlined Eclipse mode, only those functions that are directly related to the product are enabled in the workbench. When you install the product, by default, the check box to use the streamlined Eclipse mode is selected. With the full Eclipse mode, you have access to all Eclipse functions.

The streamlined Eclipse mode disables options from the menus that are not typically used during testing. Both the fully-enabled and streamlined Eclipse modes can operate using the same workspace, so if you start the product in the streamlined mode and discover that you cannot accomplish all of your tasks, you can close the workbench and restart it in the full Eclipse mode.

The choice of the mode in which to start the product depends on the user's activity and objectives. The streamlined mode is designed for straightforward testing and shows only those menu items that are related to testing. However, this restricts functions. The following list includes use cases where the full Eclipse mode might be preferred:

- You have multiple products installed and you want to use them in the same session.
- You are using the profiling and logging features. The profiling and logging view is not available in the streamlined mode.
- You are using advanced features of custom code including debugging custom code. For more information on custom code, see [Extending test execution with custom code on page 402](#).

Starting Rational® Service Tester for SOA Quality in full Eclipse mode

You can start the product in the full Eclipse mode to continue to use native Eclipse features along with Rational® Service Tester for SOA Quality.

Before you begin

You must have installed Rational® Service Tester for SOA Quality.

1. Click **Start > IBM Software Delivery Platform > IBM Rational Service Tester for SOA Quality - Full Eclipse**.
2. Perform the following steps to select a working directory, if you are starting the installation of Rational® Service Tester for SOA Quality for the first time.

- a. Enter the path of a `working directory` in the **Workspace** field or click **Browse** to select the directory.
- b. Select **Use this as the default and do not ask again** to make this your default workspace.
You can change your workspace from Rational® Service Tester for SOA Quality by clicking **File > Switch Workspace**.

3. Click **OK**.

Results

You have started Rational® Service Tester for SOA Quality in full Eclipse mode.

Starting Rational® Service Tester for SOA Quality in streamlined Eclipse mode

If you do not want to view native Eclipse UI, you can start in streamlined mode.

Before you begin

You must have installed Rational® Service Tester for SOA Quality.



Note: To start Rational® Service Tester for SOA Quality in the streamlined mode, the streamlined mode must be installed as an optional feature. It is automatically selected during the installation process.

1. Click **Start > IBM Software Delivery Platform > IBM Rational Service Tester for SOA Quality**.
2. Perform the following steps to select a working directory, if you are starting the installation of for the first time.
 - a. Enter the path of a `working directory` in the **Workspace** field or click **Browse** to select the directory.
 - b. Select **Use this as the default and do not ask again** to make this your default workspace.
You can change your workspace from by clicking **File > Switch Workspace**.

3. Click **OK**.

Results

You have started in streamlined Eclipse mode.

Chapter 4. Tutorials

This section contains the tutorials which explains the main features of .

You can find the following information:

- [Test an SOA application on page 38](#)

Test an SOA application

The movies in this tutorial show you the main features of IBM® Rational® Service Tester for SOA Quality or the SOA extension for IBM® Rational® Performance Tester. The tutorial requires Flash Player to view.

Learning objectives

Learn how to perform the following tasks:

- Record service calls and generate service tests
- Configure your test environment to support HTTP, Java™ Message Service (JMS), or WebSphere® MQ transport protocols
- Edit and manage variable data using datasets
- Run tests and view reports
- Create and deploy service stubs
- Generate HTML or PDF functional reports

45 minutes

Introduction: Testing services

This tutorial introduces you to testing services in an SOA environment.

The service testing capabilities of IBM Rational Performance Tester or IBM Rational Service Tester for SOA Quality automate the creation, execution and analysis of functional, regression and performance tests for SOAP-based web services, XML services or plain text service.

Learning objectives

The tutorial is divided into four modules, each with its own learning objectives. Learn to perform the following tasks:

- Creating a service test project
- Importing a WSDL file
- Creating a WSDL security stack
- Invoking a service call and generating a service test
- Simulating services with service stubs

This tutorial requires approximately 20 minutes to finish. If you explore other concepts related to this tutorial, it might take longer to complete.

Skill level

Beginner

Prerequisites

To complete this tutorial, you need to be familiar with the underlying concepts behind SOA and functional testing. Experience in using the perspectives and views in IBM® Rational® Software Development Platform are also required.

Module 1: Recording service calls and creating tests

In this module, learn how to create projects and record service calls for testing. The second part of this module helps you understand a service test.

Learning objectives

After completing the lessons in this module, you will know how to do the following tasks and understand the associated concepts:

- Create a test project
- Record a service test with the generic service client
- View a test in the test editor

This module requires approximately 12 minutes to complete.

Lesson 1.1: Creating a project and sending a request to a web service

When you create a service test, the first step is to set up your project and to import all the required resources.

About this task

For SOAP-based web services, the main resource is the Web Services Description Language (WSDL) specification. You can import a WSDL from a WebSphere® Service Registry and Repository or Universal Description, Discovery, and Integration (UDDI) repository. Of course, you can also simply import one from the workspace.

If you are testing services that do not use a WSDL file, you can skip this step.

See video

Lesson 1.2: Using the generic service client to create a service test

The generic service client is a multipurpose tool that is designed to send requests to any kind of service that uses an HTTP, JMS, or WebSphere® MQ transport and to view the message that the service returns.

About this task

The generic service client is useful for debugging or testing a service when you do not have access to a dedicated client to invoke the service call. You can set up a wide variety of transport and security configurations for the service, edit the parameters of the call and send attachments.

Although you can record a service test in a variety of ways, this lesson focuses on using the generic service client to send a series of service requests and to generate a service test with the results.

See video

Lesson 1.3: Creating security stacks

Algorithm stacks contain digital certificate information and the security algorithms that are applied to messages to perform secure communication with a web service.

About this task

In this lesson, learn how to use the WSDL security editor to create and edit algorithm stacks for service requests and responses. The WSDL security editor supports most industry standards for encrypting, decrypting, signing, and processing requests and responses from SOAP-based services. The WSDL security editor contains two pages, which relate to the two steps of setting up a security configuration:

- Describing an algorithm stack as a sequence of algorithm blocks that you can customize
- Associating an algorithm stack with each request and response operation in the WSDL

See video

Module 1 summary

In this module, you learned how to create projects and record service calls for testing.

Lessons learned

By completing this module, you learned about the following concepts and tasks:

- Creating a project for the first step in service testing
- Importing a WSDL file and creating transport and SSL configurations
- Invoking service calls with the generic service client and generating a test

Module 2: Editing service tests

In this module, you see how to edit your service test to include verification points and variable data by using datasets.

With the test editor, you can inspect or customize a test that you recorded. The test editor lists the web service call elements for a test, in sequential order. You can add, remove, and edit test elements in the test editor, including verification points and data correlation.

Learning objectives

After completing the lessons in this module, you will know how to do the following tasks and understand the associated concepts:

- Add a verification point
- Create a dataset
- Use a dataset in a test

This module requires approximately 8 minutes to complete.

Lesson 2.1: Enabling verification points

In this lesson, learn how to add verification points to check whether an expected behavior occurs during a run.

About this task

With verification points, you can test the behavior of the service during a test. For example, you can use verification points to ensure that a particular response contains the expected XML content or that a specific binary attachment is returned.

Each verification point returns a `Pass`, `Fail` or `Inconclusive` verdict in the test log. You can view a summary of verification point verdicts in the verification point report after running the test.

See video

Lesson 2.2: Using datasets

In this lesson, learn how to create a dataset that can provide tests with variable data and how to enable your test to use a dataset during a run.

About this task

Datasets provide tests with variable data during a run. When you record a test, you perform a sequence of steps that you expect a typical user to perform. From the recording, a test is generated that exactly reproduces these interactions. When you run this test, it uses the same data that you used during recording. To vary the data in the test, you use a data pool, which is typically a table that contains variable data. At run time, this variable data is substituted for the data in the recorded test.

See video

Module 2 summary

In this module, you learned how to use variable data to edit and manage testing situations.

Lessons learned

By completing this module, you learned about the following concepts and tasks:

- Adding verification points to check whether an expected behavior occurs during a test run
- Adding a reference to a dataset so that the test can use variables from a dataset during a run

Module 3: Running service tests

In this module, you see how to run a service test and to obtain the results of the test run.

You evaluate the results that the tests produce through the performance and verification point reports that are generated during execution. You can also customize reports by manipulating various counters or using custom report designs. Functional reports provide a comprehensive view of the behavior of the service under test. Reports can be exported and archived for validation.

Learning objectives

After completing the lessons in this module, you will understand how to do the following tasks and understand the associated concepts:

- Run a single service test
- View the test log and message contents
- Generate a functional report in HTML or PDF.

This module requires approximately 10 minutes to complete.

Lesson 3.1: Running a single test and viewing the test log

In this lesson, learn how to run a service test and view the results in the test log.

About this task

If you are using IBM® Rational® Performance Tester, then you can also create a schedule that contains service tests. A schedule that contains services tests works in the same way as other performance tests.

See video

Lesson 3.2: Generating a functional test report

You can generate functional test reports of your tests, which summarize the pass or fail verdicts of elements in the test log. The functional reports from the test run are generated as HTML or PDF files that use predefined report designs.

About this task

In this lesson, you learn how to produce a PDF or HTML report that covers the functional behavior of the service. The following report types are available:

- Extensible Stylesheet Language Transformation (XSLT) reports: These reports are faster to generate, but do not contain graphs.
- Business Intelligence and Reporting Tools (BIRT) report: These reports contain graphs but are slower to generate. You can customize and create your own BIRT report designs in the Report Design perspective of the workbench.

See video

Module 3 summary

In this module, you learned how run a test and display the results of the test run.

Lessons learned

By completing this module, you learned about the following concepts and tasks:

- Running a single service test
- Viewing the test log and message contents
- Generating a functional report in HTML or PDF

Module 4: Simulating services

In this module, you become familiar with simulating services with service stubs.

Learning objectives

After completing the lessons in this module, you will know how to do the following tasks and understand the associated concepts:

- Create service stubs
- Deploy and run a stub server

This module requires approximately 15 minutes to complete.

Lesson 4.1: Creating a service stub

In this lesson, learn how to create a service stub from a Web Service Description Language (WSDL) specification.

About this task

Service stubs are simulations of an actual service, which can be used to functionally replace the service in a test environment. A stub server replaces the actual application server in cases where it is not practical to use the server. For example, use a stub server in these instances:

- If you are testing a local service that uses data from a remote service, you might need to inject specific content into the service under test from the remote service. You can simulate the remote service with a service stub to ensure that the local service responds correctly to specific input.
- Some commercial service providers charge users for each request. If you are testing such a service, you can develop and debug your test against a stub service, which is based on the WSDL of the actual service, without being charged by the service provider.
- During integration of a large application that involves multiple clients and services, some services might not yet be operational, although their WSDL specifications are available. You can simulate the missing services with service stubs so that you can proceed with the integration work.

From the point of view of the client application, the service stub looks identical to the actual service that it simulates. To use a service stub as a replacement of the actual service, you must be able to replace the URL of the original service in the client application with the URL of the stub server.

You create a service stub by providing a current WSDL specification. The service stub is generated with the exact same ports and bindings as the original service so that it can be addressed with exactly the same interface. Each operation in the service returns a default response of the type defined by the WSDL.

See video

Lesson 4.2: Running the service stub server

In this lesson, learn how to deploy a service stub onto a stub server that is running on your local computer.

About this task

When you have finished editing the service stub, you can deploy the stub on a local stub server, which runs in the workbench. The stub server simulates an actual application server and can host multiple service stubs. You control the stub server from the stub monitor view.

Finally, to use the service stub instead of the original service, change the URL that the client application uses to point to the local stub server instead of the original application server. This URL and the WSDL of the service stub are provided in the stub monitor view.

See video

Module 4 summary

In this module, you learned how to simulate a service with a service stub and to validate that the stub server runs correctly.

Lessons learned

By completing this module, you learned about the following concepts and tasks:

- Creating service stubs
- Deploying and running a stub server

Summary

This tutorial introduced you the basics of service testing. You have learned how this product provides the following functions and features:

- The generic service client for sending and receiving service calls
- The test editor for creating and editing service tests
- Datasets for supplying realistic test data.
- The stub editor and stub monitor for simulating services and deploying service stubs.

Lessons learned

After completing all of the modules, you can perform these tasks:

- Record service calls and generate service tests
- Configure your test environment to support HTTP, JMS or WebSphere® MQ transport protocols
- Edit and manage variable data using datasets
- Run tests and view reports
- Create and deploy service stubs
- Generate HTML or PDF functional reports

Resources

To learn more about using Rational® Service Tester for SOA Quality or Rational® Service Tester, visit developerWorks®, IBM's resource for developers, at: <http://www.ibm.com/developerworks/>

Chapter 5. Administrator Guide

This guide describes how to install Rational® Service Tester for SOA Quality. After you install the product, you can perform administration tasks such as license configuration and integration with other products. This guide is intended for administrators.

Installation of Rational® Performance Tester

Installation of the product involves verifying requirements, planning, managing licenses, and configuring web-based help. This section lists all such topics.

This installation guide covers two independent products: Rational® Performance Tester and the Rational® Performance Tester Agent. The Rational® Performance Tester Agent is a tool that you use with Rational® Performance Tester. It is included as part of the Rational® Performance Tester product kit.

Rational® Performance Tester Agent consists of two capabilities:

- Generate load for the application under test by using the virtual users. You can increase the load generation capacity by installing additional agents on remote computers.
- Gather data for the Response Time Breakdown feature and in support of the startup and control of web services stubs in the SOA protocol.



Note: The Rational® Performance Tester and Rational® Performance Tester Agent are separate offerings and must be installed separately.

Installation requirements

Installation requires the correct hardware, software, server environment, operating systems, and user privileges for installing and running your software.

Hardware and Software requirements

Before you install the product, verify that your system meets the hardware and software requirements.

For information about hardware and software compatibility, see [System Requirements on page 16](#).

User privileges requirements

You must have a user ID that meets the following requirements to install Rational® Service Tester for SOA Quality and Rational® Performance Tester Agent.



Notes:



- Your user ID must not contain double-byte characters.
- For Windows operating system, you must have a user ID that belongs to the Administrators group.
- For Linux operating system, you must be able to log in as root.

Installation conventions and terminology

Understanding these terms and conventions can help you take full advantage of the installation information and your product.

The following conventions are used in this installation information:

- The default installation directory is written as `C:\installation_directory\product\inst.file`.
- The default log location for installation information is `C:\log_file_dir\log.txt`.

These terms are used in the installation topics.

Installation directory

The location of product artifacts after the package is installed.

Package

An installable unit of a software product. Software product packages are separately installable units that can operate independently from other packages of that software product.

Package group

A package group is a directory in which different product packages share resources with other packages in the same group. When you install a package using Installation Manager, you can create a new package group or install the packages into an existing package group. Eclipse-based packages installed in the same package group are able to use the shell-sharing features of Eclipse. Some packages cannot share a package group, in which case the option to use an existing package group is unavailable.

Repository

A storage area for installable software packages. A repository can be disc media, a folder on a local hard disk, or a server or web location.

Shared directory

In some instances, product packages can share resources. These resources are located in a directory that the packages share.

Capacity Planning

This document offers a methodology for assessing the CPU and memory characteristics of a given test and provides a set of best practices for scaling up to high volume loads using Rational® Performance Tester.

For information about capacity planning, see [Capacity planning for Rational® Performance Tester](#).

Installation Manager overview

Installation Manager is a program for installing, updating, and modifying packages. It helps you to manage the applications or packages that it installs on your computer. Installation Manager also helps you to keep track of what you have installed, determine what is available for you to install, and to organize installation directories.

Installation Manager provides features that help you keep packages up to date, modify packages, manage the licenses for your packages, and uninstall packages.

Installation Manager includes six wizards that make it easy to maintain packages:

- The **Install** wizard walks you through the installation process. You can install a package by simply accepting the defaults or you can modify the default settings to create a custom installation. Before you install, you get a complete summary of your selections throughout the wizard. Using the wizard you can install one or more packages at one time.
- The **Update** wizard searches for available updates to packages that you have installed. An update might be a released fix, a new feature, or a new version of the product. Details of the contents of the update are provided in the wizard. You can choose whether to apply an update. The **Update** wizard searches connected repositories for updates. If you are not connected to the Internet, you may not see newly available updates for your installed products. To apply an update to a computer that is not connected to the Internet, you must download the update and extract it to a local repository.
- The **Modify** wizard helps you modify certain elements of a package that you have already installed. During the first installation of the package, you select the features that you want to install. Later, if you require other features, you can use the modify packages wizard to add them to your package. You can also remove features and add or remove languages.
- The **Manage Licenses** wizard helps you set up the licenses for your packages. Use this wizard to change your trial license to a full license, to set up your servers for floating licenses, and to select which type of license to use for each package. Rational® Service Tester for SOA Quality requires runtime floating license keys to run tests with multiple virtual users and to use product extensions such as protocols. Runtime floating license keys are not managed using Installation Manager.
- The **Roll Back** wizard helps you to revert to a previous version of a package.
- The **Uninstall** wizard removes a package from your computer. You can uninstall more than one package at a time.

Installation locations

Installation Manager retrieves product packages from specified repositories and installs the products into selected locations, which are referred to as package groups.

Package groups

During installation, you specify a *package group* into which to install a product.

- A package group represents a directory in which products share resources.
- When you install a product by using Installation Manager, you either create a package group or install the product into an existing package group. A new package group is assigned a name automatically; however, you choose the installation directory for the package group.
- After you create a package group you cannot change the installation directory. The installation directory contains files and resources that are shared by the products that are installed into that package group.
- Product resources that are designed to be shared with other packages are installed in the shared resources directory. Not all products can share a package group, in which case the option to use an existing package group is disabled.
- When you install multiple products at the same time, all products are installed into the same package group.

Shared resources directory

The *shared resources directory* is where product resources are installed so that they can be used by multiple product package groups. You define the shared resources directory the first time that you install the first product package. For best results, use your largest disk drive for shared resources directories. You cannot change the directory location unless you uninstall all product packages.

Offering coexistence considerations

Some products are designed to coexist and share functions when they are installed in the same package group. A package group is a location where you can install one or more software product packages.

When you install each product package, you select whether to install the product package into an existing package group or to create a package group. Installation Manager prevents you from installing products into package groups products that are not designed to share or do not meet version compatibility and other requirements. To install more than one product at a time, the products must be able to share a package group.

Any number of eligible products can be installed to a package group. When a product is installed, the product functions are shared with all the other products in the package group. If you install a development product and a testing product into one package group, when you start either of the products, you have both the development and testing functions available to you in your user interface. If you add a product with modeling tools, all the products in the package group have the development, testing, and modeling functions available.

Eclipse instance overview

The product package that you install using Installation Manager comes with a version of Eclipse, which is the base platform of this product package. If you already have Eclipse installed on your workstation, you can add your product package directly to that Eclipse installation and extend the functions of the Eclipse integrated development environment (IDE).


Extending an Eclipse IDE adds the functions of the newly installed product, but maintains your IDE preferences and settings. Previously installed plug-ins are also still available.

In most cases, your current Eclipse IDE must be the same version as the Eclipse that the product you are installing uses. Installation Manager checks that the Eclipse instance that you specify meets the requirements for the installation package and helps you install the latest updates from eclipse.org, if required.

Increasing the number of file handles on Linux™ workstations

For best product performance, increase the number of file handles above the default setting of 1024 handles.


About this task

 **Important:** Before you work with your product, increase the number of file handles. Most products use more than the default limit of 1024 file handles per process. A system administrator might need to make this change.


Exercise caution when using the following steps to increase your file descriptors on Linux™. If the instructions are not followed correctly, the computer might not start correctly.

To increase your file descriptors:


1. Log in as root. If you do not have root access, you must obtain it before continuing.
2. Change to the `etc` directory.

 **Attention:** If you decide to increase the number of file handles in the next step, *do not* leave an empty `initscript` file on your computer. If you do so, your computer will not start up the next time that you turn it on or restart.

3. Use the `vi` editor to edit the `initscript` file in the `etc` directory. If this file does not exist, type `vi initscript` to create it.
4. On the first line, type `ulimit -n 30000`. The point is that 30000 is significantly larger than 1024, the default value on most Linux™ computers.

 **Important:** Do not set the number of handles too high, because doing so can negatively impact system-wide performance.

5. On the second line, type `eval exec "$@"`.
6. Save and close the file after making sure that you have completed steps 4 and 5.

 **Note:** Ensure that you follow the steps correctly. If this procedure is not completed correctly, your computer will not start.

7. **Optional:** Restrict the number of handles available to users or groups by modifying the `limits.conf` file in the `etc/security` directory. Both SUSE Linux™ Enterprise Server (SLES) Version 9 and Red Hat Enterprise Linux™ Version 4.0 have this file by default. If you do not have this file, consider using a smaller number in step

4 in the previous procedure (for example, 2048). Do this so that most users have a reasonably low limit on the number of open files that are allowed per process. If you use a relatively low number in step 4, it is less important to do this. However, if you set a high number in step 4 earlier and you do not establish limits in the `limits.conf` file, computer performance can be significantly reduced.

The following sample `limits.conf` file restricts all users, and then sets different limits for others afterwards. This sample assumes that you set handles to 8192 in step 4 earlier.

```
*      soft nofile 1024
*      hard nofile 2048
root   soft nofile 4096
root   hard nofile 8192
user1  soft nofile 2048
user1  hard nofile 2048
```

Note that the `*` in the preceding example sets the limits for all users first. These limits are lower than the limits that follow. The root user has a higher number of allowable handles open, while the number that is available to user1 is between the two. Make sure that you read and understand the documentation that the `limits.conf` file contains before making changes.

What to do next

For more information on the `ulimit` command, see the main page for `ulimit` in the Linux™ documentation.




Starting the launchpad

You can start the launchpad program to install the product.

Before you begin

You must have downloaded the electronic disks from the IBM® Passport Advantage® portal.

1. Open the command-line interface.
2. Change the directory to a location where you extracted the disk images.
3. Run the following command to start the launchpad program:

- On : `RST_SETUP\launchpad.exe`
- On  and : `RST_SETUP/launchpad.sh`

Results

You have started the launchpad program.

Installation of the product by using IBM® Installation Manager

The Setup disk includes the launchpad program, which provides you with a single location to start the installation process. You must download the product bits and then from the Installation Manager point to the Setup disk.

Use the launchpad program to start the installation of software by using any of the following methods:

- Installing from an electronic image on your local file system
- Installing from an electronic image on a shared drive



Note: For Linux™ computers, you must log in as the root user before you begin the installation process.

For products that are installed by Installation Manager, starting the installation process from the launchpad program causes Installation Manager to be automatically installed if it is not already on your computer. Furthermore, the installation process is already configured with the location of the repository that contains the installation package. If you install Installation Manager separately, you must configure the repository preferences manually. Also, you can start the installation of a number of supporting software items from the launchpad.

To install Rational® Service Tester for SOA Quality as a non-administrator, you cannot use the launchpad program to start the installation process. Instead, you must manually run the `userinst` program from the Setup disk. Running the `userinst` program provides the same functions as starting the installation of Rational® Service Tester for SOA Quality from the launchpad.

You can also install Rational® Service Tester for SOA Quality from the command-line interface by using response files rather than using the graphical user interface. For more information about silent mode installation, refer to the [IBM® Installation Manager](#) documentation.

Installing IBM® Rational® Service Tester for SOA Quality

You must install Rational® Service Tester for SOA Quality to automate the creation, execution and analysis of functional, regression and performance tests for SOAP-based web services.

About this task

If you use the Launchpad program to install the product, IBM Installation Manager is installed automatically if you do not have it on your computer. After the installation is complete, Installation Manager starts the product installation by using the preconfigured repository that contains the product package.

If you install Installation Manager and then install the product, you must set the repository preferences manually.

For more information about installing the product from the command prompt in the silent mode, see [IBM Installation Manager Knowledge Centre](#).

1. Perform one of the following steps to install the product:
 - a. Perform the following steps to install the product from compressed files, such as .zip or ISO files:
 - i. Extract the files into a common directory, and then navigate to *Common_Directory/RST/disk1/Platform_Directory*.
 - ii. Double-click the **install.exe** file
 - iii. Select a language in which to run the launchpad and Installation Manager.

- iv. Select the product to install from the launchpad menu.

The **Install Packages** window opens.

- b. Perform the following steps to install the product from Installation Manager:

- i. Open Installation Manager.

- ii. Click **File > Preferences > Repositories** to add a repository link to the product's setup disk, and then click **OK**.

- iii. Click **Install**.

2. Click a product package to highlight it.

Result

The description of the package is displayed in the **Details** pane at the end of the window.

3. **Optional:** Click **Check for Other Versions, Fixes, and Extensions** to search for any updates to the product packages.

If updates for a product package are found, they are displayed in the **Installation Packages** list under their corresponding product. Installation Manager displays only the recommended updates by default.

4. **Optional:** Select the **Show all versions** checkbox to view all the updates that are available for the packages.

You can click the package name to view the package description under the **Details** pane. If additional information about the package is available, such as a `readme` file or release notes, a **More info** link is included at the end of the description text. You can click the link to view additional information in a browser.



Note: For Installation Manager to search the predefined IBM® repository locations for the installed packages, you must select the **Search service repositories during installation and updates** checkbox on the **Repositories** preference page. This checkbox stays selected as the default value. A progress indicator shows the status of the ongoing process. You can install updates simultaneously when you install the base product package.

5. Select the product package and its updates to install it on your computer, and then click **Next**.



Note: Updates with dependencies are automatically selected and cleared together. If you install multiple packages simultaneously, all the packages are installed into the same package group.

6. Read and understand the terms of all of the license agreements for the selected package, and then perform the following steps:

- a. If you agree to the terms of all of the license agreements, click **I accept the terms of the license agreements**.

- b. Click **Next** to continue.

7. Select a location for the shared resources directory, and then click **Next**.

The shared resources directory contains resources that can be shared with one or more package groups. You can either browse the location in the **Shared Resources Directory** field or enter the path of the location for the shared resources directory.

The default path of the **Shared Resources Directory** is as follows:

- For **Windows**: C:\Program Files\IBM\IBMIMShared
- For **Linux**: /opt/IBM/IBMIMShared
- For **Mac OS X**: /Applications/IBM/IBMIMShared



Remember: You can specify the shared resources directory only for the first time when you install a package. You must use your largest disk for this directory. The largest disk ensures that you have adequate space for the shared resources of future packages. You cannot change the location of the directory unless you uninstall all packages.

8. Select one of the following options to use an existing or new package group:

Choose from:

- **Use the existing package group:** You can use this option if you already created the package group.
- **Create a new package group:** You can use this option either to install or update the product.

A package group represents a directory in which packages share resources with other packages in the same group.



Note: The **Use the existing package group** option is disabled when you install a package for the first time.

9. Perform the following steps to create a new package group:

- a. Select **Create a new package group**.
- b. Enter the path in the **Installation Directory** field for the package group.
The name for the package group is created automatically.

The default paths are as follows:

- For **Windows**: C:\Program Files\IBM\SDP
- For **Linux**: /opt/IBM/SDP
- For **Mac OS X**: /Applications/IBM/SDP

- c. Select **32-bit** or **64-bit** as **Architecture Selection**.
- d. Click **Next**.

10. Click **Next**.

Rational® Service Tester for SOA Quality does not support extending an existing Eclipse IDE.

11. Select the languages for the package group, and then click **Next**.

The corresponding translations for the user interface and documentation for the product package are installed.

12. Select all the features that you want to install, and then click **Next**.

You can perform the following actions to install or view information about the features:

- Select or clear checkboxes to enable or disable the features.
 - Select the **Show dependencies** checkbox to view the dependency relationships between features.
 - Click a feature to view its brief description under the **Details** section.
13. Review your choices, and then click **Install**.

You can click **Back** to change the choices that you made on previous pages and make your changes.

Result

A progress indicator shows the percentage of the installation that is complete.

14. **Optional:** Click **View Log File** to open the installation log file for the current session in a new window.



Note: You must close the **Installation Log** window to continue.

15. License the product.

See the [License management on page 63](#) topics.

16. Click **Finish** to exit the installation wizard.

Results

You have installed Rational® Service Tester for SOA Quality.

What to do next

You can configure licenses for Rational® Service Tester for SOA Quality. See [License enablement on page 66](#).

Installing Rational® Performance Tester Agent

You must install Rational® Performance Tester Agent on different computers to apply load on the server that hosts the application under test.

About this task

If you use the Launchpad program to install the product, IBM Installation Manager is installed automatically if you do not have it on your computer. After the installation is complete, Installation Manager starts the product installation by using the preconfigured repository that contains the product package.

If you install Installation Manager and then install the product, you must set the repository preferences manually.

1. Perform one of the following steps to install Rational® Performance Tester Agent:
 - a. Perform the following steps to install Rational® Performance Tester Agent from compressed files, such as .zip or ISO files:
 - i. Extract the files into a common directory, and then navigate to *Common_Directory/RPTAGENT_SETUP/disk1/Platform_Directory*.
 - ii. Double-click the **install.exe** file
 - iii. Select a language in which to run the launchpad and Installation Manager.
 - iv. Select the product to install from the launchpad menu.

The **Install Packages** window opens.

- b. Perform the following steps to install Rational® Performance Tester Agent from Installation Manager:
 - i. Open Installation Manager.
 - ii. Click **File > Preferences > Repositories** to add a repository link to the product's setup disk, and then click **OK**.
 - iii. Click **Install**.
2. Click a product package to highlight it.

Result

The description of the package is displayed in the **Details** pane at the end of the window.

3. **Optional:** Click **Check for Other Versions, Fixes, and Extensions** to search for any updates to the product packages.

If updates for a product package are found, they are displayed in the **Installation Packages** list under their corresponding product. Installation Manager displays only the recommended updates by default.
4. **Optional:** Select the **Show all versions** checkbox to view all the updates that are available for the packages. You can click the package name to view the package description under the **Details** pane. If additional information about the package is available, such as a `readme` file or release notes, a **More info** link is included at the end of the description text. You can click the link to view additional information in a browser.



Note: For Installation Manager to search the predefined IBM® repository locations for the installed packages, you must select the **Search service repositories during installation and updates** checkbox on the **Repositories** preference page. This checkbox stays selected as the default value. A progress indicator shows the status of the ongoing process. You can install updates simultaneously when you install the base product package.

5. Select the product package and its updates to install it on your computer, and then click **Next**.



Note: Updates with dependencies are automatically selected and cleared together. If you install multiple packages simultaneously, all the packages are installed into the same package group.

6. Read and understand the terms of all of the license agreements for the selected package, and then perform the following steps:
 - a. Click **I accept the terms in the license agreement**.
 - b. Click **Next** to continue.
7. Select a location for the shared resources directory, and then click **Next**.

The shared resources directory contains resources that can be shared with one or more package groups. You can either browse the location in the **Shared Resources Directory** field or enter the path of the location for the shared resources directory.

The default path of the **Shared Resources Directory** are as follows:

- For **Windows**: C:\Program Files\IBM\SDP
- For **Linux**: /opt/IBM/SDP
- For **Mac OS X**: /Applications/IBM/SDP



Remember: You can specify the shared resources directory only for the first time when you install a package. You must use your largest disk for this directory. The largest disk ensures that you have adequate space for the shared resources of future packages. You cannot change the location of the directory unless you uninstall all packages.

8. Select one of the following options to use an existing or new package group:

Choose from:

- **Use the existing package group:** You can use this package group if you already created the package group.
- **Create a new package group:** You can use this package group either to install or update the product.

A package group represents a directory in which packages share resources with other packages in the same group.



Note: The **Use the existing package group** option is disabled when you install a package for the first time.

9. Perform the following steps to create a new package group:
 - a. Select **Create a new package group**.
 - b. Enter the path in the **Installation Directory** field for the package group.
The name for the package group is created automatically.

The default paths are as follows:

- For **Windows**: C:\Program Files\IBM\SDP
- For **Linux**: /opt/IBM/SDP
- For **Mac OS X**: /Applications/IBM/SDP

c. Select **32-bit** or **64-bit** as **Architecture Selection**.

d. Click **Next**.

10. Select all the features that you want to install, and then click **Next**.

You can perform the following actions to install or view information about the features:

- Select or clear checkboxes to enable or disable the features.
- Select **Show dependencies** to view the dependency relationships between features.
- Click a feature to view its brief description under the **Details** section.

You can select the **Windows desktop Application testing (Next Generation)** and **UI Test Agent** checkboxes to install the UI Test Agent which provides the necessary prerequisites to test native and hybrid mobile applications, and Windows desktop applications.

The UI Test Agent is installed along with Rational® Performance Tester Agent on your computer.



Note: After the agent installation, the UI Test Agent starts to run automatically on your computer whenever you restart your computer. On Linux and macOS, you must set the environment variable to start the UI Test Agent automatically.



Notes:

- The Load Generation Agent is used to generate a load on the system under test and gather data for the Response Time Breakdown feature.
- Installation Manager automatically enforces any dependencies with other features and displays updated download sizes and disk space requirements for the installation.

11. Perform the following steps to configure the agent:

- a. Select **Rational Functional Tester Panel** if you shell-share Rational® Performance Tester with Rational® Functional Tester and run a UI test on the remote agent computers.



Notes:



- On Windows system, clear the **The agent will be used primarily to support remote execution of UI tests from RFT** checkbox if you are not running a UI test, so that `Majordomo` runs as a service.
- On Linux and Mac operating systems, the **The agent will be used primarily to support remote execution of UI tests from RFT** option is not available. The agent runs as a service and can run a UI test. This is the default behavior.
- When `Majordomo` runs as a service, it starts the service automatically after the computer is restarted. However, if you want to run the UI test on remote agent computers, then you must select **The agent will be used primarily to support remote execution of UI tests from RFT** checkbox so that after the installation of the agent is complete, `Majordomo` runs as a batch file instead of a service.
- When `Majordomo` runs as a batch file, it stops after the computer is restarted. You must restart `Majordomo` by double-clicking the `AgentInstallDir/Majordomo/Majordomo.bat` file.


b. Select **IBM Rational Load Generation Agent Configuration** and perform the following steps:

i. Specify the values for the following parameters for Rational® Performance Tester:

Field name	Description	Example
Workbench hostname	The hostname of .	localhost
Workbench port	The port number of .	7080

ii. Specify the values of the following parameters for Rational® Test Automation Server:

Field name	Description	Example
Server Host-name	<p>The hostname of Rational® Test Automation Server.</p> <p> Note:</p> <ul style="list-style-type: none"> ▪ The hostname of Rational® Test Automation Server must be resolvable through a Domain Name Server (DNS). ▪ An IP address cannot be the hostname of Rational® Test Automation Server. 	localhost

Field name	Description	Example
	 <ul style="list-style-type: none"> The hostname of Rational® Test Automation Server through host files must not be specified. <p>For example, <code>/etc/hosts</code> or <code>C:/Windows/system32/drivers/etc/host</code></p>	
Server Port	The port number of Rational® Test Automation Server.	443
Server Token	An offline user token that is created from Rational® Test Automation Server.	eyJhbGciOiJIUzI1NiIsInR5cCI6IkpzLnR
Server URL Alias	The name of the Server URL Alias that you provided during the creation of the team space in Rational® Test Automation Server.	testteam

c. Select **IBM Rational Performance Tester Agent Configuration Panel** and select the product from the available list that is being installed with **IBM Rational Performance Tester Agent**.

d. Click **Next**.

12. Review your choices, and then click **Install**.

You can click **Back** to change the choices that you made on previous pages and make your changes.

Result

A progress indicator shows the percentage of the installation that is complete.

13. **Optional:** Click **View Log File** to open the installation log file for the current session in a new window.



Note: You must close the **Installation Log** window to continue.

14. Click **Finish** to exit the installation wizard.

Results

You have installed Rational® Performance Tester Agent.

What to do next

You can check the status of agents from Rational® Performance Tester. See Checking the status of agents.

Installing Rational® Performance Tester Agent on an AIX 7.1 or later

IBM® Rational® Performance Tester Agent includes the Load Generation agent to generate load for the application under test by using the virtual users, to gather data for the Response Time Breakdown feature, and in support of the startup and control of web services stubs in the SOA protocol. When you install the Agent, the Data Collection

Infrastructure application is also installed. This application is used to instrument the servers for Response Time Breakdown.

About this task

By starting the installation process from the launchpad program, Installation Manager is automatically installed if it is not already on your computer. Furthermore, Installation Manager starts preconfigured with the location of the repository that contains the product package.

If you install and start Installation Manager directly, then you must set repository preferences manually. To learn how to install the product from a command prompt in silent mode, see the [Installing Silently](#) section in the [Installation Manager Knowledge Center](#).

1. Log in as the root user before you begin installing the product.
2. If you are installing from compressed files, such as .zip or ISO files, extract the files into a common directory. Extract the disk images to directories named /disk1, /disk2, and so on. Extract the Setup disk image to a directory. The Setup disk contains the launchpad program.
3. Run the `launchpad.sh` file, which is located in the root directory of the Setup disk installation image.
4. Select a language in which to run the launchpad and Installation Manager, and click **OK**.
5. Select the product to install from the launchpad menu. The Install Packages window opens.
6. Click a product package to highlight it. The description of the package is displayed in the Details pane at the end of the screen.
7. **Optional:** To search for updates to the product packages, click **Check for Other Versions and Extensions**. If updates for a product package are found, then they are displayed in the Installation Packages list on the Install Packages page under their corresponding products. Only recommended updates are displayed by default.
8. **Optional:** To view all updates that are found for the available packages, click **Show all versions**.
9. **Optional:** To display a package description, under Details, click the package name. If additional information about the package is available, such as a readme file or release notes, a **More information** link is included at the end of the description text. Click the link to display the additional information in a browser. To fully understand the package you are installing, review all information beforehand.
10. Click **Next** to continue.



Note: If you install multiple packages at the same time, then all the packages are installed into the same package group.

11. On the Licenses page, read the license agreement for the selected package. If you select more than one package to install, there might be a license agreement for each package. On the left side of the License page, click each package version to display its license agreement. The package versions that you select to install (for example, the base package and an update) are listed under the package name.
12. If you agree to the terms of all the license agreements, click **I accept the terms of the license agreements**; then click **Next** to continue.

13. On the Location page, type the path for the shared resources directory in Shared Resources Directory, or accept the default path. The shared resources directory contains resources that can be shared by one or more package groups. The default path is `/opt/IBM/IBMIMShared`. Click **Next** to continue.



Note: You can specify the shared resources directory only the first time that you install a package. Use your largest disk for this operation to help ensure adequate space for the shared resources of future packages. You cannot change the directory location unless you uninstall all packages.

14. On the Location page, create a package group to install the product package into, or, if this is an update, use the current package group. A package group represents a directory in which packages share resources with other packages in the same group. The default path is .
15. To create a package group, click **Create a new package group**, and then type the path for the installation directory for the package group. The name for the package group is created automatically. Click **Next** to continue.
16. On the Features page under Translations, select the languages for the package group. The corresponding translations for the user interface and documentation for the product package will be installed.
17. On the next Features page, select the package features to install and click **Next**.
18. On the next Features page, for Load Generation Agent, type the workbench host name. If you do not want to use 7080 as the port number, change the port number. You can also change the port number after you install Rational® Performance Tester Agent.
19. On the Summary page, review your choices and click **Install**.

Uninstalling the product by using IBM® Installation Manager

When you no longer require Rational® Service Tester for SOA Quality, you can use IBM® Installation Manager to uninstall Rational® Service Tester for SOA Quality that you have installed.

Before you begin

You must have completed the following tasks:

- Installed Installation Manager.
- Closed any open windows of Rational® Service Tester for SOA Quality.
- Closed any open web browsers.
- Closed all the other applications that are enabled by Rational® Service Tester for SOA Quality.

1. Open Installation Manager.
2. Click **Uninstall**.
3. Select the Rational® Service Tester for SOA Quality package checkbox on the **Uninstall Packages** window, and then click **Next**.
4. Review the list of packages that are ready to uninstall, and then click **Uninstall**.

Result

The **Complete** page is displayed after the uninstallation process is complete.

5. Click **Finish** to exit the Installation Manager wizard.

Results

You have uninstalled Rational® Service Tester for SOA Quality from your computer.

License management

Licensing for your installed IBM® software and customized packages is administered through the Manage Licenses wizard in the IBM® Installation Manager. The Manage Licenses wizard displays license information for each installed package.

Using the Manage Licenses wizard, you can apply a license to a product or upgrade trial versions of an offering to a licensed version by importing a product activation kit. You can also enable floating license enforcement for offerings with trial or permanent licenses to use floating license keys from a license server.

For more information about managing licenses for your Rational® product, see these resources:

- [Product Activation of Eclipse-based Rational products](#)
- [Rational licensing support](#)
- [Rational License Key Server Knowledge Center](#)

License descriptions

As a purchaser of an IBM® Rational® software product, you can choose from four types of product licenses:

- [Floating licenses on page 63](#)
- [Token licenses on page 64](#)
- [Processor value unit \(PVU\) licensing on page 64](#)
- [Runtime licenses on page 64](#)

The best choice for your organization depends upon how many people use the product, how often they require access, and how you prefer to purchase software.

Floating licenses

An IBM® Rational® Floating license is a license for a single software product that can be shared among multiple team members; however, the total number of concurrent users cannot exceed the number of floating licenses you purchase. For example, if you purchase one floating license for a Rational® software product, any user in your organization can use the product at any given time. Another person who wants to access the product must wait until the current user logs off.

To use floating licenses, you must obtain floating license keys and install them on a Rational® License Server. The server responds to user requests for access to the license keys; the server grants access to the number of concurrent users that equals the number of licenses the organization purchased.

Floating license enforcement provides these benefits:

- License compliance enforcement throughout the organization
- Fewer license purchases
- License keys served for IBM® Rational® Team Unifying and IBM® Rational® Software Delivery Platform desktop products from the same license server
- Some versions of Rational® products require an upgraded version of the Rational® License Server.
See this support article for license upgrade information: <http://www.ibm.com/support/docview.wss?uid=swg21250404>

Token licenses

The token-based license model means that you can buy a certain number of token licenses. If you use a Rational® tool that checks out a feature that is token-based, the feature line in the license file specifies the number of tokens that are checked out. Token-based licenses can only be used with floating licenses. They cannot be used for authorized user license. For more details about token licensing, contact your local IBM® marketing representative.

Processor value unit (PVU) licensing

Processor value unit (PVU) licensing is for providing you pricing structures that are responsive to both the type and number of processors that are available to installed products.

Entitlements can be full capacity or subcapacity. Under the processor value unit licensing structure, you license software based on the number of value units assigned to each processor core. For example, processor type A is assigned 80 value units and processor type B is assigned 100 value units. If you license a product to run on two type A processors, you must acquire an entitlement for 160 value units. If the product is to run on two type B processors, the required entitlement is 200 value units.

The processor value units table, which assigns a number of value units to each checkmark_perf.jpg processor type, is regularly updated to provide for the introduction of new processor technologies. Agents retrieve information about the number and type of processor on the monitored computer or partition and the table is used to determine the level of license use in terms of processor value units.

For more information about managing licenses for your Rational® product, see the [IBM License Metric Tool](#) Knowledge Center to learn about processor value unit licensing.

Runtime licenses

In addition to product activation, Rational® Performance Tester also checks for runtime licensing when you run a performance schedule. Runtime licensing consists of the correct protocol keys and virtual tester license key packs. You must have a protocol key to run performance tests with any protocol other than HTTP. You must have a virtual

tester license key pack to run tests with more than five virtual users. You can check out these keys from the same Rational® License Server that you use for floating license product activation.

For more information about Runtime licenses, see [IBM Common Licensing](#)

Runtime license examples

To run tests, you must have the correct license keys installed.

The trial licensing key permits you to run Rational® Performance Tester for 30 days from the initial installation. After 30 days from the first time you install the product, the trial license expires.

The following table shows the number of virtual users you can use, depending on the license keys that you have installed:

Protocol	Trial license	Permanent license	Permanent with Protocol key	500-user virtual tester key pack and protocol key
HTTP	5	5	n/a	505
Siebel	5	1	5	505
SAP(GUI)	5	1	5	505
SAP(Web)	5	1	5	505
Citrix	5	1	5	505
TN3270	5	5	n/a	505
Socket	5	5	n/a	505
SOA	5	1	5	505

Purchasing licenses

You can purchase new licenses if your current product license is about to expire or to acquire additional product licenses for team members.

1. Determine the type of license to purchase.
2. Go to [ibm.com®](#) or contact your IBM® sales representative to purchase the product license. For details, visit the IBM® web page on [Find a product](#).
3. Depending on the type of license you purchase, use the Proof of Entitlement that you receive and complete one of these steps to enable your product:

Choose from:

- If you purchase Authorized User licenses for your product, go to [Passport Advantage®](#), and follow the instructions there for downloading your product activation kit. After you have downloaded the activation kit, import the product activation .jar file by using Installation Manager.

Back up the product activation .jar file. If you uninstall the product and then install the product again, you might need to use the product activation .jar file to license the product again.

- If you purchase floating licenses for your product, go to the [IBM® Rational® Licensing and Download Center](#), and then click the link to connect to the IBM® Rational® License Key Center. There you can use your Proof of Entitlement to obtain floating license keys for your license server.

Optionally, you can go to IBM® Passport Advantage® to download the activation kit for your product. After importing the activation kit, you can switch from a floating to a permanent license type if you use your computer offline for long periods.

What to do next

To import the activation kit or enable floating license support for your product, use the Manage Licenses wizard in IBM® Installation Manager.

License enablement

If you are installing the software for the first time or want to extend a license to continue using the product, you have options on how to enable licensing for your product.

Licenses for this product are enabled in two ways:

- Importing a product activation kit
- Enabling Rational® Common Licensing to obtain access to floating license keys

Activation kits

The Product Activation Kit contains the permanent license key for you product. You use IBM® Installation Manager to import the activation kit to your product.

To enable the PVU licensing capability, download the PVU activation kit from the Rational® Common Licensing server, unzip it, and use Installation Manager to import the activation kit.

Floating license enforcement

Optionally, you can obtain floating license keys, install IBM® Rational® License Server, and enable floating license enforcement for your product. Floating license enforcement provides these benefits:

- License compliance enforcement throughout the organization
- Fewer license purchases
- License keys served for IBM® Rational® Team Unifying and Software Delivery Platform desktop products from the same license server



Note: Some 7.0 and later versions of Rational® products require an upgraded version of the Rational® License Server. See this support article for license upgrade information: <http://www.ibm.com/support/docview.wss?uid=swg21250404>

For more information about obtaining activation kits and floating licenses, see [Purchasing licenses on page 65](#).

Viewing license information for installed packages

You can review license information for your installed packages, including license types and expiration dates, from IBM® Installation Manager.

1. Start IBM® Installation Manager.
2. On the main page, click **Manage Licenses**.

Results

The package vendor, current license types, and expiration dates are displayed for each installed package.

Product upgrade and migration

When you want to use the enhanced functionalities of Rational® Service Tester for SOA Quality, you must upgrade to the latest version of the product software.

You can upgrade Rational® Service Tester for SOA Quality to the latest version by uninstalling the existing version of Rational® Service Tester for SOA Quality. You can then install the latest version of Rational® Performance Tester.

Migrating test assets to new version of the product

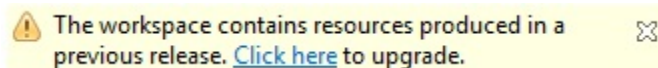
After you install a later version of the product and you choose to open the product from an old workspace, you are prompted to migrate test projects, tests, schedules, rules, and reports. Tests and schedules are migrated automatically when you modify and save them.


You cannot have two versions of the products installed on your computer at one time. Before you install a new version, uninstall the previous version of the product. If you update the product with IBM® Installation Manager, you do not have to uninstall the previous version. Uninstalling a previous version does not delete your test assets.



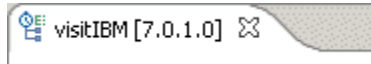
Note: When you want to uninstall the previous version of the product and then install the latest version, you must install the latest version in the same folder that contained the previous version. Thereby, you can avoid compilation errors in the project in the latest version.


When you open a project that contains an older test asset, a message is displayed in the **Test Navigator** view. Typically, you upgrade your tests, schedules, and rules.



 **Important:** Ensure that you back up the test assets before upgrading them for the new version of the product. Do not open a migrated test project with a previous version of the product.

If you leave tests, schedules, and rules unchanged, they will not have the new functions that current release adds. You can always save a modified test asset under a new name, which preserves the older asset. You can identify an older asset by its version, which is listed in brackets:



 **Note:** A new release might include enhancements to the default reports. When you run a test or schedule or open a report, you are prompted to upgrade reports to the latest version. If you upgrade the default reports to the latest version, you lose customizations that you have made to the reports.

If you encounter errors when you open a workspace from a different version of the product, reset the perspective. To reset the perspective, click **Window > Reset Perspective**.

Integration with other products


You can integrate IBM® Rational® Performance Tester with certain products to run tests, manage test assets, and create defects.

The following topics provides more information about the integration of Rational® Performance Tester with other products:

Integration plugin compatibility matrix

You can find information about the versions of the integration plugin that are compatible with Rational® Performance Tester.

The following table lists the versions of the integration plugin that are required to integrate Rational® Performance Tester with Jenkins, Ant, and UrbanCode™ Deploy.

 **Note:** You must download the required version of the integration plugin from the portal based on the existing version of Rational® Performance Tester. You can then integrate Rational® Performance Tester with Jenkins, Ant, and UrbanCode™ Deploy.

Rational® Performance Tester	Ant plugin	Jenkins plugin	UrbanCode™ Deploy plugin
10.1.0	RPT-Ant-5.0	RPT-Jenkins-6.0	RPT-UCD-5.0
10.1.1	RPT-Ant-6.0	RPT-Jenkins-7.0	RPT-UCD-6.0

Rational® Performance Tester	Ant plugin	Jenkins plugin	UrbanCode™ Deploy plugin
10.1.2	RPT-Ant-7.0	RPT-Jenkins-8.0	RPT-UCD-6.0
10.1.3	RPT-Ant-7.0	RPT-Jenkins-8.0	RPT-UCD-7.0
10.2.0	RPT-Ant-8.0	RPT-Jenkins-8.0	RPT-UCD-7.0
10.2.1	RPT-Ant-8.0	RPT-Jenkins-8.0	RPT-UCD-7.0
10.2.2	RPT-Ant-9.0	RPT-Jenkins-9.0	RPT-UCD-7.0
10.2.3	RPT-Ant-9.1	RPT-Jenkins-9.1	RPT-UCD-9.2
10.5.0	RPT-Ant-9.1	RPT-Jenkins-9.2	RPT-UCD-9.3
10.5.1	RPT-Ant-9.1	RPT-Jenkins-9.2	RPT-UCD-9.3
10.5.2	RPT-Ant-9.1	RPT-Jenkins-9.4	RPT-UCD-9.3

Running tests by using Ant

When you want to automate repetitive tasks, you can run performance test assets from the command-line interface by using `Ant`.

Before you begin

You must have completed the following tasks:

- Installed .

If you installed Rational® Performance Tester on the Mac operating system, you must have added `TEST_WORKBENCH_HOME` as an environment variable. You must have pointed that environment variable to the installation directory of Rational® Performance Tester.



Note: On Windows® and Linux™ operating systems, the `TEST_WORKBENCH_HOME` environment variable is set after the installation of Rational® Performance Tester is complete.

- Created test assets within Rational® Performance Tester. See [Test Author Guide on page 204](#).

- Installed Apache Ant. For more information, refer to the Installing Apache Ant section in the [Apache Ant™ User Manual](#).
- Downloaded the Rational® Performance Tester `Ant` plugin from the [IBM WebSphere, Liberty & DevOps Community portal](#) onto the computer where you install Rational® Performance Tester.

For more information about specific versions of plugin, see [Integration plugin compatibility matrix on page 68](#).

- Closed Rational® Performance Tester.

About this task

You must provide the values for certain parameters in the `Ant` test script to run tests from the command-line interface. The minimum parameters that you need to provide are **name**, **workspace**, **projectname**, and **suite** or **configfile**. The remaining parameters are optional.

For example, if you provide a complete path of a configuration file that contains the parameters for a `test` or `schedule` run in the **configfile** parameter, then you need not be required to provide values for the **name**, **workspace**, **projectname**, and **suite** parameters.



Note: If Rational® Performance Tester is not installed in default location, then you must provide the value for the **imsharedloc** parameter along with other mandatory parameters.



Remember: You must consider the following requirements:

- Enter the parameter values within the double quotation marks.
- Ensure that the special characters in the parameter values do not break the validation of the XML file. For example, you must enter the `&` character as `&`.

1. Extract the following files from the downloaded `Ant` plugin:

The `Ant` plugin includes the following files:

- `RPT-Ant-x.y.jar`

Where `x.y` is the version number of the `Ant` plugin.

- `README.txt`



2. Open the `ExecutePerformanceTest.xml` file in a text editor and provide the required parameter values within `<pt>` tag by referring to the following table:


The following table lists the required parameters that you must provide to run the test from Ant:




Parameters	Description
configfile	<p>Enter the complete path to a configuration file that contains the parameters for a <code>test</code> or <code>schedule run</code>.</p> <p>For example,</p> <pre><pt configfile=C:\Users\IBM\rational\workspace\Project1\config.txt /></pre> <p>For more information about creating a command line config file from Rational® Performance Tester, see the Related information section.</p>
name	<p>Enter a name for the <code>Ant</code> task.</p> <p>For example,</p> <pre><pt name="runtest_ant" /></pre>
projectname	<p>Enter the name of the project that has performance test assets.</p> <p>For example,</p> <pre><pt name="runtest_ant" workspace="C:/Users/IBM/rational\workspace1" projectname="proj123"/></pre>
suite	<p>Enter the name of the test asset that you want to run.</p> <p>For example,</p> <pre><pt name="runtest_ant" workspace="C:/Users/IBM/rational\workspace1" projectname="proj123" suite="test1.testsuite"/></pre>
workspace	<p>Enter the complete path of the Eclipse workspace.</p> <p>For example,</p> <pre><pt name="runtest_ant" workspace="C:/Users/IBM/rational\workspace1"/></pre>

The following table lists the optional parameters that you can provide to run the test from Ant:



Parameters	Description
duration	<p>Use this parameter to change the duration of the stages in the rate schedule.</p> <p>For example,</p> <pre><pt name="runtest_ant" workspace="C:/Users/IBM/rational\workspace1" projectname="proj123" suite="test1.testsuite" duration="Stage1=10s, Stage2=3m"/></pre>



Parameters	Description
	<p>The stage number specified must exist in the rate schedule.</p> <p> Note: The duration parameter creates a new copy of the rate schedule that contains the specified number of duration.</p>
exportlog	<p>Use this parameter to specify the path of the file directory to store the exported HTTP test log in a <code>.txt</code> format.</p> <p>For example,</p> <pre data-bbox="526 659 1419 751"><pt name="runtest_ant" workspace="C:/Users/IBM/rational_sdp/workspace1" projectname="proj123" suite="test1.testsuite" exportlog="C:/Users/Documents/ANT/LogExp.txt"/></pre> <p> Note: When you do not use the exportlog parameter, then the <code>Ant</code> execution output is logged into the <code>logfile.txt</code> file and stored in a <code>temp</code> directory where you extracted the files of the <code>Ant</code> plugin.</p>
exportstatreportlist	<p>Use this parameter to specify IDs of reports that you want to export in place of the default reports.</p> <p>You can provide multiple report IDs separated by a comma. You can navigate to Preferences of Rational® Performance Tester (Window > Preferences > Test > Performance Test Reports > Export Reports), and then select Show Report IDs checkbox to view the report IDs.</p> <p>You must use the exportstatreportlist parameter along with the exportstats or exportstatshtml parameter.</p> <p>For example, you can provide the value for the exportstatreportlist parameter as follows to export the Performance Report:</p> <pre data-bbox="526 1493 1419 1604"><pt name="runtest_ant" workspace="C:/Users/IBM/rational_sdp/workspace1" projectname="proj123" suite="test1.testsuite" exportstats="C:/Users/Documents/ANT" exportstatreportlist="http"/></pre>
exportstats	<p>Use this parameter to provide the complete path to a directory that you can use to store the exported report in a comma-separated values (CSV) format.</p> <p>For example, you can provide the value for the exportstatreportlist parameter as follows to export the Performance Report:</p>


Parameters	Description
	<pre data-bbox="527 262 1344 346"><pt name="runtest_ant" workspace="C:/Users/IBM/rationalSDP/workspace1" projectname="proj123" suite="test1.testsuite" exportstats="C:/Users/Documents/ANT"/></pre> <p data-bbox="527 388 1409 556"> Note: If you do not specify a value for the exportstatereportlist parameter, then the reports that you selected in Preferences of Rational® Performance Tester (Window > Preferences > Test > Performance Test Reports > Export Reports) are exported.</p>
exportstatsformat	<p data-bbox="511 640 1377 709">Use this parameter to specify a format for the result that you want to export along with the exportstats parameter.</p> <p data-bbox="511 745 1393 814">You must use at least one of the following options with the exportstatsformat parameter:</p> <ul data-bbox="576 829 711 1060" style="list-style-type: none"> ◦ simple.csv ◦ full.csv ◦ simple.json ◦ full.json ◦ csv ◦ json <p data-bbox="511 1096 646 1123">For example,</p> <pre data-bbox="527 1144 1344 1260"><pt name="runtest_ant" workspace="C:/Users/IBM/rationalSDP/workspace1" projectname="proj123" suite="test1.testsuite" exportstats="C:/Users/Documents/ANT" exportstatsformat="simple.json"/></pre> <p data-bbox="511 1291 1075 1318">You can add multiple options separated by a comma.</p> <p data-bbox="511 1354 646 1381">For example,</p> <pre data-bbox="527 1402 1344 1518"><pt name="runtest_ant" workspace="C:/Users/IBM/rationalSDP/workspace1" projectname="proj123" suite="test1.testsuite" exportstats="C:/Users/Documents/ANT" exportstatsformat="simple.json,full.csv"/></pre> <p data-bbox="511 1554 1399 1623">When you want to export both simple and full type of test results in a <code>json</code> or <code>csv</code> format, you can specify either <code>json</code> or <code>csv</code> as options.</p> <p data-bbox="511 1659 646 1686">For example,</p> <pre data-bbox="527 1707 1344 1822"><pt name="runtest_ant" workspace="C:/Users/IBM/rationalSDP/workspace1" projectname="proj123" suite="test1.testsuite" exportstats="C:/Users/Documents/ANT" exportstatsformat="json"/></pre> <p data-bbox="511 1848 1421 1875">When the test run completes, the test result exports to <code>simple.json</code> and <code>full.json</code> files.</p>

Parameters	Description
	 Note: The values provided in the <code>ExecutePerformanceTest.xml</code> file always take precedence over the Export Reports options set in the Preferences of Rational® Performance Tester (Window > Preferences > Test > Performance Test Reports > Export Reports).
exportstatshhtml	<p>Use this parameter to provide the complete path to a directory that you can use to export web analytic results.</p> <p>The results are exported to the specified directory. You can analyze the results on a web browser without using Rational® Performance Tester.</p> <p>For example,</p> <pre data-bbox="526 785 1425 873"><pt name="runtest_ant" workspace="C:/Users/IBM/rationalsdp/workspace1" projectname="proj123" suite="test1.testsuite" exportstatshhtml="C:/Users/Documents/ANT"/></pre>  Note: When you do not use the exportstatshhtml parameter, then the Performance report is stored in a <code>temp</code> directory where you extracted the files of the <code>Ant</code> plugin.
history	<p>Use this parameter when you want to view a record of all events that occurred during a <code>test</code> or <code>schedule</code> run.</p> <p>You can use any of the following options:</p> <ul style="list-style-type: none"> ◦ jaeger: To send test logs to the Jaeger UI during the <code>test</code> or <code>schedule</code> run.  Note: You must set the <code>JAEGER_AGENT_HOST</code> property as an environment variable by using the command line before you use the jaeger option to send test logs to the Jaeger UI. <ul style="list-style-type: none"> ◦ testlog: To send test logs as traditional test logs in Rational® Performance Tester during the <code>test</code> or <code>schedule</code> run. ◦ null: To send no test logs either to the Jaeger UI or Rational® Performance Tester during the <code>test</code> or <code>schedule</code> run. <p>For example,</p>


Parameters	Description								
	<pre data-bbox="527 262 1344 346"><pt name="runtest_ant" workspace="C:/Users/IBM/rationalSDP/workspace1" projectname="proj123" suite="test1.testsuite" history="jaeger"/></pre> <p data-bbox="511 380 1414 449">You can add multiple options separated by a comma to send test logs during the <code>test</code> or <code>schedule run</code> to Rational® Performance Tester and the Jaeger UI.</p> <p data-bbox="511 485 646 512">For example,</p> <pre data-bbox="527 535 1344 619"><pt name="runtest_ant" workspace="C:/Users/IBM/rationalSDP/workspace1" projectname="proj123" suite="test1.testsuite" history="jaeger,testlog"/></pre> <p data-bbox="511 655 1414 724">For more information about how to view test logs in the Jaeger UI and Rational® Performance Tester, see the Related information section.</p>								
imsharedloc	<p data-bbox="511 787 1414 856">Use this parameter to enter the complete path to the IBMIMShared location, if it is not in the default location.</p> <p data-bbox="511 892 1154 919">The default location of the IMShared directory is as follows:</p> <table border="1" data-bbox="511 947 1414 1171"> <thead> <tr> <th data-bbox="511 947 813 1003">Operating system</th> <th data-bbox="813 947 1414 1003">The default path to the directory</th> </tr> </thead> <tbody> <tr> <td data-bbox="511 1003 813 1060">Windows®</td> <td data-bbox="813 1003 1414 1060">C:\Program Files\IBM\IBMIMShared</td> </tr> <tr> <td data-bbox="511 1060 813 1117">Linux™</td> <td data-bbox="813 1060 1414 1117">/opt/IBM/IBMIMShared</td> </tr> <tr> <td data-bbox="511 1117 813 1171">Mac</td> <td data-bbox="813 1117 1414 1171">/Application/IBM/IBMIMShared</td> </tr> </tbody> </table> <p data-bbox="511 1207 646 1234">For example,</p> <pre data-bbox="527 1257 1344 1341"><pt name="runtest_ant" workspace="C:/Users/IBM/rationalSDP/workspace1" projectname="proj123" suite="test1.testsuite" imsharedloc="D:\Testtool\IBM\SDP"/></pre>	Operating system	The default path to the directory	Windows®	C:\Program Files\IBM\IBMIMShared	Linux™	/opt/IBM/IBMIMShared	Mac	/Application/IBM/IBMIMShared
Operating system	The default path to the directory								
Windows®	C:\Program Files\IBM\IBMIMShared								
Linux™	/opt/IBM/IBMIMShared								
Mac	/Application/IBM/IBMIMShared								
labels	<p data-bbox="511 1392 1328 1419">Use this parameter to add labels to test results when the test run is complete.</p> <p data-bbox="511 1455 646 1482">For example,</p> <pre data-bbox="527 1509 1344 1593"><pt name="runtest_ant" workspace="C:/Users/IBM/rationalSDP/workspace1" projectname="proj123" suite="test1.testsuite" labels="label1, label2"/></pre> <p data-bbox="511 1629 1414 1734">You can add multiple labels to a test result separated by a comma. When you run test assets, then the labels that you added are displayed on the Performance Report in Rational® Performance Tester.</p> <p data-bbox="511 1770 1370 1839">The Results page of Rational® Test Automation Server displays the label that you added, in the specific test asset in the following conditions:</p>								



Parameters	Description
	<ul style="list-style-type: none"> ◦ When you set Publish result after execution as Always or Prompt in the Preferences of Rational® Performance Tester (WindowsPreferencesTestRational Test Automation Server). ◦ When you use the labels parameter along with the publish parameter. <p> Note: If the value of the labels parameter contains double quotation marks (""), for example, "100" users, then the test run fails due to non-validation of xml file.</p> <p>To work around this problem, you must create a command-line config file, and then run the test by using the configfile parameter.</p>
overridermlabels	<p>Use this parameter to perform any of the following actions:</p> <ul style="list-style-type: none"> ◦ Enable the Resource Monitoring from Service option for a performance <code>schedule</code> if the Resource Monitoring from Service option is not enabled from the schedule editor in Rational® Performance Tester. ◦ Ignore Resource Monitoring sources that were set in the performance schedule and change for a label matching mode. ◦ Replace an existing set of Resource Monitoring labels that were set in the performance schedule and run the schedule with a new set of Resource Monitoring labels. <p>If you have added a label in Rational® Test Automation Server for a Resource Monitoring source as <code>rm1</code>, then you can provide value as shown in the following example to collect data from the source:</p> <pre data-bbox="526 1402 1419 1493"><pt name="runtest_ant" workspace="C:/Users/IBM/rationalstdp/workspace1" projectname="proj123" suite="test1.testsuite" overridermlabels="rm1"/></pre> <p>If your label for resource monitoring contains a comma (,) then you must replace the single comma with the double comma while providing the value. For example, if you added a label to a Resource Monitoring source as <code>rm1,test</code>, then you must provide the value as shown in the following example to collect data from source:</p> <pre data-bbox="526 1696 1419 1787"><pt name="runtest_ant" workspace="C:/Users/IBM/rationalstdp/workspace1" projectname="proj123" suite="test1.testsuite" overridermlabels="rm1,,test"/></pre> <p> Notes:</p>

Parameters	Description
	 <ul style="list-style-type: none"> ◦ You can use this field only when you want to run a Rate schedule or VU schedule. ◦ You can add multiple Resource Monitoring labels separated by a comma. ◦ You must add the Resource Monitoring labels to the Resource Monitoring sources on the Resource Monitoring page in the Rational® Test Automation Server project.
overwrite	<p>Set this parameter value to <code>true</code> or <code>false</code> to determine whether a result file with the same name must be overwritten or not.</p> <p>The default value is <code>false</code>.</p> <p>For example,</p> <pre data-bbox="526 877 1417 968" style="background-color: #f0f0f0; padding: 5px;"> <pt name="runtest_ant" workspace="C:/Users/IBM/rationalsdp/workspace1" projectname="proj123" suite="test1.testsuite" overwrite="true"/></pre>
publish	<p>Use this parameter to publish test results to Rational® Test Automation Server by using <code>ANT</code>.</p> <p> Remember: Before you use the publish parameter, you must provide the offline user token of Rational® Test Automation Server by using any of the following methods:</p> <ul style="list-style-type: none"> ◦ Set the value of the RTCP_OFFLINE_TOKEN environment variable to the offline user token of Rational® Test Automation Server by using the command-line interface. ◦ Provide the offline user token of Rational® Test Automation Server in the preferences of Rational® Performance Tester (Window > Preferences > Test > Rational Test Automation Server) <p>You must use one of the following values in the publish parameter:</p> <ul style="list-style-type: none"> ◦ <code>serverURL#project.name=name_of_the_project&teamspace.name=name_of_the_teamspace</code> ◦ <code>serverURL#project.name=name_of_the_project&teamspace.alias=name_of_the_teamspace_alias</code>

Parameters	Description
	<ul style="list-style-type: none"> ◦ no <p>The no option is useful if the product preferences are set to publish the results, but you do not want to publish them.</p> <p>For example:</p> <pre data-bbox="526 478 1417 596"><pt name="runtest_ant" workspace="C:/Users/IBM/rationalmdp/workspace1" projectname="proj123" suite="test1.testsuite" publish= "URL_OF_RationalTest_Server#project.name=test&team.space.name=ts1"/></pre> <pre data-bbox="526 617 1417 701"><pt name="runtest_ant" workspace="C:/Users/IBM/rationalmdp/workspace1" projectname="proj123" suite="test1.testsuite" publish=no"/></pre> <p>Where:</p> <ul style="list-style-type: none"> ◦ <code>URL_OF_RationalTest_Server</code> is the URL of Rational® Test Automation Server. ◦ <code>test</code> is the name of the project in Rational® Test Automation Server. ◦ <code>ts1</code> is the name of the team space. <p> Notes:</p> <ul style="list-style-type: none"> ◦ If you have a project with the same name in different team spaces, then you must append either the &team.space.name=name_of_the_team.space or &team.space.alias=name_of_the_team.space_alias options. ◦ If the name of the project or team space contains a special character, then you must replace it with <code>%<Hexvalue_of_special_character></code>. <p>For example, if the name of the team space is <i>Initial Team Space</i>, then you must provide it as <i>Initial%20Team%20Space</i>.</p> <p>Where <code>%20</code> is the Hex value of <code>Space</code> character.</p> <ul style="list-style-type: none"> ◦ The values provided in the Publish field always take precedence over the Results options set in the product preferences (Window > Preferences > Test > Rational Test Automation Server > Results). <p>The Reports information section on the <code>Log</code> file displays the names of the report along with its corresponding URLs in the following conditions:</p>

Parameters	Description
	<ul style="list-style-type: none"> ◦ When you configured the URL of in Preferences of Rational® Performance Tester (Window > Preferences > Test > Rational Test Automation Server). ◦ When you set Publish result after execution as Always or Prompt in the Preferences of Rational® Performance Tester (Window > Preferences > Test > Rational Test Automation Server > Results). ◦ When you used publish parameter or publish parameter along with publishre-ports parameter.
publish_for	<p>Use this parameter to publish test results to Rational® Test Automation Server by using <code>ANT</code> based on the completion status of the tests.</p> <p>You must use the publish_for parameter along with the publish parameter. You must add multiple parameters separated by a comma. The available options that you can use for the publish_for parameter are as follows:</p> <ul style="list-style-type: none"> ◦ ALL: You can use this option irrespective of status of the test. ◦ PASS: You can use this option to publish test results for the tests that have passed. ◦ FAIL: You can use this option to publish test results for the tests that have failed. ◦ ERROR: You can use this option to publish test results for the tests that included errors. ◦ INCONCLUSIVE: You can use this option to publish test results for the inconclusive tests. <p>For example,</p> <pre><pt name="runtest_ant" workspace="C:/Users/IBM/rationalSDP/workspace1" projectname="proj123" suite="test1.testsuite" publish= "URL_OF_RationalTest_Server#project.name=test&teamspace.name=ts1" publish_for="FAIL,ERROR"/></pre>
publishreports	<p>Use this parameter to publish specific test results to Rational® Test Automation Server.</p> <p>You must use the publishreports parameter along with the publish parameter. The options that you can use with publishreports are as follows:</p> <ul style="list-style-type: none"> ◦ STATS ◦ TESTLOG

Parameters	Description
	<p>For example,</p> <pre data-bbox="526 310 1419 457"><pt name="runtest_ant" workspace="C:/Users/IBM/rational_sdp/workspace1" projectname="proj123" suite="test1.testsuite" publish= "URL_OF_RationalTest_Server#project.name=test&team.space.name=ts1" publishreports="STATS"/></pre> <p>You can prefix the publishreports value with ! to publish the reports except for the specified one in the test script.</p> <p>For example,</p> <pre data-bbox="526 646 1419 793"><pt name="runtest_ant" workspace="C:/Users/IBM/rational_sdp/workspace1" projectname="proj123" suite="test1.testsuite" publish= "URL_OF_RationalTest_Server#project.name=test&team.space.name=ts1" publishreports="!TESTLOG"/></pre>
rate	<p>Use this parameter to change the rate of the rate runner group.</p> <p>For example,</p> <pre data-bbox="526 953 1419 1037"><pt name="runtest_ant" workspace="C:/Users/IBM/rational_sdp/workspace1" projectname="proj123" suite="test1.testsuite" rate="Rate Runner Group1=1/s, 3/m"/></pre> <p>Where, <i>Rate Runner Group1</i> is the name of the rate runner group that has two stages. The desired rate for the first stage is one iteration per second and the rate for the second stage is three iterations per minute.</p> <p> Notes:</p> <ul data-bbox="646 1276 1419 1430" style="list-style-type: none"> ◦ The name of the Rate Runner group must match with the name in the rate schedule. ◦ The rate parameter creates a new copy of the rate schedule that contains the specified number of rate.
results	<p>Use this parameter to provide a different name to the result file.</p> <p>The default name of the result file is the name of the <i>test</i> or <i>schedule</i> with a time-stamp appended.</p>
swapdatasets	<p>Use this parameter to replace the dataset values during a <i>test</i> or <i>schedule</i> run.</p> <p>You must ensure that both original and new datasets are in the same workspace and have the same column names. When you enter a value for the swapdatasets parame-</p>

Parameters	Description
	<p>ter, you must also include the path to the dataset. You must provide the values for the swapdatasets parameter in the following format:</p> <pre data-bbox="511 373 1317 436">/project_name/ds_path/original_ds.csv:/project_name/ds_path/new_ds.csv</pre> <p>For example,</p> <pre data-bbox="526 527 1403 638"><pt name="runtest_ant" workspace="C:/Users/IBM/rationalsdp/workspace1" projectname="proj123" suite="test1.testsuite" swapdatasets="/proj1023/Datasets/ds1.csv:/proj1023/Datasets/new_ds1.csv"/></pre> <p>You can override multiple datasets that are saved in a different project by adding multiple paths to the dataset separated by a semicolon. For a <code>test</code> or <code>schedule</code>, the default value is the dataset specified in the test editor or schedule editor.</p>
usercomments	<p>Use this parameter to add text that you want to display in the user comments row of the report.</p> <p>For example,</p> <pre data-bbox="526 1003 1344 1087"><pt name="runtest_ant" workspace="C:/Users/IBM/rationalsdp/workspace1" projectname="proj123" suite="test1.testsuite" usercomments="test run with dataset"/></pre> <p> Note: If the value of the usercomments parameter contains double quotation marks (""), for example, <code>test run with "dataset"</code>, then the test run fails due to non-validation of xml file.</p> <p>To work around this problem, you must create a command-line config file, and then run the test by using the configfile parameter.</p>
users	<p>Use this parameter to overrides the default number of users, if required.</p> <p>For a <code>schedule</code>, the default is the number of users specified in the schedule editor and for a <code>test</code>, the default is one user.</p> <p> Note: The users parameter creates a new copy of the <code>schedule</code> that contains the specified number of users.</p>
varfile	<p>Use this parameter to provide a complete path to an XML file that contains the variable name and value pairs.</p>

Parameters	Description
vmargs	Use this parameter to specify the maximum heap size for the Java process. You can add multiple VM Arguments separated by a comma.



Note: You can add multiple `<pt>` tasks and provide the details for each test to run multiple tests simultaneously.

3. Save the `ExecutePerformanceTest.xml` file.
4. Open the command prompt, and then change the directory to a location where you downloaded the `Ant` plugin.

For example,

```
cd D:\Downloads\Ant\RPT-Ant-x.y
```

Where `x.y` is the version number of the `Ant` plugin.

5. Run the following command to initiate the test run:

```
ant -f ExecutePerformanceTest.xml
```

Results

You have run the test from the command-line interface by using `Ant`.

What to do next

You can view the `Ant` execution output from the `logfile.txt` file that is stored in a `temp` directory. The `temp` directory is located where you extracted the files of the `Ant` plugin. If you provided the location for the **exportlog** parameter, then the `Ant` execution output is stored in the specified location.

The **Reports information** section on the `Log` file displays the names of the report along with its corresponding URLs if you publish test results to Rational® Test Automation Server.

Related information

[Creating a command-line config file on page 370](#)

[Viewing test logs in Jaeger on page 156](#)

[Viewing test logs on page 467](#)

[Generating offline user tokens](#)

Integration with Azure DevOps

When you use Azure DevOps for continuous integration and continuous deployment of your application, you can create tests for your application in and run those tests in Azure DevOps pipelines.

You must use the **IBM Rational Test Workbench** extension from the Visual Studio Marketplace portal to integrate with Azure DevOps.

The following table lists the tasks that you must perform to run performance test assets from Azure DevOps:

Tasks	Go to...
Install Rational® Performance Tester.	.
Create any or all the following types of test assets in to test your application: <ul style="list-style-type: none"> • Compound tests • Performance tests • Schedules (Rate or VU Schedules) 	Test Author Guide on page 204.
Install at least one agent in Azure Pipelines.	Azure DevOps documentation.
Create an Organization and a Project in Azure DevOps to run test assets as jobs in Azure DevOps pipelines.	Azure DevOps documentation.
Install the IBM Rational Test Workbench extension.	Installing the IBM Rational Test Workbench extension on page 83
Run the test assets from an Azure DevOps pipeline.	Running tests from Azure DevOps as a Pipeline job on page 84

Installing the IBM Rational Test Workbench extension

You must install the **IBM Rational Test Workbench** extension for an **Organization** in Azure DevOps to run tests from an Azure DevOps pipeline.

Before you begin

- You must be familiar with working with Azure DevOps.
- You must have access to the following portals:
 - Azure DevOps
 - Visual Studio Marketplace

1. Log in to the **Visual Studio Marketplace** portal, if you are not already logged in.
2. Click the **Azure DevOps** tab.
3. Search for the **IBM Rational Test Workbench** extension.
4. Click the **IBM Rational Test Workbench** extension.
5. Click **Get it free**.

Result

The **Visual Studio Marketplace** portal for the **IBM Rational Test Workbench** is displayed.

6. Select the **Organization** from the drop-down list.
7. Click **Install**.

Result

The installation process begins and after a few minutes the installation is complete.

8. Click **Proceed to organization**.

Result

The **Organization** page in Azure DevOps is displayed.

9. Click **Organization settings**, and then **Extensions**.

Result

The **IBM Rational Test Workbench** extension is displayed as an installed extension.

Results

You have installed the **IBM Rational Test Workbench** extension for the **Organization** in Azure DevOps.

What to do next

You can add the details of performance test assets to task, and then run the tests from an Azure DevOps pipeline.

See [Running tests in an Azure DevOps Pipeline on page 84](#).

Running tests from Azure DevOps as a Pipeline job

You can run test assets from Azure DevOps as a Pipeline job to test the performance of your application under test.

Before you begin

- You must be familiar with working with Azure DevOps.
- You must have completed the following tasks:
 - Been granted access to the Azure DevOps portal.
 - Created an **Organization** in Azure DevOps.
 - Created and configured the repository for the project in Azure DevOps.
 - Started the Azure DevOps pipeline agent.
 - Closed Rational® Performance Tester.
 - **Optional:** Created test cases under test plans in Azure DevOps if you want to view the results of the test runs on the **Test Plans** dashboard.
 - **Optional:** Created encrypted secrets or environment variables for your personal access token (PAT), if you want to use an encrypted PAT.

About this task

You must provide the values for certain fields for the selected task to run tests from Azure DevOps. The minimum fields that you need to configure along with **Display name** are **Workspace Location**, **Project Name**, and **Test Suite Name** or **Configfile**. The remaining fields are optional.

For example, if you provide a complete path of a configuration file that contains the parameters for a `test` or `schedule` run in the **Configfile** field, then you need not provide values for the **Workspace Location**, **Project Name**, and **Test Suite Name** fields.

If you have created test cases under test plans in your Azure DevOps project, you can provide the details in the **Azure DevOps URL**, **Test Plan Name**, **Test Case Name**, and **Azure DevOps PAT** fields while you configure the job in a pipeline so that you can view the results of the test run on your Test Plan dashboard.

1. Sign in to the Azure DevOps portal, if you are not already logged in.
2. Open the **Organization** page, and then select the project that you want to use to run tests.
3. Initialize the repository if you have created a new project by performing the following steps:
 - a. Click **Repos** from the left pane.
 - b. Select the **Add a README** checkbox if it is not selected.
 - c. Click **Initialize** from the **Initialize <branch_name> with a README or gitignore** section.
4. Click **Pipelines** from the left pane, and then click **Create Pipeline**.




Note: If your project already contains a pipeline, then you can click **New Pipeline** to create another pipeline.

5. Click **Use the classic editor** to create a pipeline without YAML.
6. Verify the values for the **Team project**, **Repository**, and **Default branch for manual and scheduled builds** fields, and then click **Continue**.
7. Click **Empty job** to add a task to the pipeline.



Note: The values for the **Name**, **Agent pool**, and **Agent Specification** fields are auto populated. You can change the values for the fields, if required.

8. Perform the following steps to add a task to the **Agent job**:
 - a. Click the **Add a task to Agent job** icon .

Result
The **Add tasks** pane is displayed.
 - b. Search for the IBM tasks defined in the **IBM Rational Test Workbench** extension.


Result
The tasks that you can select are displayed.
 - c. Select the **IBM Rational Performance Tester Task** option, and then click **Add**.

The selected task is added to the agent job and it is displayed with a warning that some settings require attention. When you configure the details about the test run, the warning disappears.

You can also remove the tasks that are not required in your job by selecting the tasks from the list. You can then right-click the tasks, and click **Remove selected task(s)** to remove them.


- Configure **IBM Rational Performance Tester Task** to provide the details about the test run by referring to the following table:


The following table lists the required fields that you must provide to run the test from Azure DevOps pipeline:



Fields	Description								
Configfile	<p>Enter the complete path to a file that contains the parameters for a <code>test</code> or <code>schedule</code> run.</p> <p>For example, <code>C:\Users\IBM\rational\workspace\Project1\config.txt</code></p> <p>For more information about creating a command line config file from Rational® Performance Tester, see the Related information section.</p> <p> Remember: If you provide a complete path of a configuration file in the Config File field, then you need not provide values for the Workspace, Project, and Test Suite Name fields.</p>								
Display name	<p>Enter the name for the task.</p> <p>For example, <code>testrun1</code></p>								
IMShared Path	<p>Enter the complete path to the <code>IMShared</code> directory, if it is not in the default location.</p> <p>For example, <code>D:\Testtool\IBM\IBMIMShared</code></p> <p>The default location of the <code>IMShared</code> directory is as follows:</p> <table border="1" data-bbox="511 1486 1421 1715"> <thead> <tr> <th data-bbox="511 1486 813 1541">Operating system</th> <th data-bbox="813 1486 1421 1541">The default path to the directory</th> </tr> </thead> <tbody> <tr> <td data-bbox="511 1541 813 1598">Windows®</td> <td data-bbox="813 1541 1421 1598">C:\Program Files\IBM\IBMIMShared</td> </tr> <tr> <td data-bbox="511 1598 813 1654">Linux™</td> <td data-bbox="813 1598 1421 1654">/opt/IBM/IBMIMShared</td> </tr> <tr> <td data-bbox="511 1654 813 1715">Mac</td> <td data-bbox="813 1654 1421 1715">/Application/IBM/IBMIMShared</td> </tr> </tbody> </table>	Operating system	The default path to the directory	Windows®	C:\Program Files\IBM\IBMIMShared	Linux™	/opt/IBM/IBMIMShared	Mac	/Application/IBM/IBMIMShared
Operating system	The default path to the directory								
Windows®	C:\Program Files\IBM\IBMIMShared								
Linux™	/opt/IBM/IBMIMShared								
Mac	/Application/IBM/IBMIMShared								
Project Name	<p>Enter the name of the project that has performance test assets.</p> <p>For example, <code>testproj1</code></p>								


Fields	Description
Test Suite Name	Enter the file name of the performance test assets relative to the project. For example, <i>test1.testsuite</i>
Workspace Location	Enter the complete path of the Eclipse workspace. For example, <i>C:/Users/IBM/rationalmdp/workspace1</i>



The following table lists the optional fields that you can provide to run the test from Azure DevOps pipeline:



Fields	Description
Azure DevOps PAT	Use this field to enter the personal access token of the Azure server where the test project is hosted. You can also enter the variable for the encrypted PAT that you create in Azure DevOps. The token must have read and write access. You must use Azure DevOps PAT with the following fields: <ul style="list-style-type: none"> ◦ Azure DevOps Project URL ◦ Test Plan Name ◦ Test Case Name
Azure DevOps Project URL	Use this field to enter the URL of the test project in the Azure server in the following format: <code>https://<host>/<orgname>/<projectName></code>  Note: You must enter the URL of the Azure DevOps project if you have created test cases under Test Plans in your Azure project and want to view the test results on the Test Plans dashboard. You must use Azure DevOps Project URL with the following fields: <ul style="list-style-type: none"> ◦ Azure DevOps PAT ◦ Test Plan Name ◦ Test Case Name
Dataset Override	Use this field to replace the dataset values during a <code>test</code> or <code>schedule</code> run. You must ensure that both original and new datasets are in the same workspace and have the same column names. When you enter a value for the Dataset Override field,

Fields	Description
	<p>you must also include the path to the dataset. You must provide the values for the Dataset Override field in the following format:</p> <pre data-bbox="511 373 1312 436">/project_name/ds_path/original_ds.csv:/project_name/ds_path/new_ds.csv</pre> <p>For example, <i>/proj1/Datasets/ds1.csv:/proj1023/Datasets/new_ds1.csv</i></p> <p>You can override multiple datasets that are saved in a different project by adding multiple paths to the dataset separated by a semicolon. For a <code>test</code> or <code>schedule</code>, the default value is the dataset specified in the test editor or schedule editor.</p>
Duration	<p>Use this field to change the duration of the stages in the rate schedule.</p> <p>For example, <i>Stage1=10s, Stage2=3m</i></p> <p>The stage number specified must exist in the rate schedule.</p> <p> Note: The Duration field creates a new copy of the rate schedule that contains the specified number of duration.</p>
Exportlog	<p>Use this field to specify the path of the file directory to store the exported HTTP test log in a <code>.txt</code> format.</p> <p>For example, <i>C:/Users/Documents/tests/logexp.txt</i></p>
Exportstatreportlist	<p>Use this field to specify IDs of reports that you want to export in place of the default reports.</p> <p>You can provide multiple report IDs separated by a comma. You can navigate to Preferences of Rational® Performance Tester (Window > Preferences > Test > Performance Test Reports > Export Reports), and then select Show Report IDs checkbox to view the report IDs.</p> <p>You must use the Exportstatreportlist field along with the Exportstats or Exportstatshhtml field.</p> <p>For example, you can provide <code>http</code> as a value in the Exportstatreportlist field to export a Performance Report.</p>
Exportstats	<p>Use this field to provide the complete path to a directory that you can use to store exported reports in a comma-separated values (CSV) format.</p>



Fields	Description
	<p>For example, <i>C:/Users/Documents/tests</i></p> <p> Note: If you do not specify a value for the Exportstatreportlist field, then the reports that you selected in the Preferences of Rational® Performance Tester (Window > Preferences > Test > Performance Test Reports > Export Reports) are exported.</p>
Exportstatsformat	<p>Use this field to specify a format for the report that you want to export.</p> <p>You must use at least one of the following options along with the Exportstats field:</p> <ul style="list-style-type: none"> ◦ <i>simple.csv</i> ◦ <i>full.csv</i> ◦ <i>simple.json</i> ◦ <i>full.json</i> ◦ <i>csv</i> ◦ <i>json</i> <p>For example, <i>json</i>.</p> <p>You can add multiple formats for the report separated by a comma. When you want to export both simple and full type of report in a <i>json</i> or <i>csv</i> format, you can specify <i>json</i> or <i>csv</i> as the format in the field.</p> <p>The reports are saved to the location specified in the Exportstats field.</p> <p> Note: The values provided in the Exportstatsformat field always take precedence over the Export Reports options set in the product preferences Window > Preferences > Test > Performance Test Reports > Export Reports).</p>
Exportstatshhtml	<p>Use this field to provide the complete path to a directory that you can use to export web analytic results.</p> <p>The results are exported to the specified directory. You can analyze the results on a web browser without using Rational® Performance Tester.</p> <p>For example, <i>C:/Users/Documents/Reports</i></p>



Fields	Description
History	<p>Use this field when you want to view a record of all events that occurred during a test or schedule run.</p> <p>You can use any of the following options:</p> <ul style="list-style-type: none"> ◦ jaeger: To send test logs to the Jaeger UI during the <code>test</code> or <code>schedule</code> run. <p> Note: You must set the <code>JAEGER_AGENT_HOST</code> property as an environment variable by using the command line before you use the jaeger option to send test logs to the Jaeger UI.</p> <ul style="list-style-type: none"> ◦ testlog: To send test logs as traditional test logs in Rational® Performance Tester during the <code>test</code> or <code>schedule</code> run. ◦ null: To send no test logs either to the Jaeger UI or Rational® Performance Tester during the <code>test</code> or <code>schedule</code> run. <p>For example, <i>jaeger</i></p> <p>You can add multiple options separated by a comma to send test logs during the <code>test</code> or <code>schedule</code> run to Rational® Performance Tester and the Jaeger UI.</p> <p>For example, <i>jaeger,testlog</i></p> <p>For more information about how to view test logs in the Jaeger UI and Rational® Performance Tester, see the Related information section.</p>
Labels	<p>Use this field to add labels to test results when the test run is complete.</p> <p>For example, <i>label1, label2</i></p> <p>You can add multiple labels to a test result separated by a comma. When you run test assets, then the labels that you added are displayed on the Performance Report in Rational® Performance Tester.</p> <p>The Results page of Rational® Test Automation Server displays the label that you added in the Labels field for the specific test asset in the following conditions:</p> <ul style="list-style-type: none"> ◦ When you set Publish result after execution as Always or Prompt in the preferences of Rational® Performance Tester (Windows > Preferences > Test > Rational Test Automation Server). ◦ When you use the Labels field along with the Publish field.



Fields	Description
	<p> Note: When you run tests by using the double quotation marks ("") for the Labels field, then the labels in the test result do not include double quotation marks.</p> <p>For example, if you provide the value for the Labels field as "100" users, then the labels in the test result display as 100 users.</p> <p>To work around this problem, you must create a command-line config file, and then run the test by using the Configfile field.</p>
Overwrite	<p>Use this field to determine whether a result file with the same name is overwritten or not.</p> <p>The default value is <code>false</code>. If the value is set to <code>true</code>, the file is overwritten and retains the same file name.</p>
Publish	<p>Use this field to publish test results to Rational® Test Automation Server from Azure DevOps.</p> <p> Remember: Before you use the Publish field, you must provide the offline user token of Rational® Test Automation Server by using any of the following methods:</p> <ul style="list-style-type: none"> ◦ Set the value of the RTCP_OFFLINE_TOKEN environment variable to the offline user token of Rational® Test Automation Server by using the command-line interface. ◦ Provide the offline user token of Rational® Test Automation Server in the preferences of Rational® Performance Tester (Window > Preferences > Test > Rational Test Automation Server) <p>You must use one of the following values in the Publish field:</p> <ul style="list-style-type: none"> ◦ <code>serverURL#project.name=name_of_the_project&team.space.name=name_of_the_team.space</code> ◦ <code>serverURL#project.name=name_of_the_project&team.space.alias=name_of_the_team.space_alias</code>


Fields	Description
	<ul style="list-style-type: none"> ◦ no <p>The no option is useful if the product preferences are set to publish the results, but you do not want to publish them.</p> <p> Note:</p> <p>You must not include the double quotation marks (") while providing the value for the Publish field.</p> <p>For example, <i>URL_OF_RationalTest_Server#project.name=test&team.space.name=ts1</i></p> <p>Where:</p> <ul style="list-style-type: none"> ◦ <i>URL_OF_RationalTest_Server</i> is the URL of Rational® Test Automation Server. ◦ <i>test</i> is the name of the project in Rational® Test Automation Server. ◦ <i>ts1</i> is the name of the team space. <p> Notes:</p> <ul style="list-style-type: none"> ◦ If you have a project with the same name in different team spaces, then you must append either the &team.space.name=name_of_the_team.space or &team.space.alias=name_of_the_team.space_alias options. ◦ If the name of the project or team space contains a special character, then you must replace it with <code>%<Hexvalue_of_special_character></code>. <p>For example, if the name of the team space is <i>Initial Team Space</i>, then you must provide it as <i>Initial%20Team%20Space</i>.</p> <p>Where <code>%20</code> is the Hex value of <code>Space</code> character.</p> <ul style="list-style-type: none"> ◦ The values provided in the Publish field always take precedence over the Results options set in the product preferences (Window > Preferences > Test > Rational Test Automation Server > Results). <p>The Reports information section on the IBM Rational Performance Tester Task pane displays the names of the report along with its corresponding URLs in the following conditions:</p>

Fields	Description
	<ul style="list-style-type: none"> ◦ When you configured the URL of in Preferences of Rational® Performance Tester (Window > Preferences > Test > Rational Test Automation Server). ◦ When you set Publish result after execution as Always or Prompt in the Preferences of Rational® Performance Tester (Window > Preferences > Test > Rational Test Automation Server > Results). ◦ When you used the Publish field or the Publish field along with the Publishreports field.
Publish_for	<p>Use this field to publish test results to Rational® Test Automation Server from Azure DevOps based on the completion status of the tests.</p> <p>You must use the Publish_for field along with the Publish field. You can add multiple options separated by a comma. The following are the available options that you can use for the Publish_for field:</p> <ul style="list-style-type: none"> ◦ ALL: You can use this option irrespective of status of the test. ◦ PASS: You can use this option to publish test results for the tests that have passed. ◦ FAIL: You can use this option to publish test results for the tests that have failed. ◦ ERROR: You can use this option to publish test results for the tests that included errors. ◦ INCONCLUSIVE: You can use this option to publish test results for the inconclusive tests. <p>For example, <i>FAIL,ERROR</i></p>
Publishreports	<p>Use this field to publish specific test results to Rational® Test Automation Server.</p> <p>The options that you can use with Publishreports are as follows:</p> <ul style="list-style-type: none"> ◦ STATS ◦ TESTLOG <p>For example, <i>STATS</i></p> <p>You must use the Publishreports field along with the Publish field. You can prefix the value of Publishreports with ! to publish the reports except the specified one.</p> <p>For example, <i>!STATS</i></p>

Fields	Description
Rate	<p>Use this field to change the rate of the rate runner group.</p> <p>For example, <i>Rate Runner Group1=1/s, 3/m</i></p> <p>Where, <i>Rate Runner Group1</i> is the name of the Rate Runner group that has two stages. The desired rate for the first stage is one iteration per second and the rate for the second stage is three iterations per minute.</p> <p> Notes:</p> <ul style="list-style-type: none"> ◦ The name of the Rate Runner group must match with the name in the rate schedule. ◦ The Rate field creates a new copy of the rate schedule that contains the specified number of rate.
Resource Monitoring Labels Override	<p>Use this field to perform any of the following actions:</p> <ul style="list-style-type: none"> ◦ Enable the Resource Monitoring from Service option for a performance <code>schedule</code> if the Resource Monitoring from Service option is not enabled from the schedule editor in Rational® Performance Tester. ◦ Ignore Resource Monitoring sources that were set in the performance schedule and change for a label matching mode. ◦ Replace an existing set of Resource Monitoring labels that were set in the performance schedule and run the schedule with a new set of Resource Monitoring labels. <p>If you have added a label in Rational® Test Automation Server for a Resource Monitoring source as <code>rm1</code>, then you can provide value as shown in the following example to collect data from the source:</p> <p><code>rm1</code></p> <p>If your label for resource monitoring contains a comma (,) then you must replace the single comma with the double comma while providing the value. For example, if you added a label to a Resource Monitoring source as <code>rm1,test</code>, then you must provide the value as shown in the following example to collect data from source:</p> <p><code>rm1,,test</code></p> <p> Notes:</p>

Fields	Description
	 <ul style="list-style-type: none"> ◦ You can use this field only when you want to run a Rate schedule or VU schedule. ◦ You can add multiple Resource Monitoring labels separated by a comma. ◦ You must add the Resource Monitoring labels to the Resource Monitoring sources on the Resource Monitoring page in the Rational® Test Automation Server project.
Results	<p>Use this field to provide a different name to the result file.</p> <p>The results file is stored in the Results directory. The default name of the result file is the name of the <code>test</code> or <code>schedule</code> with a timestamp appended.</p>
Test Case Name	<p>Use this field to enter the name of the test case in the Azure server.</p> <p>You must use Test Case Name with the following fields:</p> <ul style="list-style-type: none"> ◦ Azure DevOps Project URL ◦ Azure DevOps PAT ◦ Test Plan Name
Test Plan Name	<p>Use this field to enter the name of the test plan that is associated with the test case in the Azure server.</p> <p>You must use Test Plan Name with the following fields:</p> <ul style="list-style-type: none"> ◦ Azure DevOps Project URL ◦ Azure DevOps PAT ◦ Test Case Name
Usercomments	<p>Use this field to add text that you want to display in the user comments row of the report.</p> <p>For example, <i>test run with dataset</i></p>  <p>Note: When you run tests by using the double quotation marks ("") for the Usercomments field, then the user comments row of a report does not contain double quotation marks.</p> <p>For example, if you provide the value for the Usercomments field as <code>test run with "dataset"</code>, then the user comments row of a report displays the value as <code>test run with dataset</code>.</p>

Fields	Description
	 To work around this problem, you must create a command-line config file, and then run the test by using the Configfile field.
Users	<p>Use this field to override the default number of virtual users in the <code>test</code> or <code>schedule</code> run.</p> <p>For a <code>schedule</code>, the default is the number of users specified in the schedule editor and for a <code>test</code>, the default is one user.</p> <p> Note: The Users field creates a new copy of the <code>schedule</code> that contains the specified number of users.</p>
Var File	<p>Use this field to provide a complete path to an XML file that contains the variable name and value pairs.</p>
VM Arguments	<p>Use this field to specify the maximum heap size for the Java process.</p> <p>You can add multiple VM Arguments separated by a comma.</p>

 **Note:** The fields in the **Control Options** and **Output Variables** sections are from Azure DevOps that you can use to configure your test run. You can accept the default values or change the values based on your requirements.

For more information about **Control Options** and **Output Variables**, see the **Related information** section.

10. Perform any one of the following steps as per your requirement:

- a. Click **Save & queue** to save the configurations and queue the test run in the pipeline.

Result

The **Run pipeline** dialog is displayed.

- b. Click **Save**, and then provide a comment to save the configured settings for the task.

Result

The task is not queued for a run. You can save the task and run the job later. You can then click **Queue** to run the job at a later time.

11. Complete the following steps to run the test:

- a. Enter a comment for the test in the **Save comment** field.
- b. Select the agent that you configured for the test from the **Agent pool** list.
- c. Select the **Agent Specification** from the list.
- d. Select the **Branch/tag** from the list.
- e. **Optional:** Add the variables and demands for the task run from the **Advanced Options** pane, if required.
- f. Select the **Enable system diagnostics** checkbox for a detailed log view.
- g. Click **Save and run**.

Result

The test run begins as a pipeline job and displays the progress of the job run.

Results

You have run the tests by using the **IBM Rational Test Workbench** extension in the Azure DevOps pipeline.

What to do next

You can open the job to view the task logs from the **Jobs** section. You must click the **IBM Rational Performance Tester Task** to view a detailed log of the test run.

The **Reports information** section on the **IBM Rational Performance Tester Task** pane displays the names of the report along with its corresponding URLs if you publish test results to IBM® Rational® Test Automation Server.

If you provided the details in the **Azure DevOps URL**, **Test Plan Name**, **Test Case Name**, and **Azure DevOps PAT** fields, then you can view the results of the test run on your **Test Plan** dashboard.

Related information

[Creating a command-line config file on page 370](#)

[Viewing test logs in Jaeger on page 156](#)

[Viewing test logs on page 467](#)

[Generating server offline user tokens](#)

[Test results and reports in Rational® Test Automation Server](#)

[Specify conditions in Azure DevOps](#)

[Define variables in Azure DevOps](#)

Integration with Apache JMeter

You can use JMeter tests extension with IBM® Rational® Performance Tester to run JMeter tests.

In Rational® Performance Tester, you have the option to import JMeter tests, add tests to a schedule or compound test to run them. Additionally, JMeter test helps to simulate a heavy load on a server, group of servers, or to investigate overall sample response time under different load types.

JMeter samples are terminal elements in JMeter tests that inform JMeter to send requests to a server and wait for a response. When you run a JMeter test, a JMeter performance report is generated during a run and saved after a run. This report contains the data most significant to the run, shows the response trend of the lowest 25 samples in the test, and graphs the response trend of each sample for a specified interval.

With JMeter test, you can load and test the performance of an application that uses the following protocols:

- HTTP/HTTPS
- SOAP/REST
- FTP
- LDAP
- MOM
- SMTP/POP3/IMAP
- TCP

JMeter installation

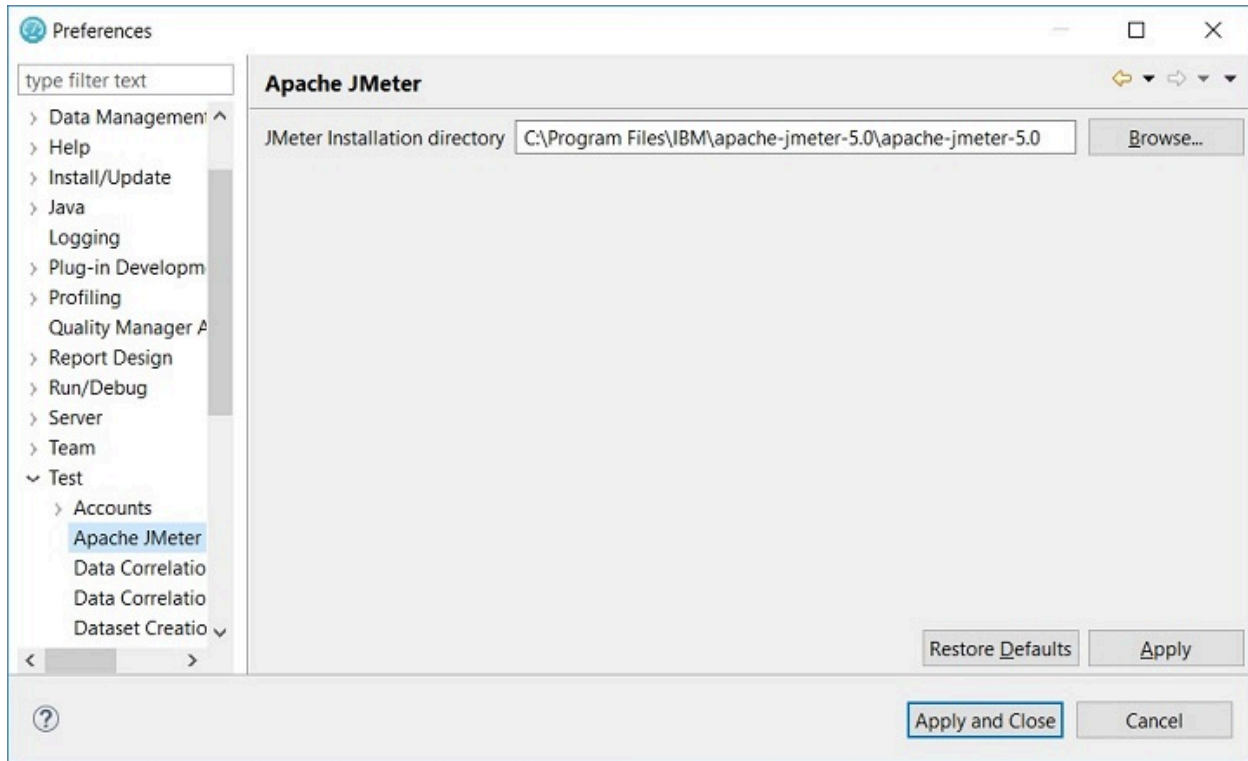
To work with JMeter tests, you must download the JMeter executable from https://jmeter.apache.org/download_jmeter.cgi and unzip it. To run the JMeter test as part of schedule, you must install JMeter on the remote agent machine and set the `JMETER_HOME` environment variable to the root installation folder. To run the JMeter tests, you must either specify the Apache JMeter path in the product **Preferences** or set the environment variable.



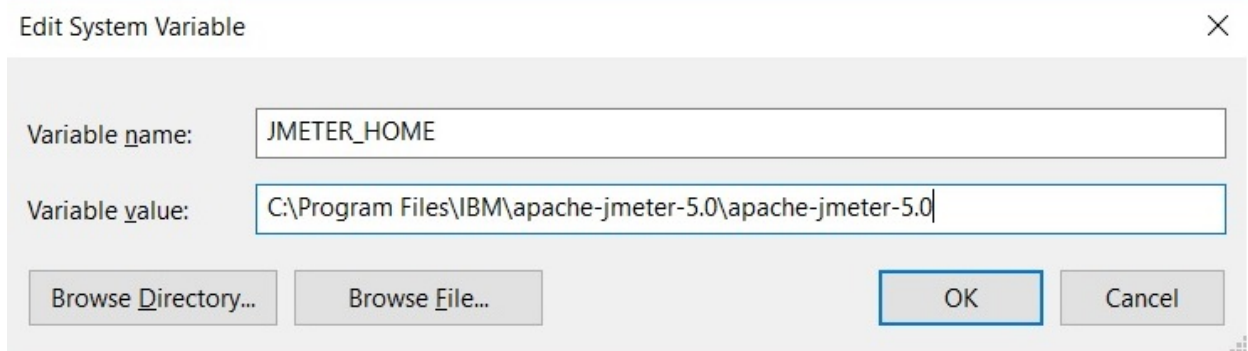
Note: When you run the compound test or schedule, an error message is displayed, if you have not specified the preferences or set the environment variable.

You can click **Window > Preferences > Test > Apache JMeter** to access the preference settings for Apache JMeter and point it to the `apache-jmeter-x.0` directory.

Where, x is the version number of the JMeter executable.



You can set the environment variable `JMETER_HOME` and point it to the `apache-jmeter-x.0` directory.



Importing a JMeter test to a project

You can import your JMeter tests to IBM® Rational® Performance Tester to run them as part of VU schedule.

About this task

If you have an existing JMeter test, you can import the test by dragging and dropping a JMeter file (JMX file) into the project in the test navigator. Alternatively, you can use the following procedure to import the tests.

1. In the Test Navigator, right-click and click **Import**.
2. In the Import dialog, expand **General** in the source list, select **File System** and then click **Next**.
3. Specify the directory where the JMeter test resides.

Click **Browse** to select a directory from where you can import the JMeter test. The JMeter test assets in the folder you selected are displayed.

4. Select the JMeter test you want to import.
5. Click **Browse** to choose the location to import JMeter test.
6. Click **Finish**. The imported JMeter test is displayed in the **JMeter Tests** folder.

Adding a JMeter test to an existing VU schedule

You can add a JMeter test to an existing VU schedule to test the performance of both static and dynamic resources and web applications.

About this task

When you add a JMeter test to a VU schedule, a user group with a loop is created and this loop contains the JMeter test invocation.

1. In the Test Navigator, browse to the schedule and double-click it. The schedule editor opens.
2. In the VU schedule editor, right-click the Schedule, and then click **Add > JMeter Test**.
3. In the Select Tests window, expand the project name and **JMeter Tests** folder and choose the test that you want to add.
4. Click **OK**. A new User Group with a loop that contains the JMeter test invocation is created.

What to do next

You must run the schedule or compound test, to view the statistics on the executed sequences. See [Running a local schedule or test on page 366](#) or Running compound tests.

Converting JMeter tests to VU schedule

You can convert a JMeter test to a VU schedule to load and test the performance of an application under test.

About this task

Rational® Performance Tester analyzes the selected JMeter test to add the number of users and loop iteration count in the VU schedule. When you convert the JMeter test to a VU schedule, the following events occur:

- The load information identified within the **Thread Group** nodes from the original JMeter test is examined to build a new VU schedule.
- The content of each **Thread Group** node is extracted from the original JMeter test and copied into a new JMeter test.

- The new JMeter test is then invoked by the VU schedule as an external test.



Note: The extracted JMeter test does not contain any load information such as the number of users and loop count, because the VU schedule manages all the information.



Remember:

- The content of the original **Thread Group** is not considered during the conversion process. Therefore, if there are any loops in the JMeter **Thread Group**, those cannot convert into a loop element in the new VU schedule.
 - If you have a JMeter test with more than one **Thread Group** node, each **Thread Group** is extracted to separate the JMeter test.
 - If you have a complex JMeter test, you must extract the functionalities that are included in the **Modules** or **Include controllers** into another JMeter tests. You must then add those JMeter tests to a VU schedule to run it.
1. Browse and select the JMeter test from the **Test Navigator**.
 2. Right-click the selected test, and then click **Convert to VU Schedule**.
 3. Verify that the name of the schedule is same as name of the JMeter test.
 4. Click **Finish**.

Result

The schedule editor opens.

What to do next

You must perform the following tasks:

- Run the converted test assets against successive builds of the application under test. See [Running a local schedule or test on page 366](#).
- Analyze the test results that are recorded. See [Running compound tests](#).

JMeter Performance report

The JMeter performance report summarizes the validity of the run, shows the average sample response time for the requests in the test, and graphs the sample response time of each sample for a specified interval.

Overall page

The Overall page provides a progress indicator that shows the status of the run and a bar chart that shows percentage of passed JMeter samples.

Summary page

The Summary page provides important information about the run. This page shows the following Run Summary information:

- The name of the test.
- The number of active users and the number of users who completed testing. This number is updated during the run.
- The elapsed time is the run duration, which is displayed in hours, minutes, and seconds.
- The status of the run. For example, the status can be Initializing Computers, Adding Users, Running, Transferring data to test log, Stopped, or Complete.

JMeter Samples page

JMeter samples are terminal elements in JMeter tests that informs JMeter to send requests to a server and wait for a response. The JMeter Samples page shows the average sample response time for all the requests in the test. The bar chart shows the average sample response time for all the requests. Each bar represents a sampler of the JMeter test. The corresponding table provides the following additional information:

- The minimum, average, and maximum duration for each sample in the run.
- The standard deviation of the duration.
- The completed sample rate and total number of completed samples per request.

JMeter Transaction page

The JMeter Transaction page shows the average transaction response time for all the requests in the test. The bar chart shows the average transaction response time for all the requests. Each bar represents a page that you visited during recording. The corresponding table provides the following additional information:

- The minimum, average, and maximum duration for each transaction in the run.
- The standard deviation of the duration.
- The completed transaction rate and total number of completed transaction per request.

Samples versus Time Summary page

The Samples versus Time Summary page shows the sample response trend as graphed for a specific interval. The Sample Response versus Time graph shows the sample response time for all the requests during the run. Each point on the graph is an average of what has occurred during that sample interval. The table after the graph lists the total average duration for all requests in the run and the standard deviation. To set the sample interval value, open the schedule, choose the **Statistics** tab from the drop-down menu, and then view or modify **Statistics sample interval**.

Samples versus Time Detail page

The Samples versus Time Detail page shows sample response trend for each of the request in the test. The line graph shows the average sample response time of each requests for a specific interval. The table after the graph provides the minimum, average, and maximum duration for the run and the standard deviation in the average sample response time.

Sample Throughput page

The Sample Throughput page summarizes the frequency of requests that are transferred per sample interval. The line graph on the left side shows the sample rate and passed sample rate per interval for all samples. The summary table after the graph lists the passed rate of total samples and counts for each passed samples. The line graph on the right side shows active users and the users who completed testing, per interval, over the course of a run. You can set the Statistics sample interval value in the schedule, as a schedule property. As the run nears completion, the number of active users decreases and the number of completed users increases. The summary table after the graph lists the active and completed users for the entire run.

To set the sample interval value, open the schedule, choose the **Statistics** tab from the drop-down menu, and then view or modify **Statistics sample interval**.

Server Throughput page

The Server Throughput page lists the rate and number of bytes that are transferred per interval and for the entire run. The page also lists the status of the virtual users for each interval and for the entire run. The line graph on the left side shows the rate of bytes sent and received per interval for all intervals in the run. The summary table after the graph lists the total number of bytes sent and received and bytes sent and received throughput rate for the entire run.

The line graph on the right side shows active users and users who are completed testing, per interval, over the course of a run. You set the Statistics sample interval value in the schedule, as a schedule property. As the run nears completion, the number of active users decreases and the number of completed users increases. The summary table after the graph lists the active and completed users for the entire run.

Server Health Summary page

The Server Health Summary page gives an overall indication of how well the server is responding to the load. The bar chart shows the total number of samples and total number of passed samples for the run. The table under the bar chart lists the same information.

Server Health Detail page

The Server Health Detail page provides specific details for 25 samples with the lowest success rates. The summary table lists the number of samples completed and passed in the run, and the passed sample percent and completion rate.

EGit integration

You can store your test assets in the remote Git repository and use EGit for version control operations. EGit is an Eclipse plugin for the Git version control system.

When you install Rational® Performance Tester, EGit is part of the product. To avoid the clutter, EGit is a separate Eclipse perspective in Rational® Performance Tester. For more information about EGit, refer to [EGit User Guide](#) documentation.

You can click **Windows > Perspective > Open Perspective > Other > Git** to open the Git perspective within Rational® Performance Tester.

Refer to the following topics to learn more about EGit integration with Rational® Performance Tester.


Importing test assets from the remote Git repository

You can import the test assets from the remote Git repository to Rational® Performance Tester and run it to test the performance of the application.

Before you begin

You must have completed the following tasks:

- Stored test assets of an application under test in the remote Git repository.
- Cloned the remote Git repository. For more information refer to the [EGit User Guide](#) documentation.


1. Open Rational® Performance Tester.
2. Click **File > Import > Git**, and then click the expand icon  .
3. Select **Projects from Git**, and then click **Next**.
4. Select the **Existing local repository** option, and then click **Next**.

Result

The list of all cloned remote Git repositories is displayed.

5. Select a remote Git repository from the list that has test assets, and then click **Next**.
6. Select one of the following options and perform the steps described in the table based on your requirement:

Options	Descriptions	Steps
Import existing Eclipse projects	Select this option to choose projects that you want to import from the remote Git	Perform the following steps:

Options	Descriptions	Steps
	repository to Rational® Performance Tester.  Note: If your project folder contains the <code>.project</code> file, only then the wizard displays the project that is available for the selection.	a. Select Import existing Eclipse projects , and then click Next . b. Select the checkbox to choose the projects from the list. c. Click Finish to import the projects. The imported project is displayed in the Test Navigator pane.
Import as general project	Select this option to choose a project that does not have a <code>.project</code> file in your project folder. When you select this option, a <code>.project</code> file is created within the project folder.	Perform the following steps: a. Click Import as general project , and then click Next . b. Enter a name for the project in the Project name field. c. Click Finish .

Results

You have imported test assets from the remote Git repository.

Pulling the changes from the remote Git repository

You must make your local repository up-to-date by pulling the latest changes in the remote Git repository before you commit and push your updated test assets to the Git repository.

Before you begin

You must have completed the following tasks:

- Cloned the remote Git repository. For more information refer to the [EGit User Guide](#) documentation.
- Ensured that any of your team members have updated the test assets and committed those files to the remote Git repository.

About this task

When you updated any of the test assets in your project, you might see merge conflicts errors while pulling the changes, if other members are also working on the same project.

1. Open Rational® Performance Tester.
2. Go to **Windows > Perspective > Open Perspective > Other**.

Result

The **Open Perspective** window is displayed.

3. Select the **Git** option from the list, and then click **Open**.

Result

The **Git perspective** view is displayed.

4. Right-click the remote Git repository from the list, and then select **Pull**.

Results

You have pulled the new changes from the remote Git repository to your existing local Git repository.

What to do next

You can publish your changes to the remote Git repository. See [Publishing test assets to the remote Git repository on page 106](#).

Publishing test assets to the remote Git repository

When you create or update any test assets in Rational® Performance Tester, you can publish your changes to the remote Git repository. Therefore, when you publish test assets, other members in the project can use your test assets in their test run, if required.

Before you begin

You must have completed the following tasks:

- Created or updated test assets in Rational® Performance Tester.
- Made the local repository up-to-date by pulling the changes from the remote Git repository. See [Pulling the changes from the remote Git repository on page 105](#).



About this task

You can either publish only the updated test asset files or entire project to the remote Git repository.

1. Open Rational® Performance Tester.
2. Locate the test assets files or projects that you updated from the **Test Navigator** pane.
3. Right-click the test assets or project, and then click **Team > Commit**.

Result

The **Git Staging** view is displayed.

4. Select the updated test asset files from the **Unstaged Changes** pane.
5. Click  or  to select a specific file or all the files from the **Unstaged Changes** pane to the **Staged Changes** pane.
6. Enter a message in the **Commit Message** pane.
7. Click **Commit and Push**.

Result


The **Push Result** dialog is displayed with the status of the commit to the remote Git repository.

Results

You have published test assets to the remote Git repository.

Techniques for troubleshooting issues in EGit

You might encounter issues while working with EGit in Rational® Performance Tester. You can find the information about certain issues and how to resolve them.

 **Warning:** When you initialize a new Git repository for a project in an Eclipse perspective, a `.gitignore` file is created in the project folder. While you commit the contents of a project to the Git repository, the `.gitignore` file ignores the report files. You can specify additional file types in the `.gitignore` file so that those file types are ignored when you commit the project contents to the Git repository.

The following topics provide more information about EGit integration issues, their causes, and the resolutions to fix the issues while you work with the Git perspective in Rational® Performance Tester.

Error: Missing library files in the project

After you pull a project in the Git perspective, `Project is missing the required Library` error is displayed in the **Verify Problems** tab. This error occurs due to the `.classpath` file which is specific to a workspace or system. The `.classpath` file needs to be pointed to the newly imported location.

You can perform the following steps to resolve the missing library issue:

1. Go to **Windows > Perspective > Open Perspective > Other**.
2. Select the **Java** perspective.
3. Expand the project from the **Test Navigator** pane.
4. Navigate to **Java Build Path > Libraries**.
5. Delete all the jar files that are missing after you pull the changes.



Note: The files are marked with a red cross.

6. Play back the project.

The required jar files are added to the project.

Git merge conflicts

Performance test projects contain certain metafiles and test assets in binary form. When you work in a collaborative test creation environment, you might encounter Git merge conflicts while you commit test assets to the remote Git repository.

When you view a project in the File Explorer, the folder structure is displayed as follows:

MyProj

```

.project
.classpath
.settings/
    assets.xml
    foo.prefs
    bar.prefs
    etc.prefs
META-INF/
    MANIFEST.MF
Tests/
    test1.testsuite
    test2.testsuite
Schedules/
    sched1.testsuite
    sched2.testsuite
Datasets/
    ds1.csv
    ds1.csv.metadata
Results/
CustomFolder/ (If custom folders are created for organizing assets)|

```

The metafiles such as `.project`, `.classpath`, `assets.xml`, and the test assets must be committed to the remote Git repository for seamless collaboration. Due to the specialized format of information in the metafiles and test assets being binary files, the Git merges cannot happen automatically. You must resolve the resulting merge conflicts manually, if any.

Best practices to minimize the merge conflicts

You must adhere to the following best practices to minimize the merge conflicts when you commit test assets to the remote Git repository:

- Refrain from working on the same test assets to the extent, if possible.
- Ensure that the team members coordinate and work closely with other members while making changes on the same test assets. Thereby, if there are any merge conflicts on binary files, those errors can be identified and addressed at the earliest.

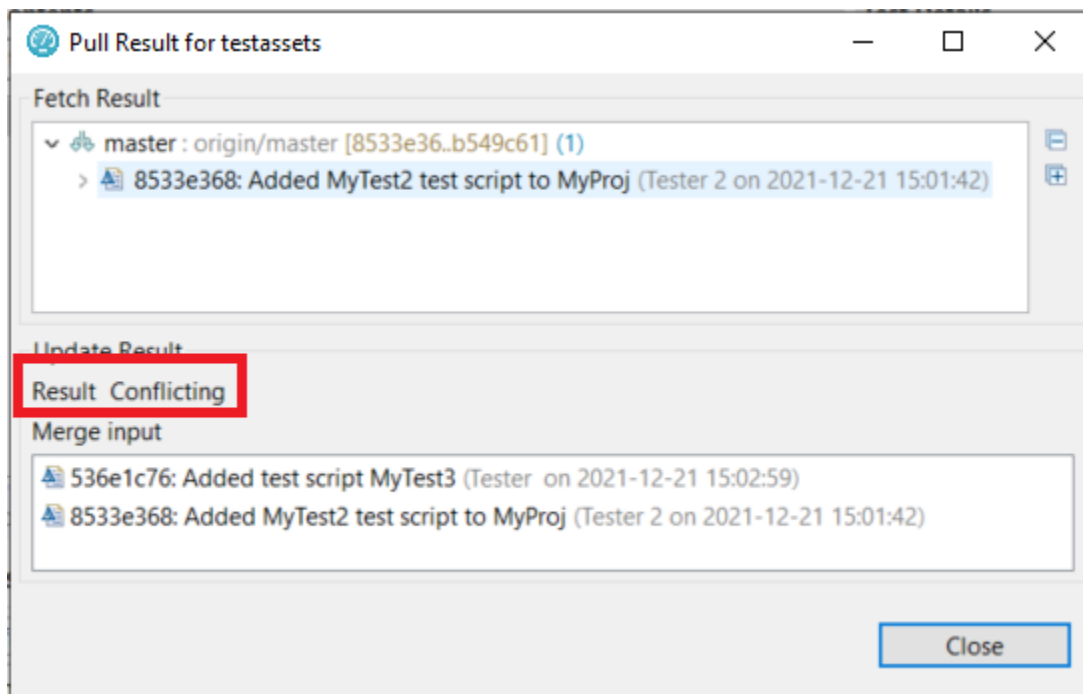
- Keep the local Git repository up-to-date with the remote upstream branch by pulling in changes periodically. The local Git branch must be made up-to-date with the master branch by re-basing frequently.
- Use the descriptive messages for your commits, so that other members can refer to those messages to resolve merge conflicts efficiently.

Conflict management: errors in assets.xml

In a collaborative work environment, the merge conflicts in `assets.xml` arise despite best efforts when you pull the changes from the remote upstream.

The project includes a metafile called `assets.xml` that contains a list of all test asset files in a project along with their dependencies in an XML format. The `assets.xml` file is automatically generated by Rational® Performance Tester from the contents of the project. The `assets.xml` file is refreshed each time you modify the files in the project. These changes in the `assets.xml` file are the most common source of merge conflicts.

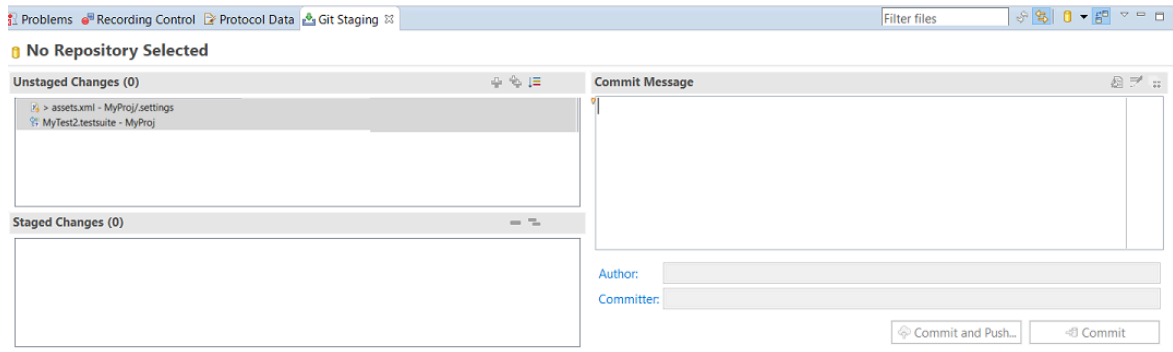
The following image displays the merge conflicts when you attempt to perform a pull request on the local Git repository by using the **Git** perspective:



You can perform the following steps to resolve the merge conflicts in the `assets.xml` file:

1. Click **Close** to dismiss the merge conflict result dialog.
2. Click **Window > Show View > Other**.
3. Expand **Git**, and then select **Git Staging**.
4. Click **Open**.

The conflicting files are displayed in the **Unstaged Changes** pane.

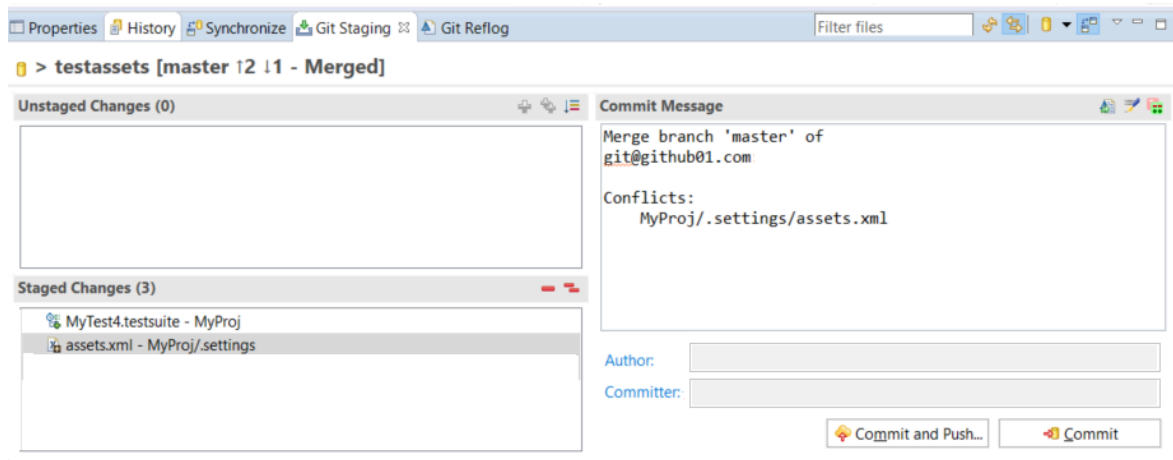


5. Right-click the project from the **Test Navigator** pane, and then click **Refresh**.

When you refresh the project, the `assets.xml` file is updated to include the local files in the project and also files that were pulled down from the remote upstream.

6. Go to the **Git Staging** view, and then move the files from **Unstaged Changes** to **Staged Changes** by using the

Add selected files to the index  or  icon.



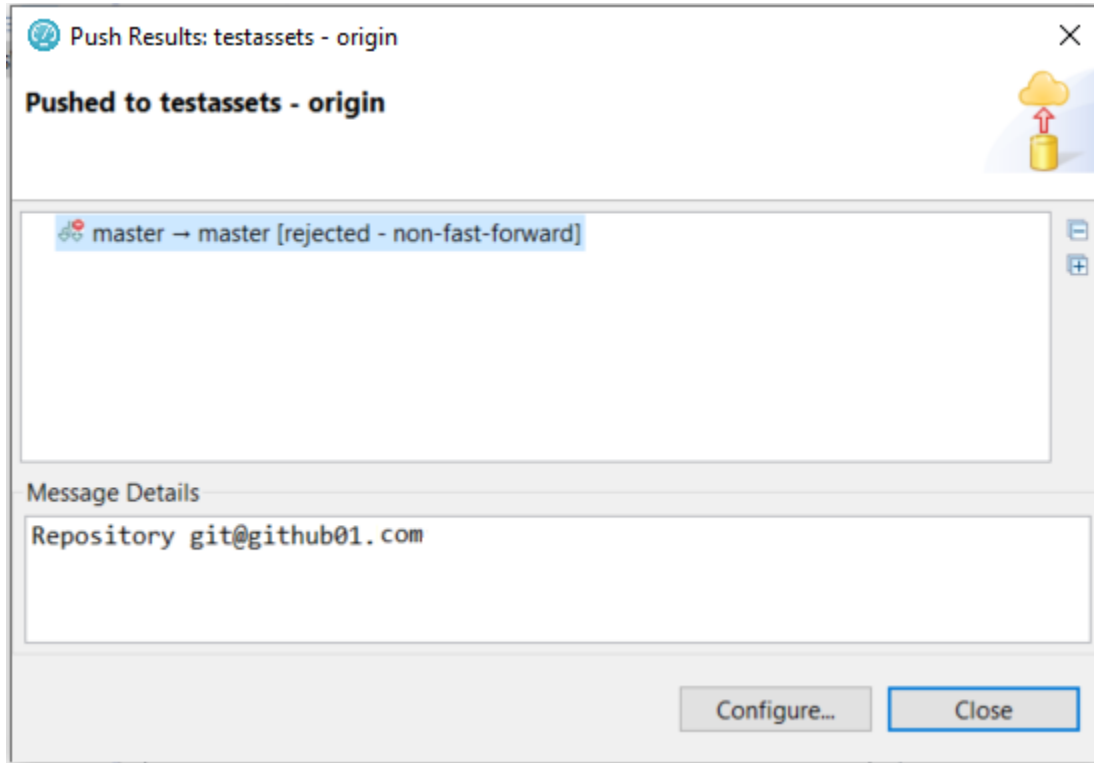
7. Enter a message in the **Commit Message** pane, and then click **Commit and Push**.

The **Push Result** dialog is displayed with the status of the commit to the remote Git repository.

Conflict management: errors in test assets

When multiple users edit and commit the same set of test assets, then while you pull or push changes to the remote Git repository, a merge conflict occurs.

Consider two users are working on the same VU Schedule called `MyVUSchedule1.testsuite`. One of the users committed and pushed changes to the remote Git repository. Later, when the other user tries to commit and push changes, the commit fails with the following error message:



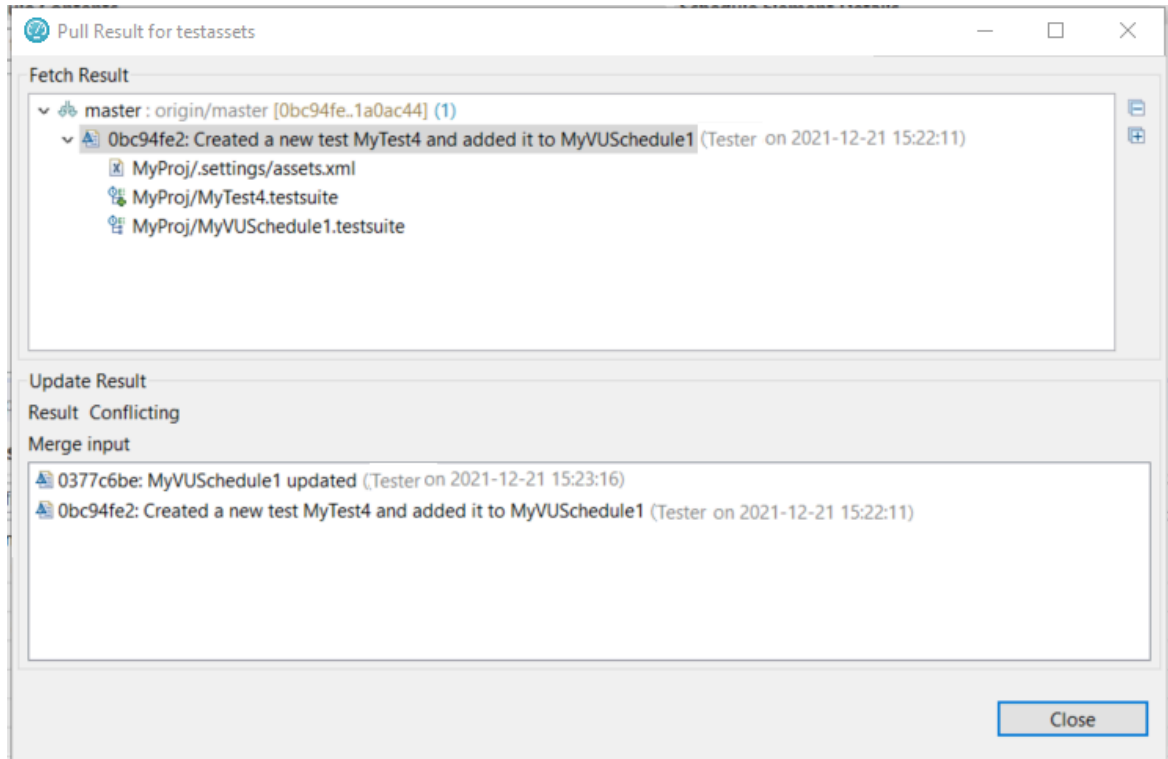
The error occurs because the remote Git repository contains changes pushed by the other user that are not present in the local branch. To push the changes, the changes in the remote Git repository must be pulled down to the local repository.


You can perform the following steps to resolve the merge conflicts in test assets:

1. Go to the **Git perspective** view, and then right-click the remote Git repository from the list.
2. Select **Pull** from the list to make your local Git repository up-to-date.

The **Pull Request** window is displayed.

3. Expand the commits fetched from the remote Git repository, and then expand each commit to view the list of changed files in them.



 **Note:** You must make a note of the test assets listed in the **Pull Request** window that were modified locally.

4. Examine the **Git Staging** view to find the conflicting or unmerged files.

 **Note:** The conflicting files are marked with a red icon.

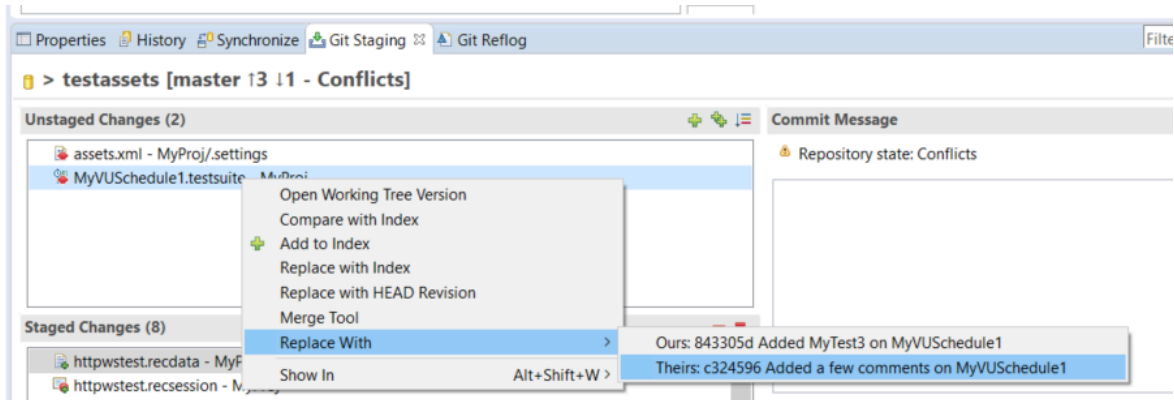
5. Right-click the conflicted test assets, and then select the **Replace With** option.



6. Select one of the following options based on your requirement:

- **Theirs:commitID commitmessage**
- **Ours:commitID commitmessage**

Since `MyVUSchedule1.testsuite` is binary, the remote and local changes must have to be merged manually. If necessary, you must first discuss with the user who authored or committed the changes. If their changes are extensive and the locally performed changes are relatively minor, you can overwrite the local version with the remote by selecting **Theirs:commit ID commit message**. You can then do the local changes again.

Alternatively, if the local changes are extensive when compared to the changes in the remote branch, then you can overwrite the remote version by selecting the **Ours:commit ID commit message** option. Later, you must manually redo the remote changes on the local copy.



7. Redo the changes that were broken because of the replace action.
8. Move the files from **Unstaged Changes** to **Staged Changes** by using the **Add selected files to the index**  or  icon.
9. Enter a message in the **Commit Message** pane, and then click **Commit and Push**.

The **Push Result** dialog is displayed with the status of the commit to the remote Git repository.

Integration with Engineering Test Management

You can integrate IBM® Engineering Test Management (formerly known as IBM® Rational® Quality Manager) with Rational® Performance Tester to initiate test runs from Engineering Test Management.

To run tests from Engineering Test Management, you must configure the default adapter that is installed when you install Rational® Performance Tester.

You can run the adapter in the following modes:

- GUI
- CLI
- Windows™ service

Engineering Test Management reports

When you run a test script from Engineering Test Management, the default report that is displayed during a test run is attached to the results of Engineering Test Management. You can customize the reports based on your requirements. See [Customizing reports on page 446](#).

If you use Engineering Test Management 4.0 or later, you can view and analyze the test reports in Engineering Test Management. You can analyze the test reports while the test is in running state and after the test run is complete. You can click the option from the **Execution Results** dialog to view the test reports in Engineering Test Management.



Note: To access reports from outside of Rational® Performance Tester, you must enable the remote access from Rational® Performance Tester. See [Access reports remotely on page 468](#).

The result completion state that is reported to Engineering Test Management reflects the overall verdict of the test log that is associated with the run. See [Logs overview on page 465](#). In many cases, a test might contain a failed verification point, but still is considered as passed. You can view the attached report in the execution result of Engineering Test Management, and then set the execution results status accordingly.

You can view the full run results from within Rational® Performance Tester by opening Rational® Performance Tester in the workspace that is configured to be used by the adapter.

If the adapter is running from the command line, you must stop the adapter before opening Rational® Performance Tester. When Rational® Performance Tester is opened, you can access the full test reporting and test log capabilities. The test results for the runs that are initiated from Engineering Test Management are under the Engineering Test Management **Results** page.

For Rational® Performance Tester schedules, the result completion state that is reported to Engineering Test Management is based on the overall **Requirements** status. Only performance requirements for the last user stage that is defined in the schedule are covered by the report. If no requirements are specified, the result completion state in Engineering Test Management is set to *inconclusive*. In this case, you can view the attached performance reports and manually set the completion state in Engineering Test Management. See [Defining requirements in schedules](#).

Known limitations

- You cannot run tests from Engineering Test Management with encrypted datasets. When using such datasets, a password prompt is not displayed in the adapter service or in the command-line interface. The use of encrypted datasets are not recommended in the GUI mode, because it requires user interaction with Rational® Performance Tester to initiate test runs from Engineering Test Management.
- You can start only one adapter per product installation on a given computer. If you use multiple adapters on the same computer, it requires you to install each product as its own software package in its own directory. If you want to run multiple adapters on the same computer, you must ensure that adapters are using different workspaces.

For information about using Engineering Test Management, refer to the [IBM Engineering Lifecycle Management](#) documentation.

Refer to the following topics to learn more about integrating Engineering Test Management with Rational® Performance Tester.

Configuring the Engineering Test Management adapter

You must configure the Engineering Test Management adapter to establish a successful connection between Rational® Performance Tester and Engineering Test Management.


Before you begin

You must have the following information:



- The URL of the Engineering Test Management server.
- A user credential and valid license to access Engineering Test Management.
- The user account must be added to the project area that is being accessed by the adapter with write permissions to the project.

For more information about Engineering Test Management, refer to the [IBM Engineering Lifecycle Management](#) documentation.

1. Open Rational® Performance Tester.
2. Click **Window > Preferences > Quality Manager Adapter**.
3. Enter the following information of the Engineering Test Management:

Fields	Actions
Server URL	<p>Enter the URL of Engineering Test Management.</p> <p>For example, <code>https://<hostname>:<portnumber>/qm</code></p> <p> Note: If you rename the Engineering Test Management server, you must perform the following tasks:</p> <ol style="list-style-type: none"> a. Update the Engineering Test Management server name in the hosts file with a new name. b. Update the Server URL field with the new name. c. Configure the adapter to point to the new URL.
Adapter name	<p>Enter a unique name to identify the Engineering Test Management adapter. The Engineering Test Management adapter uses the name of the computer as the default name of the adapter.</p>
Project area	<p>Enter the name of the project area in Engineering Test Management.</p>

4. Select one of the following **Authentication type** from the drop-down list to connect to Engineering Test Management:

Authentication type	Actions
Username and Password	<p>Perform the following steps:</p> <ol style="list-style-type: none"> Enter the username associated with Engineering Test Management in the User ID field. Enter the password associated with the username of Engineering Test Management in the Password field.
KERBEROS	<p>Click Browse to locate and select the <code>kerberos.ini</code> file in the Configuration File field.</p> <p> Note: The <code>kerberos.ini</code> file is automatically created when you set up Kerberos.</p> <p>For example, on Windows systems, you can locate the file in the <code>c:\windows\krb5.ini</code>. The file name and the location might change based on the operating systems.</p>
SSLCERT	<p>Perform the following steps:</p> <ol style="list-style-type: none"> Enter the location of the SSL certificate keystore in the Certificate Location field. Enter the keystore password in the Password field. <p> Note: The expected format of the keystore is p12. The keystore must contain the client certificate that the adapter uses when you authenticate with Engineering Test Management.</p>
SMARTCARD	Select a certificate from the drop-down list from the Certificate Selection field.

- Optional:** Select the **Enable Proxy** checkbox to connect through a proxy computer and perform the following steps to enter the **Proxy Details** of the computer:
 - Enter the hostname of the proxy computer in the **Host** field.
 - Enter the port number of the proxy computer in the **Port** field.
 - Enter the username and password of the proxy computer in the **User** and **Password** fields.
- Click **Apply and Close** to save and close the configuration.

Results

You have configured the details of Engineering Test Management on Rational® Performance Tester.

What to do next

Related information

[Connecting and disconnecting the Engineering Test Management adapter from the GUI mode on page 117](#)

[Starting and stopping the Engineering Test Management adapter from the command line on page 119](#)

[Importing test assets into Engineering Test Management on page 122](#)

Configuring the workspace directory of the adapter

You must configure the workspace directory of the adapter to start or stop the Engineering Test Management adapter from command-line interface.

About this task

If the **Use resources that are local to a test machine** option is set in Engineering Test Management, then the `WORKSPACE_DIR` must be set to the same workspace where your test assets are located.

1. Locate the `adapter.config` file in the `product_install_dir\RPT-RST_RQMAadapter\config\` directory.

Where `product_install_dir` is the directory where Rational® Performance Tester is installed.

For example, `C:\Program Files\IBM\SDP`.

2. Edit the `WORKSPACE_DIR` variable in the `adapter.config` file to point to the same test workspace that you want the adapter to use.

For example, `WORKSPACE_DIR= C:\Users\username\IBM\rationalsdp\my_adapter_workspace`.

Results

You have configured the workspace directory of the adapter.

What to do next

You can start or stop the Engineering Test Management adapter from command-line interface.

Connecting and disconnecting the Engineering Test Management adapter from the GUI mode

You can use the **Quality Manager Adapter** view to connect, disconnect, and view adapter activities from Rational® Performance Tester.


Before you begin

You must have configured the Engineering Test Management adapter. See [Configuring the Engineering Test Management adapter on page 114](#).

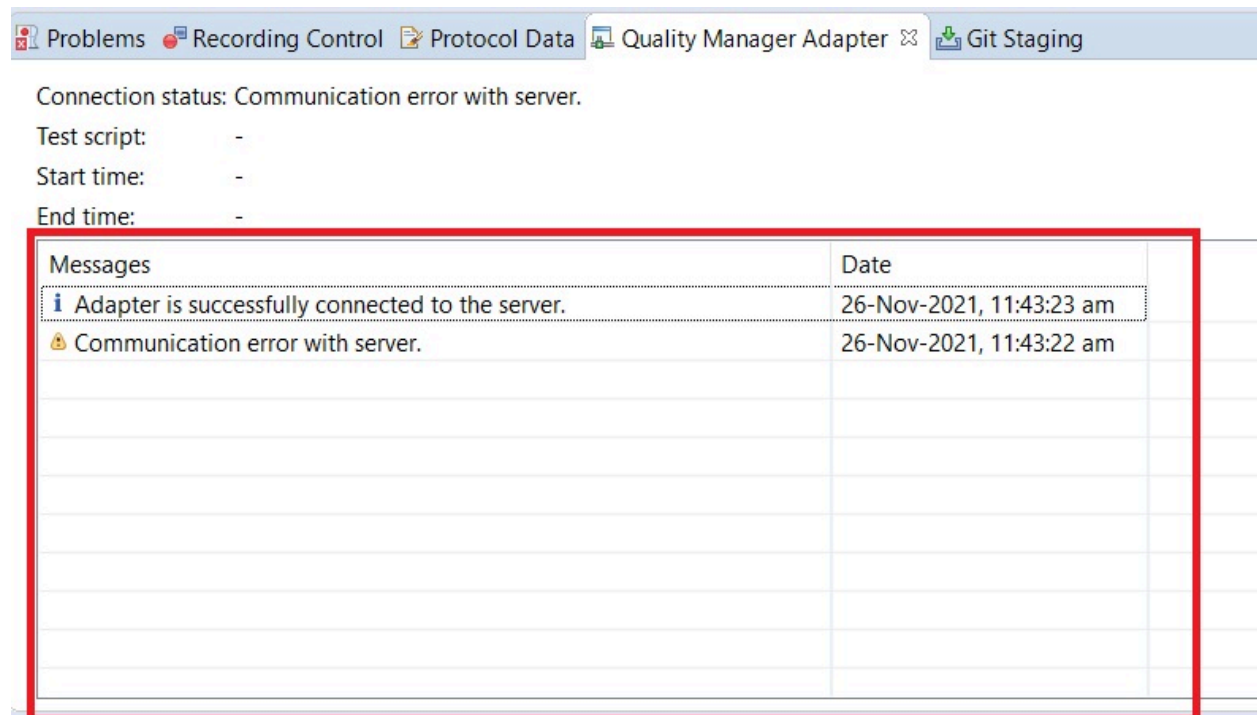
About this task

In the GUI mode, when a script is run from Engineering Test Management, you can see the test run in progress inside Rational® Performance Tester as though the test were run manually in Rational® Performance Tester.



Push buttons to connect and disconnect to the Engineering Test Management server are located in the upper-right corner of **Quality Manager Adapter** view. This view also has a local preferences menu that you can use to control some behavior of the GUI mode adapter. If you see errors or warnings, use the **Error Log** view for further investigation.

 **Note:** You must not use Rational® Performance Tester while the adapter is running. If you do so, you might interfere with the ability of the adapter to run test scripts. You must stop the adapter before you open Rational® Performance Tester.

The following image displays the activities of the adapter in the **Quality Manager Adapter** view:



1. Open Rational® Performance Tester.
2. Click **Window > Show View > Quality Manager Adapter**.
3. Perform the following actions either to connect or disconnect the adapter:

- Click the **Connect to RQM** icon  to connect the adapter.
- Click the **Disconnect from RQM** icon  to disconnect the adapter.

Results

You have connected or disconnected the Engineering Test Management adapter from Rational® Performance Tester.

Related information

[IBM Engineering Test Management overview on page 34](#)

[Configuring the Engineering Test Management adapter on page 114](#)

[Starting and stopping the Engineering Test Management adapter from the command line on page 119](#)

[Importing test assets into Engineering Test Management on page 122](#)

Starting and stopping the Engineering Test Management adapter from the command line

You can use the command-line interface to start, stop, and view activities of the Engineering Test Management adapter that you configured in Rational® Performance Tester.

Before you begin

You must have performed the following tasks:

- Configured the adapter in Rational® Performance Tester. See [Configuring the Engineering Test Management adapter on page 114](#).
- Configured the workspace directory of the adapter. See [Configuring the workspace directory of the adapter on page 117](#).

About this task

When you run test assets from the command-line interface, the adapter activities are printed to the `adapter.log` file that can be accessed from `product_install_dir\RPT-RST_RQMAadapter\logs`.

To print the current status of the adapter, you must navigate to the `product_install_dir\RPT-RST_RQMAadapter\bin` directory, and then you can run the `RQMAadapter.bat STATUS` command.

Where, `product_install_dir` is the installation directory of Rational® Performance Tester.



Warning: You must not use Rational® Performance Tester while the adapter is running. You must stop the adapter before you open Rational® Performance Tester for any reason.

1. Open a command-line interface.
2. Navigate to the `product_install_dir\RPT-RST_RQMAadapter\bin\` directory.
3. Perform the following step either to start or stop the adapter:
 - Run the following command to start the adapter from the command line:

Operating system	Command to be run
Windows™	RQMAadapter.bat START

Operating system	Command to be run
Linux™	RQMAdapter.sh START

- Run the following command to stop the adapter from the command line:

Operating system	Command to be run
Windows™	RQMAdapter.bat STOP
Linux™	RQMAdapter.sh STOP

Results

You have started or stopped the Engineering Test Management adapter from the command-line interface.

Related information

[Configuring the Engineering Test Management adapter on page 114](#)

[Connecting and disconnecting the Engineering Test Management adapter from the GUI mode on page 117](#)

[Importing test assets into Engineering Test Management on page 122](#)

Starting and stopping the Engineering Test Management adapter as a Windows™ service

You can use the Windows™ service to start, stop, and view adapter activities.




Before you begin

You must have completed the following tasks:

- Configured the adapter in . See [Configuring the Engineering Test Management adapter on page 114](#).
- Installed Microsoft™ .NET Framework 3.5.x in Windows systems for the adapter service.
- Configured the workspace directory of the adapter. See [Configuring the workspace directory of the adapter on page 117](#).

About this task

When you install Rational® Performance Tester, you must install the adapter as a Windows™ service. The default status of **Startup Type** is set to **Manual**.

 Human Interface Device Access	Enables gen...	Manual	Local System
 IBM RTW-RPT-RST adapter for RQM		Manual	Local System
 IBM Standard Asset Manager Service	Started	Automatic	Local System

Optionally, to configure the service to start automatically, you can right-click the adapter listing, and then select **Properties**. You can then select **Automatic** from the drop-down list in the **Startup Type** field. Therefore, the adapter can start automatically when you restart your computer and does not require you to log in.



Notes:

- When you start the adapter as a service, you cannot run the Web UI tests of IBM® Rational® Test Workbench from IBM® Engineering Test Management.
- You must not open in the same workspace while the adapter is running as a Windows™ service. You must stop the adapter before you open in the configured workspace.

When you run the adapter as a service, the status of the adapter is printed to the `adapter.log` file that can be accessed from `product_install_dir\RPT-RST_RQMAdapter\logs`.

You can also print the current status of the adapter by navigating to the `product_install_dir\RPT-RST_RQMAdapter\bin\` directory, and then run the `RQMAdapter.bat STATUS` command.

1. Open the Windows™ services.
2. Perform the following step either to start or stop the adapter:
 - Right-click **IBM RTW-RPT-RST adapter for RQM** and, then click **Start** to start the service.



Note: The adapter is same for Rational® Service Tester for SOA Quality, Rational® Test Workbench and Rational® Performance Tester.

- Right-click **IBM RTW-RPT-RST adapter for RQM** and, then click **Stop** to stop the service.

Results

You have started or stopped the Engineering Test Management adapter as a Windows service.

Related information

[IBM Engineering Test Management overview on page 34](#)

[Configuring the Engineering Test Management adapter on page 114](#)

[Connecting and disconnecting the Engineering Test Management adapter from the GUI mode on page 117](#)

[Starting and stopping the Engineering Test Management adapter from the command line on page 119](#)

[Importing test assets into Engineering Test Management on page 122](#)

Importing test assets into Engineering Test Management

You can import the tests into Engineering Test Management by using an adapter.

Before you begin

The adapter must be running on a computer where the test assets are located.

About this task

From Rational® Functional Tester, you cannot import AFT suites into Engineering Test Management because AFT suites are not checkmark_perf.jpg with the Engineering Test Management integration.

1. Log in to Engineering Test Management.
2. Click **Construction > Import Test Scripts**.
3. Select one of the following test scripts in the **Script Type** field:
 - a. **IBM Rational Performance Tester** to import a performance test or schedule from Rational® Performance Tester.
 - b. **Service Test** to import a service test from Rational® Service Tester for SOA Quality
 - c. **Rational Functional Tester** to import a functional test from Rational® Functional Tester.
 - d. **Rational Test Workbench** to import a Web UI test from Rational® Functional Tester or import a test from Rational® Test Workbench.
4. Select **Use test resources that are local to a test machine**, and click **Select Adapter**.
5. Select the computer on which the adapter is running, and click **Next**.
6. Enter the name of the project in the **Project Path** field, and then click **Go**.



Note: You must specify only the project name and not the entire path to the project.

7. Select the test assets that you want to import, and then click **Finish**.
8. Select those test assets to import again, and then click **Import**.

Results

You have imported the test assets to Engineering Test Management by using the adapter.

Related information

[IBM Engineering Test Management overview on page 34](#)

[Configuring the Engineering Test Management adapter on page 114](#)

[Connecting and disconnecting the Engineering Test Management adapter from the GUI mode on page 117](#)

[Starting and stopping the Engineering Test Management adapter from the command line on page 119](#)

[Testing shared assets with Engineering Test Management on page 123](#)

Testing shared assets with Engineering Test Management

You can make test projects and assets shareable in Engineering Test Management. By sharing assets, any computer with your product, that is connected to Engineering Test Management can execute a test or schedule.

Before you begin

When you are working with tests from a remote shared location, or Rational® Service Tester for SOA Quality uses a local workspace for the Engineering Test Management adapter. This adapter workspace is different from normal workspaces because the test assets are stored remotely. This means that every asset that is related to the test is downloaded from the shared location into the local workspace before execution. The following limitations apply:

- Assets in the adapter workspace might be deleted or overwritten with newer versions when updates are made to the shared location.
- If you change the shared location in the adapter workspace, the entire project is removed from the adapter workspace.
- Test results are stored in a different project, called `RQM_Results`, and are never deleted. The Engineering Test Management test result page links to the correct location.



Note: Do not edit test assets in the adapter workspace because you might lose your work. You must use these assets only for running tests.

If you are using source control and want to include only the minimum required assets, then include the following files:

- All `*.testsuite` tests files
- The `/src` directory if you use custom code
- All `*.dp` dataset files
- All `*.location` location files
- All digital certificates
- All WSDL and SOA security files



Note: All other assets, such as test results, are not required.

Custom code Java™ classes in the shared assets cannot use libraries that are outside the workspace. If your custom code must use such a library, then copy the library into the project, and update the classpath to use the local copy.

1. Create a shared directory on the computer that hosts the UNC file system that contains the test projects to share.

Example

For example, create a directory called: `C:\MyRemoteWorkspace`.

2. Copy the test projects to share into the shared directory.
If a project is stored in source control software, then copy it from there.
3. Check that the Engineering Test Management server can access the shared directory by using UNC paths.

Example

For example, the `\\MyServer\RPTRemoteAssets\` path must be mapped to the `C:\MyRemoteWorkspace` directory.

4. In Engineering Test Management, specify the directory that contains the actual test projects that are located in the shared directory.
5. Verify that you have correctly specified the UNC shared directory by browsing for the shared resource. Ensure that the first dialog contains the projects at the first level.
You must not have intermediate directories between the UNC shared directory and the project directory.

Related information

[Configuring the Engineering Test Management adapter on page 114](#)

[Importing test assets into Engineering Test Management on page 122](#)

Integration with IBM® Engineering Workflow Management


You can integrate Engineering Workflow Management (formerly known as Rational® Team Concert™) to create and track defects (bugs) or other work items, as a defect tracking tool in Rational® Performance Tester.

You can use Rational® Performance Tester to record and play back tests for the application that you develop and view their results. When you discover that you might want to raise defects, issues, or other types of work items for the test assets, you can create defects, issues, or other work items without the need to open Engineering Workflow Management.

For more information about Engineering Workflow Management, refer to the [IBM® Engineering Workflow Management](#) documentation.

The following table lists the tasks that you must perform to integrate Rational® Performance Tester with Engineering Workflow Management:

Tasks	Go to...
Install Rational® Performance Tester.	Installing Rational Performance Tester
Create any or all of the following types of test assets in Rational® Performance Tester to test your application: <ul style="list-style-type: none"> • Compound tests • Performance tests • Schedules (Rate or VU Schedules) 	Test Author Guide on page 204

Tasks	Go to...
Install Engineering Workflow Management and gain access to it.	IBM® Engineering Workflow Management documentation  Note: The System Requirements on page 26 provide more information about specific versions of Engineering Workflow Management requirements.
Run the test assets.	Running a local schedule or test on page 366
Configure the Engineering Workflow Management server URL in Rational® Performance Tester.	Configuring the URL of Engineering Workflow Management on page 125
Create defects from Rational® Performance Tester.	Creating defects in Engineering Workflow Management on page 126

Configuring the URL of Engineering Workflow Management

You must configure the URL of the Engineering Workflow Management server to use it as defect tracking tool in Rational® Performance Tester.

Before you begin

You must have the URL of the Engineering Workflow Management server.

About this task

Bugzilla is configured as the default defect tracking tool in the **Preferences** window of Rational® Performance Tester. If you are using Engineering Workflow Management to create and track defects (bugs), you can provide the URL of the Engineering Workflow Management server to search, submit, or add work items to test results from Rational® Performance Tester.

1. Open Rational® Performance Tester.
2. Click **Window > Preferences > Test > Test Log Editor**.
3. Enter the URL of the Engineering Workflow Management server in the following fields:

Fields	Format of the URL
Submit URL	<code>https://ewm.example.com:9443/ccm/web/projects/projectname#action=com.ibm.team.workitem.newWorkItem</code>
Search URL	<code>https://ewm.example.com:9443/ccm/web/projects/projectname#action=com.ibm.team.workitem.newWorkItem</code>

Fields	Format of the URL
Open URL	https://ewm.example.com:9443/ccm/web/projects/projectname#action=com.ibm.team.workitem.newWorkItem&id=

Where,

- ewm.example.com:9443 is the URL of the Engineering Workflow Management server.
- projectname is the name of the project in the Engineering Workflow Management server.



Note: You must update the URL, if there is a change in the name of the project in Engineering Workflow Management.

4. Click **Apply and Close** to save the configuration and close the **Preferences** window.

Results

You have configured the URL of Engineering Workflow Management in Rational® Performance Tester.

What to do next

You can create defects for the test results that are available in your project in Rational® Performance Tester. See [Creating defects in Engineering Workflow Management on page 126](#).

Creating defects in Engineering Workflow Management

You can create a defect in Engineering Workflow Management for the test result from the **Test Log** view in Rational® Performance Tester when the test results differ from the expected results. You can create a defect after the test run is complete.

Before you begin

- You must be familiar with working with Engineering Workflow Management.
- You must have performed the following tasks:
 - Gained access to the Engineering Workflow Management server.
 - Added user account to the project area with write permissions to the project.
 - Completed a test run. The test results must be available in Rational® Performance Tester.

1. Open Rational® Performance Tester.
2. Identify the test results from the **Test Navigator** pane for which you want to create a defect.
3. Right-click the test result, and then click **Display Test Log**.
4. Click the **Events** tab, and then click **User**.
5. Perform the following steps to create a defect in Engineering Workflow Management:

- a. Click **Submit** from the **Defects** section.

Result

The **Login** page is displayed only if you are not logged in to Engineering Workflow Management.

- b. Click **Work Items**, and then select the type of work item from the list.
- c. Enter the required information in the **Details**, **Description**, and **Discussion** sections.
- d. Click **Save**.

Results

You have created the defect for the test result in Engineering Workflow Management from Rational® Performance Tester.

What to do next

You can perform the following actions from the **Defect** section:

- Enter the defect number to associate the defect with the test result by clicking **Add**.
- Find the existing defects in Engineering Workflow Management by clicking **Search**.

Integration with IBM® Rational® Integration Tester

You can use Rational® Integration Tester extension to run integration tests from Rational® Performance Tester.

You can either import the projects from Rational® Integration Tester or manage them from Rational® Performance Tester by establishing the connection between the products. You can also create a compound test to run the tests by using the Agents.

Before you begin

You must have completed the following tasks:

- Installed Rational® Integration Tester Extension for Rational® Integration Tester.
- Installed Rational® Performance Tester Agent and Rational® Integration Tester Agent to run tests remotely. When you install Rational® Integration Tester Agent, you must have selected the **This Agent will only run probes** option.

- Defined the environment variable *INTEGRATION_TESTER_AGENT_HOME* on each location where the Agent is installed, and have pointed to the root installation directory of Rational® Integration Tester Agent.



Note: The environment variables *INTEGRATION_TESTER_HOME* and *INTEGRATION_TESTER_AGENT_HOME* are automatically created when you install Rational® Integration Tester 10.2.2 or later.

Setting environment variable

You must set the environment variable *INTEGRATION_TESTER_AGENT_HOME* and point it to the Rational® Integration Tester Agent installation directory if you are using Rational® Integration Tester 10.2.1 or earlier.

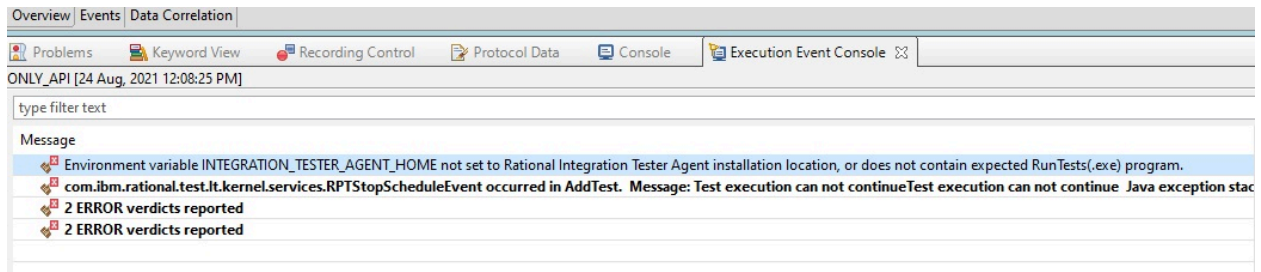
1. Run the following command to set the environment variable on Windows:

```
INTEGRATION_TESTER_AGENT_HOME = C:\Program Files\IBM\Agent
```

2. Run the following commands to set the environment variable on Linux:

```
INTEGRATION_TESTER_AGENT_HOME=/opt/IBM/RIT-Agent
export INTEGRATION_TESTER_AGENT_HOME
echo $INTEGRATION_TESTER_AGENT_HOME
```

If the environment variable is not set, the compound or schedule test run fails with an error message as depicted in the following image:

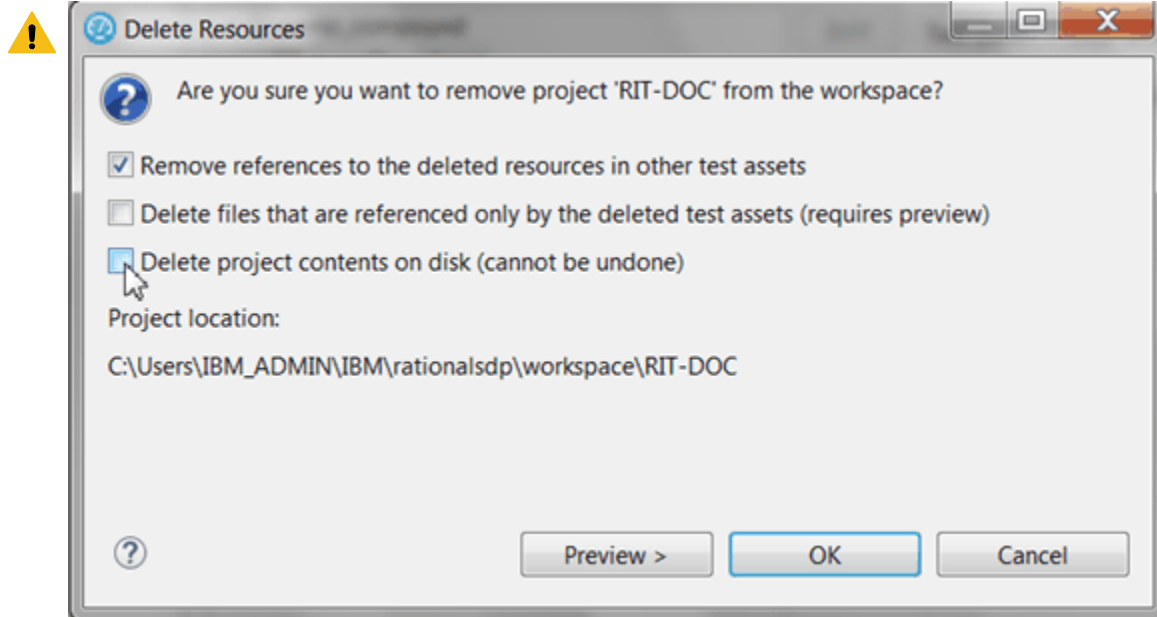


Connecting to an existing Integration project

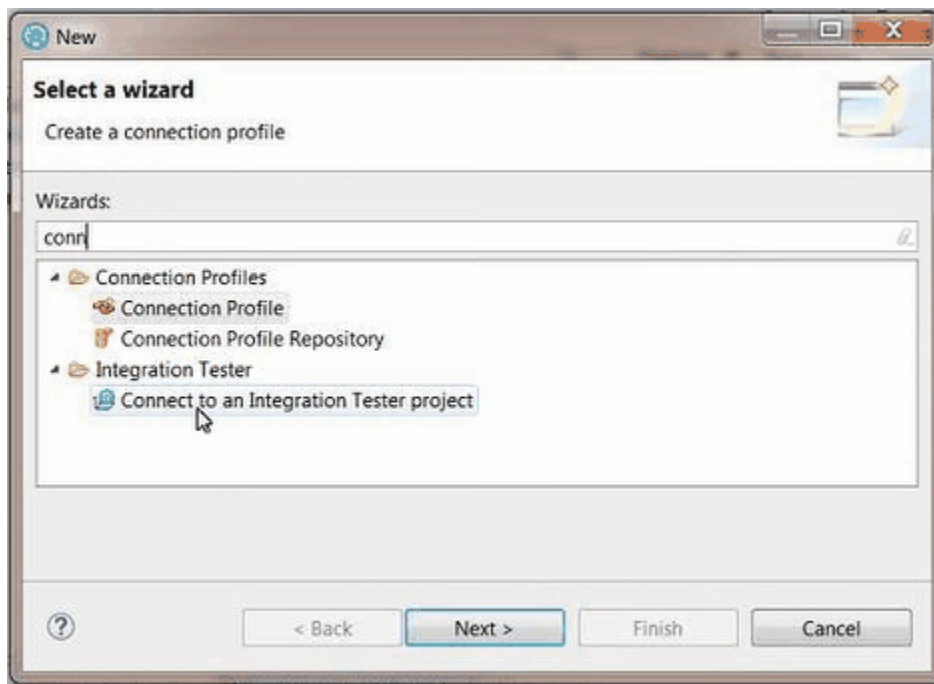
When you connect both the products any change or delete action made in one product workspace is reflected on the other product workspace, if both the products are installed on your machine.



Warning: If you delete a project from the Test Navigator, be sure that the option **Delete project contents on disk** is not selected in the **Delete Resources** dialog, otherwise the project would be deleted in Rational® Integration Tester if it is connected.



- In Rational® Performance Tester, right-click on the **Test Navigator**, select **New > Other > Rational Integration Tester > Connect to an Rational Integration Tester Project** and click **Next**.



- In the wizard page, click **Browse** and select the root folder that contains the project.

If the path contains a project, its name should automatically appear in **Project Name** and the **Finish** button should be enabled.

- In **When project is connected**, you have to perform one of the following actions:

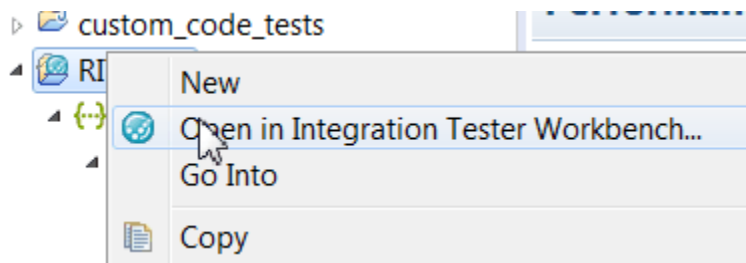
Setting Rational® Integration Tester preferences

To be able to open an Rational® Integration Tester project from Rational® Performance Tester Test Navigator, you need to have both the products installed on the same computer, and you must set the path to the execution file in the Preferences.

- In Rational® Performance Tester, click **Window > Preferences > Test > RIT Integration**.
- Click **Browse** and set the installation path to Rational® Integration Tester execution file. On Windows, the default location would be `C:\Program Files\IBM\IntegrationTester.exe`.
- Click **Apply** and **OK**.

Opening Rational® Integration Tester resources from the Test Navigator

- Once the preferences are set, you can open an Rational® Integration Tester project.
- In the **Test Navigator**, open the project root node and children nodes, and at any level, right-click and select **Open in Rational Integration Tester Workbench**.



If Rational® Integration Tester is automatically detected, the workspace opens for the selected resources.

If Rational® Integration Tester is not detected, a dialog opens on a Preference page where you need to verify the path to the execution file.

- **Warning:** Rational® Integration Tester cannot open more than one project at a time. If you have another project open, you will get an error. In that case, close Rational® Integration Tester and try to open the project again.

Importing Rational® Integration Tester project

If both the products are not installed on the same machine, you can import an Rational® Integration Tester project in your workspace. Another reason for the import is when you have Rational® Integration Tester installed but you do not want to connect to the Rational® Integration Tester project. In that case, the project is duplicated, any updates in one product workspace will not be reflected in the other product's workspace.

- To import an Rational® Integration Tester project:
- Right-click on the **Test Navigator**, choose **Import** and select **Existing project into workspace**.
- Choose **Select root directory** or **Select archive file**; select a project to import and click **Finish**.

The selected project appears in the **Test Navigator** and the compound test editor automatically opens.

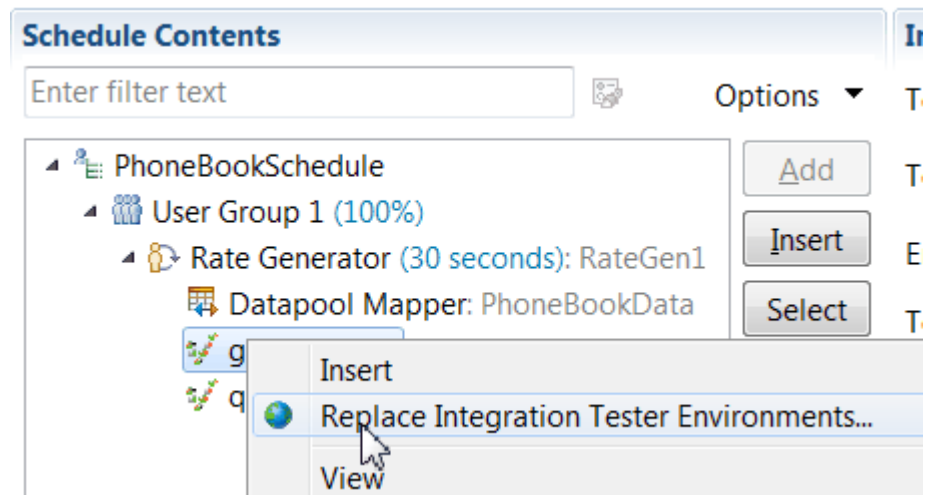
Modifying Rational® Integration Tester environments in

In the compound test, you can select Rational® Integration Tester tests and change the environment of each test. The environments are set in Rational® Integration Tester, you can only change the selection from the edited compound test.

- Open the compound editor and select a test.
- In the Rational® Integration Tester details, you can browse and change the properties of the selected test. The **Test path**, the **Environments** and **Description** are automatically updated accordingly.
- To select another environment for the Integration Tester test, use the dropdown menu.

Alternatively, you can change the environment selection for a test for a collection of tests:

- Right-click on the tree at any level under a node in the compound test and select **Replace Rational Integration**



Tester Environments.

- In the **Set Invocation Tester Environments** wizard, the first page displays the list of projects that use the selected environment and the number of tests from project that use this environment in the compound test.
- Select another used environment in the dropdown list. Click **Finish**. The new choice applies to the selected node and its children.

Next step is to create a compound test in Rational® Performance Tester to [run the Integration tests on page 132](#).

You can add a dataset mapper in the compound test or schedule for tests that are using multiple tags. See [Adding Dataset Mapper to map tags in the Rational® Integration Tester tests with the variable values of Rational® Performance Tester](#).

Running Rational® Integration Tester tests

You can use Rational® Performance Tester Extension for Rational® Integration Tester to run integration tests.

You also have the option to just import the projects to Rational® Performance Tester from Rational® Integration Tester, add the tests to a compound test to run them. You can either use Rational® Performance Tester Agent or Rational® Integration Tester Agent to generate the load. You need a compound test that contains the Rational® Integration Tester tests.

If you update integrationAPI tests in Rational® Performance Tester and want to apply the updates back to Rational® Integration Tester, you must install Rational® Integration Tester and define the path to its installation directory to set the connection.

You can find the following use cases to work with integration tests in Rational® Performance Tester:

- You have installed both the products and then you connect to the Integration project. Alternatively, you open the Integration resource directly from the **Test Navigator** view, and then work directly with the sources files.
- Rational® Integration Tester is not installed and you import the projects in the workspace of Rational® Performance Tester.



Note: The imported tests must be edited in Rational® Integration Tester. Similarly, the compounds tests must be edited in Rational® Performance Tester.

Running the compound test

Integration with IBM® UrbanCode™ Deploy


You can integrate IBM® Rational® Performance Tester with IBM® UrbanCode™ Deploy to run performance test assets as a **Process** from UrbanCode™ Deploy.

When you use UrbanCode™ Deploy for automating the deployments of your applications to an environment during the development cycle, you can create performance test assets for your application in Rational® Performance Tester. You can then run those tests from UrbanCode™ Deploy.

You must use the Rational® Performance Tester UCD plugin to integrate Rational® Performance Tester with UrbanCode™ Deploy. Integrating Rational® Performance Tester with UrbanCode™ Deploy automates the process of test runs. If you have many tests to run at regular intervals, you can use UrbanCode™ Deploy to initiate test runs automatically.

The following table lists the tasks that you must perform to run performance test assets from UrbanCode™ Deploy:

Tasks	Go to...
Install Rational® Performance Tester.	.

Tasks	Go to...
	<p>If you installed Rational® Performance Tester on the Mac operating system, you must have added <code>TEST_WORKBENCH_HOME</code> as an environment variable. You must have pointed that environment variable to the installation directory of Rational® Performance Tester.</p> <p> Note: On Windows® and Linux™ operating systems, the <code>TEST_WORKBENCH_HOME</code> environment variable is set after the installation of Rational® Performance Tester is complete.</p>
<p>Create any or all the following types of test assets in Rational® Performance Tester to test your application:</p> <ul style="list-style-type: none"> • Compound tests • Performance tests • Schedules (Rate or VU Schedules) 	<p>Test Author Guide on page 204.</p>
<p>Install UrbanCode™ Deploy with at least one agent.</p>	<p>IBM UrbanCode™ Deploy documentation.</p> <p>System requirements on page 27 for information about specific versions of UrbanCode™ Deploy requirements.</p>
<p>Install the IBM® Rational® Performance Tester UCD plugin on UrbanCode™ Deploy.</p>	<p>Installing the plugin on UrbanCode Deploy on page 134.</p>
<p>Create a Component in UrbanCode™ Deploy.</p>	<p>Creating a component in UrbanCode Deploy on page 135.</p>
<p>Create a Process for the component in UrbanCode™ Deploy.</p>	<p>Creating a process in UrbanCode Deploy on page 135.</p>
<p>Configure the Process that you created for the component in UrbanCode™ Deploy.</p>	<p>Configuring the process on page 136.</p>
<p>Create a Resource in UrbanCode™ Deploy.</p>	<p>Creating a resource in UrbanCode Deploy on page 149.</p>
<p>Configure the Resource.</p>	<p>Configuring the resource on page 149.</p>
<p>Create an Application in UrbanCode™ Deploy.</p>	<p>Creating an application in UrbanCode Deploy on page 151.</p>
<p>Configure the Application that you want to use to run performance test assets from UrbanCode™ Deploy.</p>	<p>Configuring the application on page 151.</p>

Tasks	Go to...
Run the test assets from UrbanCode™ Deploy.	Running tests from UrbanCode Deploy on page 153.

Installing the plugin on UrbanCode™ Deploy

You must install the Rational® Performance Tester UCD plugin to run performance test assets from UrbanCode™ Deploy.

Before you begin

- You must be familiar with working with UrbanCode™ Deploy.
- You must have performed the following tasks:
 - Installed UrbanCode™ Deploy. For more information about the installation, refer to the [IBM Urban Code documentation](#).
 - Been granted access to UrbanCode™ Deploy.
 - Downloaded the Rational® Performance Tester UCD plugin from the [IBM WebSphere, Liberty & DevOps Community portal](#).

For more information about specific versions of plugin, see [Integration plugin compatibility matrix on page 68](#).

1. Log in to UrbanCode™ Deploy, if you are not already logged in.

Result

The UrbanCode™ Deploy dashboard is displayed.

2. Click **Settings**.
3. Click **Automation Plugins** from the **Automation** pane.
4. Click **Load Plugin**.
5. Click **Choose File** to locate and open the compressed Rational® Performance Tester UCD plugin file.



Remember: You must not extract the Rational® Performance Tester UCD plugin compressed file contents.

6. Click **Submit**.

Result

The installed plugin is displayed on the **Automation Plugins** tab.

Results

You have installed the Rational® Performance Tester UCD plugin on UrbanCode™ Deploy.

What to do next

You must create a **Component** in UrbanCode™ Deploy. See [Creating a component in UrbanCode Deploy on page 135](#).

Creating a component in UrbanCode™ Deploy

You must create a component to include artifacts and processes. The artifacts include runnable files, images, databases, configuration instructions. Whereas the processes define the activities that components can perform.

Before you begin

- You must be familiar with working with UrbanCode™ Deploy.
- You must have been granted access to UrbanCode™ Deploy.

1. Log in to UrbanCode™ Deploy, if you are not already logged in.

Result

The UrbanCode™ Deploy dashboard is displayed.

2. Click **Components**, and then click **Create Component**.
3. Enter a name for the component in the **Name** field.
4. Enter the details in the other optional fields based on your requirement, and then click **Save**.

Result

The component that you created is displayed.

Results

You have created the component in UrbanCode™ Deploy.

What to do next

You must create a process for the component in UrbanCode™ Deploy. See [Creating a process in UrbanCode Deploy on page 135](#).

Related information

[IBM UrbanCode Deploy Documentation](#)

Creating a process in UrbanCode™ Deploy

You must create a process for the component to include step properties for the test that you want to run from UrbanCode™ Deploy.

Before you begin

- You must be familiar with working with UrbanCode™ Deploy.
- You must have performed the following tasks:

- Been granted access to UrbanCode™ Deploy.
- Created a component in UrbanCode™ Deploy. See [Creating a component in UrbanCode Deploy on page 135](#).

1. Log in to UrbanCode™ Deploy, if you are not already logged in.

Result

The UrbanCode™ Deploy dashboard is displayed.

2. Click **Components**.

Result

A list of components that are available in UrbanCode™ Deploy is displayed.

3. Select the component from the list for which you want to create a process.
4. Click the **Processes** tab, and then click **Create Process**.

Result

The **Create Process** dialog is displayed.

5. Enter a name for the process in the **Name** field.
6. Select **Operational (No Version Needed)** from **Process Type** drop-down list.
7. Verify the **Default Working Directory** field.

The **Default Working Directory** field defines the location that the agent uses to run the process. The default value is `${p:resource/work.dir}/${p:component.name}`.

Where `${p:resource/work.dir}` is the default working directory for the agent and `${p:component.name}` is the name of the component.

8. Click **Save**.

Result

The process that you created is listed in the **Processes** tab and the **Design** tab for the process is displayed.

Results

You have created the process for the component in UrbanCode™ Deploy.

What to do next

You must configure the process in UrbanCode™ Deploy. See [Configuring the process on page 136](#).

Related information

[IBM UrbanCode Deploy Documentation](#)

Configuring the process

You must configure the process that you created for the component to organize the steps in the process, specify the properties of the steps, and connect them.

Before you begin

- You must be familiar with working with UrbanCode™ Deploy.
- You must have performed the following tasks:
 - Been granted access to UrbanCode™ Deploy.
 - Created a component in UrbanCode™ Deploy. See [Creating a component in UrbanCode Deploy on page 135](#).
 - Created a process for the component in UrbanCode™ Deploy. See [Creating a process in UrbanCode Deploy on page 135](#).

About this task

When you open any process to configure, the process is displayed in the process editor. The process editor lists the plugins and steps. The required **Start** and **Finish** steps represent the beginning and the end of the process and are automatically placed on the design area.

You must provide the values for certain fields in the properties for the selected test step to run tests from UrbanCode™ Deploy. The minimum fields that you need to configure are **Name**, **Workspace**, **Project**, and **Test Suite Name** or **Config File**. The remaining fields are optional.

For example, if you provide a complete path of a configuration file that contains the parameters for a `test` or `schedule` run in the **Config File** field, then you need not provide values for the **Workspace**, **Project**, and **Test Suite Name** fields.



Note: If Rational® Performance Tester is not installed in default location, then you must provide the value for the **IMShared Location** field along with other mandatory fields.

1. Log in to UrbanCode™ Deploy, if you are not already logged in.

Result

The UrbanCode™ Deploy dashboard is displayed.

2. Click **Components**.

Result

A list of components that are available in UrbanCode™ Deploy is displayed.

3. Select the component from the list in which you created the process.

4. Click the **Processes** tab.

Result

A list of processes that are available for the component is displayed.

5. Select the process from the list that you want to configure.

Result

The **Design** tab for the process is displayed.

6. Click **IBM Rational Test Workbench**, and then **IBM Rational Performance Tester** from the left menu.

7. Drag the **Run an IBM Rational Performance Tester test** step, and then drop it into the design area.



Note: The selected test must be placed between **Start** and **Finish** steps.

8. Specify the properties for the selected test by performing the following steps:


a. Click the **Edit** icon.

Result

The **Edit Properties for Run an IBM Rational Performance Tester test** dialog is displayed.



b. Specify the properties for the selected test step by referring to the following table:



The following table lists the required fields that you must provide to run the test from UCD:


Fields	Description
Config File	<p>Enter the complete path to a file that contains the parameters for a <code>test</code> or <code>schedule run</code>.</p> <p>For example, <code>C:\Users\IBM\rationalmdp\workspace\Project1\config.txt</code></p> <p>For more information about creating a command line config file from Rational® Performance Tester, see the Related information section.</p> <p> Remember: If you provide a complete path of a configuration file in the Config File field, then you need not provide values for the Workspace, Project, and Test Suite Name fields.</p>
Name	<p>Enter the name for the test step.</p> <p>For example, <code>testrun1</code></p>
Project	<p>Enter the name of the project that has performance test assets.</p> <p>For example, <code>testproj1</code></p>
Test Suite Name	<p>Enter the file name of the performance test assets relative to the project.</p> <p>For example, <code>test1.testsuite</code></p>
Workspace	<p>Enter the complete path of the Eclipse workspace.</p> <p>For example, <code>C:/Users/IBM/rationalmdp/workspace1</code></p>




The following table lists the optional fields that you can provide to run the test from UCD:


Fields	Description
Custom Report Format Files	<p>Use this field to specify IDs of reports that you want to export in place of the default reports.</p> <p>You can provide multiple report IDs separated by a comma. You can navigate to Preferences of Rational® Performance Tester (Window > Preferences > Test > Performance Test Reports > Export Reports), and then select Show Report IDs checkbox to view the report IDs.</p> <p>You must use the Custom Report Format Files field along with the Exported Statistical Report Data File or Exportstatshtml field.</p> <p>For example, you can provide <code>http</code> as a value in the Custom Report Format Files field to export a Performance Report.</p>
Dataset Override	<p>Use this field to replace the dataset values during a <code>test</code> or <code>schedule</code> run.</p> <p>You must ensure that both original and new datasets are in the same workspace and have the same column names. When you enter a value for the Dataset Override field, you must also include the path to the dataset. You must provide the values for the Dataset Override field in the following format:</p> <pre data-bbox="574 1077 1377 1142">/project_name/ds_path/original_ds.csv:/project_name/ds_path/new_ds.csv</pre> <p>For example, <code>/proj1/Datasets/ds1.csv:/proj1023/Datasets/new_ds1.csv</code></p> <p>You can override multiple datasets that are saved in a different project by adding multiple paths to the dataset separated by a semicolon. For a <code>test</code> or <code>schedule</code>, the default value is the dataset specified in the test editor or schedule editor.</p>
Duration	<p>Use this field to change the duration of the stages in the rate schedule.</p> <p>For example, <code>Stage1=10s, Stage2=3m</code></p> <p>The stage number specified must exist in the rate schedule.</p>


Fields	Description
	 Note: The Duration field creates a new copy of the rate schedule that contains the specified number of duration.
Exported HTTP Test log file	Use this field to specify the path of the file directory to store the exported HTTP test log in a <code>.txt</code> format. For example, <code>C:/Users/Documents/tests/logexp.txt</code>
Exported Statistical Report Data File	Use this field to provide the complete path to a directory that you can use to store exported reports in a comma-separated values (CSV) format. For example, <code>C:/Users/Documents/tests</code>  Note: If you do not specify a value for the Custom Report Format Files field, then the reports that you selected in Preferences of Rational® Performance Tester (Window > Preferences > Test > Performance Test Reports > Export Reports) are exported.
Exportstatsformat	Use this field to specify a format for the report that you want to export. You must use at least one of the following options along with the Exported Statistical Report Data File field: <ul style="list-style-type: none"> ▪ <code>simple.csv</code> ▪ <code>full.csv</code> ▪ <code>simple.json</code> ▪ <code>full.json</code> ▪ <code>csv</code> ▪ <code>json</code> For example, <code>json</code> . You can add multiple formats for the report separated by a comma. When you want to export both simple and full type of report in a <code>json</code> or <code>csv</code> format, you can specify <code>json</code> or <code>csv</code> as the format in the field.


Fields	Description
	<p>The reports are saved to the location specified in the Exported Statistical Report Data File field.</p> <p> Note: The values provided in the Exportstatsformat field always take precedence over the Export Reports options set in the Preferences of Rational® Performance Tester (Window > Preferences > Test > Performance Test Reports > Export Reports).</p>
Exportstatshtml	<p>Use this field to provide the complete path to a directory that you can use to export web analytic results.</p> <p>The results are exported to the specified directory. You can analyze the results on a web browser without using Rational® Performance Tester.</p> <p>For example, <i>C:/Users/Documents/Reports</i></p>
History	<p>Use this field when you want to view a record of all events that occurred during a test or schedule run.</p> <p>You can use any of the following options:</p> <ul style="list-style-type: none"> ▪ jaeger: To send test logs to the Jaeger UI during the <code>test</code> or <code>schedule</code> run. <p> Note: You must set the <code>JAEGER_AGENT_HOST</code> property as an environment variable by using the command line before you use the jaeger option to send test logs to the Jaeger UI.</p> <ul style="list-style-type: none"> ▪ testlog: To send test logs as traditional test logs in Rational® Performance Tester during the <code>test</code> or <code>schedule</code> run. ▪ null: To send no test logs either to the Jaeger UI or Rational® Performance Tester during the <code>test</code> or <code>schedule</code> run. <p>For example, <i>jaeger</i></p> <p>You can add multiple options separated by a comma to send test logs during the <code>test</code> or <code>schedule</code> run to Rational® Performance Tester and the Jaeger UI.</p> <p>For example, <i>jaeger,testlog</i></p>



Fields	Description								
	For more information about how to view test logs in the Jaeger UI and Rational® Performance Tester, see the Related information section.								
IMShared Location	<p>Enter the complete path to the IMShared directory, if it is not in the default location.</p> <p>For example, <i>D:\Testtool\IBM\IBMIMShared</i></p> <p>The default location of the IMShared directory is as follows:</p> <table border="1" data-bbox="574 621 1417 846"> <thead> <tr> <th data-bbox="574 621 857 676">Operating system</th> <th data-bbox="857 621 1417 676">The default path to the directory</th> </tr> </thead> <tbody> <tr> <td data-bbox="574 676 857 737">Windows®</td> <td data-bbox="857 676 1417 737">C:\Program Files\IBM\IBMIMShared</td> </tr> <tr> <td data-bbox="574 737 857 791">Linux™</td> <td data-bbox="857 737 1417 791">/opt/IBM/IBMIMShared</td> </tr> <tr> <td data-bbox="574 791 857 846">Mac</td> <td data-bbox="857 791 1417 846">/Application/IBM/IBMIMShared</td> </tr> </tbody> </table>	Operating system	The default path to the directory	Windows®	C:\Program Files\IBM\IBMIMShared	Linux™	/opt/IBM/IBMIMShared	Mac	/Application/IBM/IBMIMShared
Operating system	The default path to the directory								
Windows®	C:\Program Files\IBM\IBMIMShared								
Linux™	/opt/IBM/IBMIMShared								
Mac	/Application/IBM/IBMIMShared								
Labels	<p>Use this field to add labels to test results when the test run is complete.</p> <p>For example, <i>label1, label2</i></p> <p>You can add multiple labels to a test result separated by a comma. When you run test assets, then the labels that you added are displayed on the Performance Report in Rational® Performance Tester.</p> <p>The Results page of Rational® Test Automation Server displays the label that you added in the Labels field for the specific test asset in the following conditions:</p> <ul style="list-style-type: none"> ▪ When you set Publish result after execution as Always or Prompt in the Preferences of Rational® Performance Tester (Windows > Preferences > Test > Rational Test Automation Server). ▪ When you use the Labels field along with the Publish field. <p> Note: When you run tests by using the double quotation marks ("") for the Labels field, then the labels in the test result do not include double quotation marks.</p> <p>For example, if you provide the value for the Labels field as "100" users, then the labels display 100 users in the test result.</p>								


Fields	Description
	 To work around this problem, you must create a command-line config file, and then run the test by using the Config File field.
Number of Virtual Users	<p>Use this field to override the default number of virtual users in the <code>test</code> or <code>schedule</code> run.</p> <p>For a <code>schedule</code>, the default is the number of users specified in the schedule editor and for a <code>test</code>, the default is one user.</p> <p> Note: The Number of Virtual Users field creates a new copy of the <code>schedule</code> that contains the specified number of users.</p>
Overwrite Results file	<p>Select or clear this field to determine whether a result file with the same name must be overwritten or not.</p> <p>The default status of the field is selected. Therefore, the file is overwritten and retains the same file name.</p>
Publish	<p>Use this field to publish test results to Rational® Test Automation Server from UrbanCode™ Deploy.</p> <p> Remember: Before you use the Publish field, you must provide the offline user token of Rational® Test Automation Server by using any of the following methods:</p> <ul style="list-style-type: none"> ▪ Set the value of the RTCP_OFFLINE_TOKEN environment variable to the offline user token of Rational® Test Automation Server by using the command-line interface. ▪ Provide the offline user token of Rational® Test Automation Server in the preferences of Rational® Performance Tester (Window > Preferences > Test > Rational Test Automation Server) <p>You must use one of the following values in the Publish field:</p>

Fields	Description
	<ul style="list-style-type: none"> ▪ <code>serverURL#project.name=name_of_the_project&team-space.name=name_of_the_teamspace</code> ▪ <code>serverURL#project.name=name_of_the_project&team-space.alias=name_of_the_teamspace_alias</code> ▪ no <p>The no option is useful if the product preferences are set to publish the results, but you do not want to publish them.</p> <p>For example, <code>URL_OF_RationalTest_Server#project.name=test&team-space.name=ts1</code></p> <p>Where:</p> <ul style="list-style-type: none"> ▪ <code>URL_OF_RationalTest_Server</code> is the URL of Rational® Test Automation Server. ▪ <code>test</code> is the name of the project in Rational® Test Automation Server. ▪ <code>ts1</code> is the name of the team space. <p> Notes:</p> <ul style="list-style-type: none"> ▪ If you have a project with the same name in different team spaces, then you must append either the &team-space.name=name_of_the_teamspace or &team-space.alias=name_of_the_teamspace_alias options. ▪ If the name of the project or team space contains a special character, then you must replace it with <code>%<Hexvalue_of_special_character></code>. <p>For example, if the name of the team space is <i>Initial Team Space</i>, then you must provide it as <i>Initial%20Team%20Space</i>.</p> <p>Where <code>%20</code> is the Hex value of <code>Space</code> character.</p>

Fields	Description
	 <ul style="list-style-type: none"> ▪ The values provided in the Publish field always take precedence over the Results options set in the product preferences (Window > Preferences > Test > Rational Test Automation Server > Results). <p>The Reports information section on the <code>LOG</code> file displays the names of the report along with its corresponding URLs in the following conditions:</p> <ul style="list-style-type: none"> ▪ When you configured the URL of in Preferences of Rational® Performance Tester (Window > Preferences > Test > Rational Test Automation Server). ▪ When you set Publish result after execution as Always or Prompt in the Preferences of Rational® Performance Tester (Window > Preferences > Test > Rational Test Automation Server > Results). ▪ When you used the Publish field or the Publish field along with the Publishreports field.
Publish_for	<p>Use this field to publish test results to Rational® Test Automation Server from UrbanCode™ Deploy based on the completion status of the tests.</p> <p>You must use the Publish_for field along with the Publish field. You can add multiple options separated by a comma. The following are the available options that you can use for the Publish_for field:</p> <ul style="list-style-type: none"> ▪ ALL: You can use this option irrespective of status of the test. ▪ PASS: You can use this option to publish test results for the tests that have passed. ▪ FAIL: You can use this option to publish test results for the tests that have failed. ▪ ERROR: You can use this option to publish test results for the tests that included errors. ▪ INCONCLUSIVE: You can use this option to publish test results for the inconclusive tests. <p>For example, <code>FAIL,ERROR</code></p>
Publishreports	<p>Use this field to publish specific test results to Rational® Test Automation Server.</p>

Fields	Description
	<p>The options that you can use with Publishreports are as follows:</p> <ul style="list-style-type: none"> ▪ STATS ▪ TESTLOG <p>For example, <i>STATS</i></p> <p>You must use the Publishreports field along with the Publish field. You can prefix the value of Publishreports with ! to publish the reports except the specified one.</p> <p>For example, <i>!STATS</i></p>
Rate	<p>Use this field to change the rate of the rate runner group.</p> <p>For example, <i>Rate Runner Group1=1/s, 3/m</i></p> <p>Where, <i>Rate Runner Group1</i> is the name of the Rate Runner group that has two stages. The desired rate for the first stage is one iteration per second and the rate for the second stage is three iterations per minute.</p> <p> Notes:</p> <ul style="list-style-type: none"> ▪ The name of the Rate Runner group must match with the name in the rate schedule. ▪ The Rate field creates a new copy of the rate schedule that contains the specified number of rate.
Resource Monitoring Labels Override	<p>Use this field to perform any of the following actions:</p> <ul style="list-style-type: none"> ▪ Enable the Resource Monitoring from Service option for a performance <i>schedule</i> if the Resource Monitoring from Service option is not enabled from the schedule editor in Rational® Performance Tester. ▪ Ignore Resource Monitoring sources that were set in the performance schedule and change for a label matching mode. ▪ Replace an existing set of Resource Monitoring labels that were set in the performance schedule and run the schedule with a new set of Resource Monitoring labels.

Fields	Description
	<p>If you have added a label in Rational® Test Automation Server for a Resource Monitoring source as <code>rm1</code>, then you can provide value as shown in the following example to collect data from the source:</p> <p><code>rm1</code></p> <p>If your label for resource monitoring contains a comma (,) then you must replace the single comma with the double comma while providing the value. For example, if you added a label to a Resource Monitoring source as <code>rm1,test</code>, then you must provide the value as shown in the following example to collect data from source:</p> <p><code>rm1,,test</code></p> <p> Notes:</p> <ul style="list-style-type: none"> ▪ You can use this field only when you want to run a Rate schedule or VU schedule. ▪ You can add multiple Resource Monitoring labels separated by a comma. ▪ You must add the Resource Monitoring labels to the Resource Monitoring sources on the Resource Monitoring page in the Rational® Test Automation Server project.
Results File	<p>Use this field to provide a different name to the result file.</p> <p>The results file is stored in the Results directory. The default name of the result file is the name of the <code>test</code> or <code>schedule</code> with a timestamp appended.</p>
User Comments	<p>Use this field to add text that you want to display in the user comments row of the report.</p> <p>For example, <i>test run with dataset</i></p> <p> Note: When you run tests by using the double quotation marks (") for the User Comments field, then the user comments row of a report does not contain double quotation marks.</p>

Fields	Description
	 For example, if you provide the value for the User Comments field as <code>test run with "dataset"</code> , then the user comments row of a report displays the value as <code>test run with dataset</code> . To work around this problem, you must create a command-line config file, and then run the test by using the Config File field.
Var File	Use this field to provide a complete path to an XML file that contains the variable name and value pairs.
VM Args	Use this field to specify the maximum heap size for the Java process. You can add multiple VM Arguments separated by a comma.



Note: The following fields are from UrbanCode™ Deploy:

- **Working Directory**
- **Precondition**
- **Post Processing Script**
- **Use Impersonation**
- **Auth Token Restriction**

You can accept the default values or change the values based on your requirements. For more information about these fields, see the related links.

c. Click **OK** to save the properties for the test.

9. Click **Save** in the design area.

Results

You have configured the process for the component in UrbanCode™ Deploy.

What to do next

You must create a resource in UrbanCode™ Deploy. See [Creating a resource in UrbanCode Deploy on page 149](#).

Related information

[Creating a command-line config file on page 370](#)

[Viewing test logs in Jaeger on page 156](#)

[Viewing test logs on page 467](#)

[Generating offline user tokens](#)

[IBM UrbanCode Deploy Documentation](#)

[Process step preconditions](#)

[Post-processing scripts](#)

[User impersonation for process steps](#)

[Restricting authentication tokens](#)

Creating a resource in UrbanCode™ Deploy

You must create a resource to associate agents with components that you created in UrbanCode™ Deploy.

Before you begin

- You must be familiar with working with UrbanCode™ Deploy.
 - You must have been granted access to UrbanCode™ Deploy.
1. Log in to UrbanCode™ Deploy, if you are not already logged in.

Result

The UrbanCode™ Deploy dashboard is displayed.

2. Click **Resources**, and then click **Create Top-Level Group**.

Result

The **Create Resource** dialog is displayed.

3. Enter a name for the resource in the **Name** field.
4. Click **Save**.

Results

You have created the resource in UrbanCode™ Deploy.

What to do next

You must configure the resource. See [Configuring the resource on page 149](#).

Related information

[IBM UrbanCode Deploy Documentation](#)

Configuring the resource

You must configure the resource to add an agent and associate the agent with the component.

Before you begin

- You must be familiar with working with UrbanCode™ Deploy.
- You must have performed the following tasks:

- Been granted access to UrbanCode™ Deploy.
- Created a component in UrbanCode™ Deploy. See [Creating a component in UrbanCode Deploy on page 135](#).
- Created a resource in UrbanCode™ Deploy. See [Creating a resource in UrbanCode Deploy on page 149](#).

1. Log in to UrbanCode™ Deploy, if you are not already logged in.

Result

The UrbanCode™ Deploy dashboard is displayed.

2. Click **Resources**.

Result

A list of resources that are available in UrbanCode™ Deploy is displayed.

3. Perform the following steps to add an agent to the resource:

- a. Click the resource from the list for which you want to add an agent.
- b. Click the **Actions** icon from the last column, and then click **Add Agent**.
- c. Select the agent from the drop-down list.



Note: The **Name** field is auto populated with the name of the agent.

- d. Click **Save**.

Result

The selected agent is added to the resource and you can view the status of the agent in the **Status** column.

4. Perform the following steps to add a component to the agent:

- a. Click the agent from the list for which you want to add a component.
- b. Click the **Actions** icon from the last column, and then click **Add Component**.
- c. Select the component from the drop-down list.



Note: The **Name** field is auto populated with the name of the component.

- d. Click **Save**.

Result

The selected component is added to the agent.

Results

You have configured the resource in UrbanCode™ Deploy.

What to do next

You must create an application. See [Creating an application in UrbanCode Deploy on page 151](#).

Related information

[IBM UrbanCode Deploy Documentation](#)

Creating an application in UrbanCode™ Deploy

You must create an application to fetch all the components together that you want to deploy.

Before you begin

- You must be familiar with working with UrbanCode™ Deploy.
 - You must have been granted access to UrbanCode™ Deploy.
1. Log in to UrbanCode™ Deploy, if you are not already logged in.

Result

The UrbanCode™ Deploy dashboard is displayed.

2. Click **Applications**.
3. Click **Create Applications**, and then **New Applications**.
4. Enter a name for the application in the **Name** field.

Result

The **Environments** page for the application that you created is displayed.

Results

You have created the application in UrbanCode™ Deploy.

What to do next

You must configure the application. See [Configuring the application on page 151](#).

Related information

[IBM UrbanCode Deploy Documentation](#)

Configuring the application

You must configure the application to associate resources with environments and define processes to run test assets.

Before you begin

- You must be familiar with working with UrbanCode™ Deploy.
- You must have performed the following tasks:

- Been granted access to UrbanCode™ Deploy.
- Created a component in UrbanCode™ Deploy. See [Creating a component in UrbanCode Deploy on page 135](#).
- Created a process for the component in UrbanCode™ Deploy. See [Creating a process in UrbanCode Deploy on page 135](#).
- Configure the process for the component in UrbanCode™ Deploy. See [Configuring the process on page 136](#).
- Created a resource in UrbanCode™ Deploy. See [Creating a resource in UrbanCode Deploy on page 149](#).
- Configured the resource in UrbanCode™ Deploy. See [Configuring the resource on page 149](#).
- Created an application in UrbanCode™ Deploy. See [Creating an application in UrbanCode Deploy on page 151](#).

1. Log in to UrbanCode™ Deploy, if you are not already logged in.

Result

The UrbanCode™ Deploy dashboard is displayed.

2. Click **Applications**.

Result

A list of applications that are available in UrbanCode™ Deploy is displayed.

3. Click the application that you want to configure from the **Name** column.

Result

The **Environments** page for the selected application is displayed.

4. Perform the following steps to create an environment for the application that you selected:

- a. Click **Create Environment**.
- b. Enter a name for the environment in the **Name** field.
- c. Click **Save**.

5. Perform the following steps to configure resources to the environment:

- a. Click the environment that you created.
- b. Click **Add Base Resources**.

Result

A list of resources that are available in UrbanCode™ Deploy is displayed.

- c. Select the checkbox to add resources to the environment.
- d. Click **Save**.

Result

You can view the corresponding agent and the component that you added for the resource by using the **Expand** icon.

6. Perform the following steps to add the component to the application:
 - a. Click **Applications**, and then select your application from the list.
 - b. Click the **Components** tab, and then **Add Components**.
 - c. Select the checkbox from the drop-down list to add components to the application.
 - d. Click **Save**.
7. Perform the following steps to create a process for the application:
 - a. Click the **Processes** tab, and then **Create Process**.
 - b. Enter a name for the process in the **Name** field.
 - c. Click **Save**.

Result

The **Design** tab for the process that you created is displayed.

8. Drag the component process listed under the **Component Process Steps** option from the left navigation pane and drop it into the design area.
9. Select the component process from the drop-down list in the **Operational (No Version Needed) Process** field.
10. Click **Save**.
11. Click the **Edit** icon, and then change the name of the properties.
12. Click **OK**, and then click **Save**.

Results

You have configured the application to run test assets from UrbanCode™ Deploy.

What to do next

You can run test assets from UrbanCode™ Deploy. See [Running tests from UrbanCode Deploy on page 153](#).

Related information

[IBM UrbanCode Deploy Documentation](#)

Running tests from UrbanCode™ Deploy

You can run test assets from UrbanCode™ Deploy as a **Process** to test the performance of your application under test.

Before you begin

- You must be familiar with working with UrbanCode™ Deploy.
- You must have performed the following tasks:

- Been granted access to UrbanCode™ Deploy.
 - Created a component in UrbanCode™ Deploy. See [Creating a component in UrbanCode Deploy on page 135](#).
 - Created a process for the component in UrbanCode™ Deploy. See [Creating a process in UrbanCode Deploy on page 135](#).
 - Configure the process for the component in UrbanCode™ Deploy. See [Configuring the process on page 136](#).
 - Created a resource in UrbanCode™ Deploy. See [Creating a resource in UrbanCode Deploy on page 149](#).
 - Configured the resource in UrbanCode™ Deploy. See [Configuring the resource on page 149](#).
 - Created an application in UrbanCode™ Deploy. See [Creating an application in UrbanCode Deploy on page 151](#).
 - Configured the application in UrbanCode™ Deploy. See [Configuring the application on page 151](#).
 - Started the UrbanCode™ Deploy agent. For more information, refer to the [IBM UrbanCode Deployment documentation](#).
 - Closed Rational® Performance Tester.
1. Log in to UrbanCode™ Deploy, if you are not already logged in.

Result

The UrbanCode™ Deploy dashboard is displayed.

2. Click **Applications**.

Result

A list of applications that are available in UrbanCode™ Deploy is displayed.

3. Click the application that you configured for the test run.

4. Click the **Request Process** icon.

Result

The **Run Application Process** page is displayed.

5. Select the process that you created for the application which contains the test assets from the **Process** drop-down list.

6. Click **Submit**.

Result

The UrbanCode™ Deploy dashboard shows the progress of the process request.

Results

You have run the performance test assets as **Process** from UrbanCode™ Deploy.

What to do next

After the process completes, you can view the status of the process displayed as follows:

- **Success:** When the test run is successful
- **Failed:** When the test run is failed

You can view the details of the test run in the **Execution Log** section. The **Reports information** section on the `Log` file displays the names of the report along with its corresponding URLs if you publish test results to IBM® Rational® Test Automation Server.

Integration of Jaeger with the product

Jaeger is software for tracing transactions between distributed services. You can use Jaeger to monitor and troubleshoot complex microservices environments.

You can set up the Jaeger UI in your local environment by using one of the following methods:

- One Jaeger agent shared by all Rational® Performance Tester agents
- One Jaeger agent for each Rational® Performance Tester agent

One Jaeger agent shared by all Rational® Performance Tester agents

When you use this method, ensure that the Jaeger agent is accessible by Rational® Performance Tester and all the Rational® Performance Tester agents. You must set the `JAEGER_AGENT_HOST` property as an environment variable by using the command line before running the schedule.

You must also ensure that the Jaeger agent ports 6831, 6832, and 5778 are accessible from other computers to communicate with the Rational® Performance Tester agent via the User Datagram Protocol (UDP). If you want to define any other Jaeger environment variables, set those environment variables only on Rational® Performance Tester.

One Jaeger agent for each Rational® Performance Tester agent

When you use this method, you must install the Jaeger agent in the same location where you installed Rational® Performance Tester and on all the Rational® Performance Tester agents.

You must also ensure that the Jaeger agent ports 6831, 6832, and 5778 are accessible from other processes on the same computer to communicate with the Rational® Performance Tester agent via the UDP. If you want to define any other Jaeger environment variables, set those environment variables on all the computers where the Jaeger agent is installed.

For more information about Jaeger, refer to [Jaeger documentation](#).

Related information

[Viewing test logs in Jaeger on page 156](#)

[Running a test or schedule from a command line on page 395](#)

Viewing test logs in Jaeger

You can use the Jaeger UI to view the test logs of the *tests* or *schedules* that you run from the command-line interface to analyze traces of transactions between distributed services.

Before you begin

You must have completed the following tasks:

- Downloaded Jaeger components from the [Jaeger](#) website.
- Created a test or schedule to run it from the command-line interface. See Recording HTTP tests or Creating a VU Schedule.

About this task

While running *tests* or *schedules* by using the command-line interface, you must include the command **-history jaeger** in your *test* or *schedule* run. Adding the **-history jaeger** enables you to view the test log of the completed *test* or *schedule* from the Jaeger UI in a web browser.

1. Run a *test* or *schedule* from the command-line interface by adding the **-history jaeger** option.

For example: **cmdline.bat -workspace workspace_full_path -project proj_rel_path -suite suite_rel_path -stdout -history jaeger**

Result

The *test* or *schedule* runs and the result of the run is displayed.

2. Open the Jaeger UI in a browser.

For example: `http://<host IP>:<port>`.

3. Select **Rational Test Product** from the **Service** list.
4. Click **Find Traces**.

In the Jaeger UI, you can view the entire test log of the *test* or *schedule* that you ran from the command-line.

What to do next

- You can use the Jaeger traces to analyze test results.
- You can compare the traces in the Jaeger UI with test logs in Rational® Performance Tester to confirm that they are the same.

Related information

[Running a test or schedule from a command line on page 395](#)

[Viewing test logs on page 467](#)

[Integration of Jaeger with the product on page 155](#)

Integration with Jenkins

You can use the Jenkins plugin to run tests on a Jenkins server.

To automate testing with Jenkins, you must configure Jenkins primary server and Jenkins secondary server. This configuration provides a single Jenkins installation on the Jenkins primary server to host multiple Jenkins secondary server for building and running tests. For more information about the Jenkins primary and secondary server relationship, refer to the [Jenkins](#) documentation.

You must install the required version of the Jenkins plugin on the Jenkins primary server, and install the product on the Jenkins secondary server, where you create tests.

You can use either the Jenkins **Freestyle** project or the **Pipeline** project to run test assets from Jenkins. With **Freestyle** project, you can create a build step from the Jenkins UI to run the test assets. Whereas the **Pipeline** project uses a simple text scripts based on the Groovy programming language. You can define pipeline scripts during the configuration of the **Pipeline** project by using any of the following methods:

- **Pipeline script:** In this method, you can directly enter a script in the field provided within the Jenkins UI or you can select a sample **Pipeline script** from the drop-down list.
- **Pipeline script from SCM:** In this method, you can create a script with a Groovy editor and then commit the script file into the Git repository.

For more information about the Jenkins Pipeline, refer to the [Jenkins](#) documentation.

Refer to the following topics to learn more about integrating Jenkins with .

Environment variables for Jenkins

You can add environment variables on the Jenkins server to run the Jenkins build by referring to environment variables.

You can add an environment variable on the Jenkins server by navigating to **Manage Jenkins > Configure System > Global properties**. You can enter the variable name by using any of the following methods for the corresponding text fields in the step:

- Use the dollar sign (\$) followed by the variable name.
For example, `$workspace`
- Use the dollar sign (\$) followed by the variable name between braces.
For example, `${workspace}`

The Jenkins plugin uses the actual value while running the job.

For example, if you add the environment variable named `workspace` with the value `C:\Users\IBM\workspace1`, then you can use `$workspace` or `${workspace}` as input to the **Workspace** field when running tests. During the run time, `$workspace` or `${workspace}` is substituted with its corresponding value `C:\Users\IBM\workspace1`.

Build parameters for Jenkins

You can define the build parameters when you configure either **Freestyle** or **Pipeline** projects. When you use the build parameters, Jenkins prompts you to enter a value for the defined parameter when you run the Jenkins job. Each parameter has a name and value.

Jenkins supports various types of build parameters. You can choose the type of parameter based on your requirements. Some of the commonly used types of parameters are as follows:

- Boolean
- Choice
- Credentials
- File
- String
- Multi-line String
- Password

After you add the parameter, the name that you provided for the parameter must be entered for the corresponding fields in the **Run IBM Rational Performance Tester test** step in the following format:

```
${name_of_the_parameter}
```

For example, `${workspace}`

Each time when you run a test from Jenkins, you might want to export the statistical report in a different format. In such a case, you can add a **Choice** parameter by selecting the **This project is parametrized** option from the **General** tab when you configure the project.

You can then provide a name for the **Choice** parameter as `Stat_Report_Format` and the available formats of the report as `csv`, `json`, `simple.csv`, `full.csv`, `simple.json`, and `full.json` in the **Choices** field. You can also describe the parameter in the **Description** field, which is an optional field.

When you provide the details of the test run in the **Run IBM Rational Performance Tester test** step, you can enter the value for the **Exported Statistical Report Format** field as `${Stat_Report_Format}`.

When you run the Jenkins job, the `Stat_Report_Format` parameter displays to choose the format of the report. You can select the format from the drop-down list. You must then click **Build** to run the test assets.

 Related information

[Configuring the Freestyle project on page 160](#)

[Creating a pipeline script from Jenkins on page 171](#)

[Running tests from Jenkins on page 184](#)

Task flows for running test assets from Jenkins

You can perform certain tasks to run test assets from the Jenkins **Freestyle** project or the **Pipeline** project.

The following table lists the task flows for running test assets from the Jenkins **Freestyle** project:

Tasks	More information
Install the Jenkins plugin.	Installing the plugin on the Jenkins primary server on page 160
Configure the Freestyle project.	Configuring the Freestyle project on page 160
Run Rational® Performance Tester tests on Jenkins.	Running tests from Jenkins on page 184

You can perform the following tasks to run test assets from the Jenkins **Pipeline** project by using the sample script:

Tasks	More information
Install the Jenkins plugin.	Installing the plugin on the Jenkins primary server on page 160
Create a pipeline script from the Jenkins UI.	Creating a pipeline script from Jenkins on page 171
Configure the Pipeline project.	Configuring the Pipeline project by using the sample script on page 182
Run Rational® Performance Tester tests on Jenkins.	Running tests from Jenkins on page 184

You can perform the following tasks to run test assets from the Jenkins **Pipeline** project by using the script from Source Code Management (SCM):

Tasks	More information
Install the Jenkins plugin.	Installing the plugin on the Jenkins primary server on page 160

Tasks	More information
Create a pipeline script and commit the script to the Git repository.	For more information, refer to the Jenkins documentation.
Configure the Pipeline project.	Configuring the Pipeline project by using the script from SCM on page 183
Run Rational® Performance Tester tests on Jenkins.	Running tests from Jenkins on page 184

Installing the plugin on the Jenkins primary server

You must install the Jenkins plugin to run test assets from the Jenkins server.

Before you begin

You must have completed the following tasks:

- Verified that you have a Jenkins primary server and secondary server.

1. Log in to the Jenkins server.

Result

The Jenkins dashboard is displayed.

2. Click **Manage Jenkins > Manage plugins**, and then click **Advanced** tab.
3. Click **Choose File** and then locate and open the Jenkins plugin.
4. Click **Upload**.

Result

The Jenkins plugin is displayed in the **Installed** tab.

5. Perform the following steps to provide Random TCP Ports for Java™ Network Launch Protocol (JNLP) agents:
 - a. Click **Manage Jenkins** from the Jenkins dashboard.
 - b. Click **Configure Global Security** from the **Security** section.
 - c. Click **Random** from the **Agents** section.
 - d. Click **Save** to save and apply the changes.

Results

You have installed the Jenkins plugin on the Jenkins primary server.

What to do next

You can run the test from the Jenkins server. See [Running tests from Jenkins on page 184](#).

Configuring the Freestyle project

You must configure a **Freestyle** project to add a build step, and then run test assets from Jenkins.

Before you begin

You must have completed the following tasks:

- Installed the Jenkins plugin on the Jenkins primary server. See [Installing the plugin on the Jenkins primary server on page 160](#).
- Created an Agent in Jenkins. For more information about creating Agents, refer to the [Jenkins](#) documentation.
- Copied the name of the labels that you provided in the **Labels** field when you created the Agent.
- Created a Jenkins **Freestyle** project.

About this task

You must provide the values for certain fields in the Jenkins UI to run tests from Jenkins. The minimum fields that you need to configure along with **Name** are **Workspace**, **Project**, and **Test Suite Name** or **Config File**. The remaining fields are optional.

For example, if you provide a complete path of a configuration file that contains the parameters for a `test` or `schedule` run in the **Config File** field, then you need not provide values for the **Workspace**, **Project**, and **Test Suite Name** fields.




Note: If Rational® Performance Tester is not installed in default location, then you must provide the value for the **IMShared Location** field along with other mandatory fields.

1. Open the Jenkins **Freestyle** project, and then click **Configure**.
2. Select the **Restrict where this project can be run** checkbox from the **General** tab, and then enter the name of the label that you provided during the creation of Agent in the **Label Expression** field.
3. Click the **Build** tab, and then click **Add build step**.
4. Select the option from the drop-down list.
5. Provide the details about the test run for the fields by referring to the following table:




The following table lists the required fields that you can provide to run the test from Jenkins:



Fields	Description
Config File	<p>Enter the complete path to a file that contains the parameters for a <code>test</code> or <code>schedule</code> run.</p> <p>For example, <code>C:\Users\IBM\rational\workspace\Project1\config.txt</code></p> <p>For more information about creating a command line config file from Rational® Performance Tester, see the Related information section.</p>



Fields	Description
	 Remember: If you provide a complete path of a configuration file in the Config File field, then you need not provide values for the Workspace, Project, and Test Suite Name fields.
Name	<p>Enter the name for the Jenkins build step.</p> <p>For example, <i>testrun1</i></p>
Project	<p>Enter the name of the project that has performance test assets.</p> <p>For example, <i>testproj1</i></p>
Test Suite Name	<p>Enter the file name of the performance test assets relative to the project.</p> <p>For example, <i>test1.testsuite</i></p>
Workspace	<p>Enter the complete path of the Eclipse workspace.</p> <p>For example, <i>C:/Users/IBM/rationalsdp/workspace1</i></p>



The following table lists the optional fields that you can provide to run the test from Jenkins:



Fields	Description
Custom Report Format Files	<p>Use this field to specify IDs of reports that you want to export in place of the default reports.</p> <p>You can provide multiple report IDs separated by a comma. You can navigate to Preferences of Rational® Performance Tester (Window > Preferences > Test > Performance Test Reports > Export Reports), and then select Show Report IDs checkbox to view the report IDs.</p> <p>You must use the Custom Report Format Files filed along with the Exported Statistical Report Data File or Exported Statistical Report in html field.</p> <p>For example, you can provide <code>http</code> as a value in the Custom Report Format Files field to export a Performance Report.</p>
Dataset Override	<p>Use this field to replace the dataset values during a <code>test</code> or <code>schedule</code> run.</p>


Fields	Description
	<p>You must ensure that both original and new datasets are in the same workspace and have the same column names. When you enter a value for the Dataset Override field, you must also include the path to the dataset. You must provide the values for the Dataset Override field in the following format:</p> <pre data-bbox="511 451 1315 520">/project_name/ds_path/original_ds.csv:/project_name/ds_path/new_ds.csv</pre> <p>For example, <i>/proj1/Datasets/ds1.csv:/proj1023/Datasets/new_ds1.csv</i></p> <p>You can override multiple datasets that are saved in a different project by adding multiple paths to the dataset separated by a semicolon. For a <code>test</code> or <code>schedule</code>, the default value is the dataset specified in the test editor or schedule editor.</p>
Duration	<p>Use this field to change the duration of the stages in the rate schedule.</p> <p>For example, <i>Stage1=10s, Stage2=3m</i></p> <p>The stage number specified must exist in the rate schedule.</p> <p> Note: The Duration field creates a new copy of the rate schedule that contains the specified number of duration.</p>
Exported HTTP Test log File	<p>Use this field to specify the path of the file directory to store the exported HTTP test log in a <code>.txt</code> format.</p> <p>For example, <i>C:/Users/Documents/Jenkins/logexp.txt</i></p> <p> Note: If you do not specify a value for the Exported HTTP Test log File, then no logs for the test run is stored in the Jenkins <code>workspace/temp</code> directory.</p>
Exported Statistical Report Data File	<p>Use this field to provide the complete path to a directory that you can use to store exported reports in a comma-separated values (CSV) format.</p> <p>For example, <i>C:/Users/Documents/tests</i></p> <p> Note: If you do not specify a value for the Custom Report Format Files field, then the reports that you selected in Preferences of Rational® Performance</p>


Fields	Description
	 Tester (Window > Preferences > Test > Performance Test Reports > Export Reports) are exported.
Exported Statistical Report Format	<p>Use this field to specify a format for the report that you want to export.</p> <p>You must use at least one of the following options along with the Exported Statistical Report Data File field:</p> <ul style="list-style-type: none"> ◦ <i>simple.csv</i> ◦ <i>full.csv</i> ◦ <i>simple.json</i> ◦ <i>full.json</i> ◦ <i>csv</i> ◦ <i>json</i> <p>For example, <i>json</i>.</p> <p>You can add multiple formats for the report separated by a comma. When you want to export both simple and full type of report in a <i>json</i> or <i>csv</i> format, you can specify <i>json</i> or <i>csv</i> as the format in the field.</p> <p>The reports are saved to the location specified in the Exported Statistical Report Data File field.</p> <p> Note: The values provided in the Exported Statistical Report Format field always take precedence over the Export Reports options set in the product preferences (Window > Preferences > Test > Performance Test Reports > Export Reports).</p>
Exported Statistical Report in html	<p>Use this field to provide the complete path to a directory that you can use to export web analytic results.</p> <p>The results are exported to the specified directory. You can analyze the results on a web browser without using Rational® Performance Tester.</p> <p>For example, <i>C:/Users/Documents/Reports</i></p>



Fields	Description								
	 Note: When you do not use the Exported Statistical Report in html field, then the Performance report is stored in a <code>workspace/temp</code> directory.								
History	<p>Use this field when you want to view a record of all events that occurred during a test or schedule run.</p> <p>You can use any of the following options:</p> <ul style="list-style-type: none"> ◦ jaeger: To send test logs to the Jaeger UI during the <code>test</code> or <code>schedule</code> run. <p> Note: You must set the <code>JAEGER_AGENT_HOST</code> property as an environment variable by using the command line before you use the jaeger option to send test logs to the Jaeger UI.</p> <ul style="list-style-type: none"> ◦ testlog: To send test logs as traditional test logs in Rational® Performance Tester during the <code>test</code> or <code>schedule</code> run. ◦ null: To send no test logs either to the Jaeger UI or Rational® Performance Tester during the <code>test</code> or <code>schedule</code> run. <p>For example, <i>jaeger</i></p> <p>You can add multiple options separated by a comma to send test logs during the <code>test</code> or <code>schedule</code> run to Rational® Performance Tester and the Jaeger UI.</p> <p>For example, <i>jaeger,testlog</i></p> <p>For more information about how to view test logs in the Jaeger UI and Rational® Performance Tester, see the Related information section.</p>								
IMShared Location	<p>Enter the complete path to the <code>IMShared</code> directory, if it is not in the default location.</p> <p>For example, <i>D:\Testtool\IBM\IBMIMShared</i></p> <p>The default location of the <code>IMShared</code> directory is as follows:</p> <table border="1" data-bbox="511 1619 1419 1839"> <thead> <tr> <th data-bbox="511 1619 813 1673">Operating system</th> <th data-bbox="813 1619 1419 1673">The default path to the directory</th> </tr> </thead> <tbody> <tr> <td data-bbox="511 1673 813 1732">Windows®</td> <td data-bbox="813 1673 1419 1732">C:\Program Files\IBM\IBMIMShared</td> </tr> <tr> <td data-bbox="511 1732 813 1791">Linux™</td> <td data-bbox="813 1732 1419 1791">/opt/IBM/IBMIMShared</td> </tr> <tr> <td data-bbox="511 1791 813 1839">Mac</td> <td data-bbox="813 1791 1419 1839">/Application/IBM/IBMIMShared</td> </tr> </tbody> </table>	Operating system	The default path to the directory	Windows®	C:\Program Files\IBM\IBMIMShared	Linux™	/opt/IBM/IBMIMShared	Mac	/Application/IBM/IBMIMShared
Operating system	The default path to the directory								
Windows®	C:\Program Files\IBM\IBMIMShared								
Linux™	/opt/IBM/IBMIMShared								
Mac	/Application/IBM/IBMIMShared								


Fields	Description
Labels	<p>Use this field to add labels to test results when the test run is complete.</p> <p>For example, <i>label1, label2</i></p> <p>You can add multiple labels to a test result separated by a comma. When you run test assets, then the labels that you added are displayed on the Performance Report in Rational® Performance Tester.</p> <p>The Results page of Rational® Test Automation Server displays the label that you added in the Labels field for the specific test asset in the following conditions:</p> <ul style="list-style-type: none"> ◦ When you set Publish result after execution as Always or Prompt in the Preferences of Rational® Performance Tester (Windows > Preferences > Test > Rational Test Automation Server). ◦ When you use the Labels field along with the Publish field. <p> Note: When you run tests by using the double quotation marks ("") for the Labels field, then the labels in the test result do not include double quotation marks.</p> <p>For example, if you provide the value for the Labels field as "100" users, then the labels display 100 users in the test result.</p> <p>To work around this problem, you must create a command-line config file, and then run the test by using the Config File field.</p>
Number of Virtual Users	<p>Use this field to override the default number of virtual users in the <code>test</code> or <code>schedule</code> run.</p> <p>For a <code>schedule</code>, the default is the number of users specified in the schedule editor and for a <code>test</code>, the default is one user.</p> <p> Note: The Number of Virtual Users field creates a new copy of the <code>schedule</code> that contains the specified number of users.</p>
Overwrite Results File	<p>Select or clear this field to determine whether a result file with the same name must be overwritten or not.</p> <p>The default status of the field is selected. Therefore, the file is overwritten and retains the same file name.</p>

Fields	Description
Publish	<p>Use this field to publish test results to Rational® Test Automation Server from Jenkins.</p> <p> Remember: Before you use the Publish field, you must provide the offline user token of Rational® Test Automation Server by using any of the following methods:</p> <ul style="list-style-type: none"> ◦ Set the value of the RTCP_OFFLINE_TOKEN environment variable to the offline user token of Rational® Test Automation Server by using the command-line interface. ◦ Provide the offline user token of Rational® Test Automation Server in the preferences of Rational® Performance Tester (Window > Preferences > Test > Rational Test Automation Server) <p>You must use one of the following values in the Publish field:</p> <ul style="list-style-type: none"> ◦ <code>serverURL#project.name=name_of_the_project&teamspace.name=name_of_the_teamspace</code> ◦ <code>serverURL#project.name=name_of_the_project&teamspace.alias=name_of_the_teamspace_alias</code> ◦ no <p>The no option is useful if the product preferences are set to publish the results, but you do not want to publish them.</p> <p>For example, <code>URL_OF_RationalTest_Server#project.name=test&teamspace.name=ts1</code></p> <p>Where:</p> <ul style="list-style-type: none"> ◦ <code>URL_OF_RationalTest_Server</code> is the URL of Rational® Test Automation Server. ◦ <code>test</code> is the name of the project in Rational® Test Automation Server. ◦ <code>ts1</code> is the name of the team space. <p> Notes:</p> <ul style="list-style-type: none"> ◦ If you have a project with the same name in different team spaces, then you must append either the &teamspace.name=name_of_the_teamspace or &teamspace.alias=name_of_the_teamspace_alias options.

Fields	Description
	 <ul style="list-style-type: none"> ◦ If the name of the project or team space contains a special character, then you must replace it with %<Hexvalue_of_special_character>. <p>For example, if the name of the team space is <i>Initial Team Space</i>, then you must provide it as <i>Initial%20Team%20Space</i>.</p> <p>Where %20 is the Hex value of <i>space</i> character.</p> <ul style="list-style-type: none"> ◦ The values provided in the Publish field always take precedence over the Results options set in the product preferences (Window > Preferences > Test > Rational Test Automation Server > Results). <p>The Reports information section on the Console Output page displays the names of the report along with its corresponding URLs in the following conditions:</p> <ul style="list-style-type: none"> ◦ When you configured the URL of in Preferences of Rational® Performance Tester (Window > Preferences > Test > Rational Test Automation Server). ◦ When you set Publish result after execution as Always or Prompt in the Preferences of Rational® Performance Tester (Window > Preferences > Test > Rational Test Automation Server > Results). ◦ When you used the Publish field or the Publish field along with the Publishreports field.
Publish_for	<p>Use this field to publish test results to Rational® Test Automation Server from Jenkins based on the completion status of the tests.</p> <p>You must use the Publish_for field along with the Publish field. You can add multiple options separated by a comma. The following are the available options that you can use for the Publish_for field:</p> <ul style="list-style-type: none"> ◦ ALL: You can use this option irrespective of status of the test. ◦ PASS: You can use this option to publish test results for the tests that have passed. ◦ FAIL: You can use this option to publish test results for the tests that have failed. ◦ ERROR: You can use this option to publish test results for the tests that included errors. ◦ INCONCLUSIVE: You can use this option to publish test results for the inconclusive tests.

Fields	Description
	For example, <i>FAIL,ERROR</i>
Publishreports	<p>Use this field to publish specific test results to Rational® Test Automation Server.</p> <p>The options that you can use with Publishreports are as follows:</p> <ul style="list-style-type: none"> ◦ STATS ◦ TESTLOG <p>For example, <i>STATS</i></p> <p>You must use the Publishreports field along with the Publish field. You can prefix the value of Publishreports with ! to publish the reports except the specified one.</p> <p>For example, <i>!STATS</i></p>
Rate	<p>Use this field to change the rate of the rate runner group.</p> <p>For example, <i>Rate Runner Group1=1/s, 3/m</i></p> <p>Where, <i>Rate Runner Group1</i> is the name of the Rate Runner group that has two stages. The desired rate for the first stage is one iteration per second and the rate for the second stage is three iterations per minute.</p> <p> Notes:</p> <ul style="list-style-type: none"> ◦ The name of the Rate Runner group must match with the name in the rate schedule. ◦ The Rate field creates a new copy of the rate schedule that contains the specified number of rate.
Resource Monitoring Labels Override	<p>Use this field to perform any of the following actions:</p> <ul style="list-style-type: none"> ◦ Enable the Resource Monitoring from Service option for a performance <i>sched-ule</i> if the Resource Monitoring from Service option is not enabled from the schedule editor in Rational® Performance Tester. ◦ Ignore Resource Monitoring sources that were set in the performance <i>sched-ule</i> and change for a label matching mode. ◦ Replace an existing set of Resource Monitoring labels that were set in the performance schedule and run the schedule with a new set of Resource Monitoring labels.

Fields	Description
	<p>If you have added a label in Rational® Test Automation Server for a Resource Monitoring source as <code>rm1</code>, then you can provide value as shown in the following example to collect data from the source:</p> <pre>rm1</pre> <p>If your label for resource monitoring contains a comma (,) then you must replace the single comma with the double comma while providing the value. For example, if you added a label to a Resource Monitoring source as <code>rm1,test</code>, then you must provide the value as shown in the following example to collect data from source:</p> <pre>rm1,,test</pre> <p> Notes:</p> <ul style="list-style-type: none"> ◦ You can use this field only when you want to run a Rate schedule or VU schedule. ◦ You can add multiple Resource Monitoring labels separated by a comma. ◦ You must add the Resource Monitoring labels to the Resource Monitoring sources on the Resource Monitoring page in the Rational® Test Automation Server project.
Results File	<p>Use this field to provide a different name to the result file.</p> <p>The results file is stored in the Results directory. The default name of the result file is the name of the <code>test</code> or <code>schedule</code> with a timestamp appended.</p>
User Comments	<p>Use this field to add text that you want to display in the user comments row of the report.</p> <p>For example, <i>test run with dataset</i></p> <p> Note: When you run tests by using the double quotation marks (") for the User Comments field, then the user comments row of a report does not contain double quotation marks.</p> <p>For example, if you provide the value for the User Comments field as <code>test run with "dataset"</code>, then the user comments row of a report displays the value as <code>test run with dataset</code>.</p>

Fields	Description
	 To work around this problem, you must create a command-line config file, and then run the test by using the Config File field.
Var File	Use this field to provide a complete path to an XML file that contains the variable name and value pairs.
VM Args	<p>Use this field to specify the maximum heap size for the Java process.</p> <p>You can add multiple VM Arguments separated by a comma.</p>

6. **Optional:** Click **Add build step** again, and provide details for the next test to run multiple tests under the same job.
7. Click **Save**.

Results

You have configured the **Freestyle** project by adding the build step.

What to do next

You can run test assets from the Jenkins server. See [Running tests from Jenkins on page 184](#).

Related information

[Creating a command-line config file on page 370](#)

[Viewing test logs in Jaeger on page 156](#)

[Viewing test logs on page 467](#)

[Generating offline user tokens](#)

Creating a pipeline script from Jenkins

You must create a pipeline script to run test assets from the Jenkins **Pipeline** project.

Before you begin

You must have completed the following tasks:

- Installed the Jenkins plugin on the Jenkins primary server. See [Installing the plugin on the Jenkins primary server on page 160](#).
- Created a Jenkins **Pipeline** project.

About this task

You must provide the values for certain fields in the Jenkins UI to run tests from Jenkins. The minimum fields that you need to configure along with **Name** are **Workspace**, **Project**, and **Test Suite Name** or **Config File**. The remaining fields are optional.


For example, if you provide a complete path of a configuration file that contains the parameters for a `test` or `schedule` run in the **Config File** field, then you need not provide values for the **Workspace**, **Project**, and **Test Suite Name** fields.



Note: If Rational® Performance Tester is not installed in default location, then you must provide the value for the **IMShared Location** field along with other mandatory fields.

1. Open your Jenkins **Pipeline** project from the list.
2. Click **Configure**, and then select the **Pipeline** tab.
3. Click **Pipeline Syntax** to generate a pipeline script.
4. Select the **step: General Build Step** option from the drop-down list in the **Sample Step** field.
5. Select the option from the drop-down list in the **Build Step** field.
6. Provide the details about the test run for the fields by referring to the following table:




The following table lists the required fields that you can provide to run the test from Jenkins:



Fields	Description
<p>Config File</p>	<p>Enter the complete path to a file that contains the parameters for a <code>test</code> or <code>schedule</code> run.</p> <p>For example, <i>C:\Users\IBM\rational\sd\workspace\Project1\config.txt</i></p> <p>For more information about creating a command line config file from Rational® Performance Tester, see the Related information section.</p> <p> Remember: If you provide a complete path of a configuration file in the Config File field, then you need not provide values for the Workspace, Project, and Test Suite Name fields.</p>
<p>Name</p>	<p>Enter the name for the Jenkins build step.</p> <p>For example, <i>testrun1</i></p>
<p>Project</p>	<p>Enter the name of the project that has performance test assets.</p> <p>For example, <i>testproj1</i></p>


Fields	Description
Test Suite Name	<p>Enter the file name of the performance test assets relative to the project.</p> <p>For example, <i>test1.testsuite</i></p>
Workspace	<p>Enter the complete path of the Eclipse workspace.</p> <p>For example, <i>C:/Users/IBM/rationalmdp/workspace1</i></p>




The following table lists the optional fields that you can provide to run the test from Jenkins:

Fields	Description
Custom Report Format Files	<p>Use this field to specify IDs of reports that you want to export in place of the default reports.</p> <p>You can provide multiple report IDs separated by a comma. You can navigate to Preferences of Rational® Performance Tester (Window > Preferences > Test > Performance Test Reports > Export Reports), and then select Show Report IDs checkbox to view the report IDs.</p> <p>You must use the Custom Report Format Files filed along with the Exported Statistical Report Data File or Exported Statistical Report in html field.</p> <p>For example, you can provide <code>http</code> as a value in the Custom Report Format Files field to export a Performance Report.</p>
Dataset Override	<p>Use this field to replace the dataset values during a <code>test</code> or <code>schedule</code> run.</p> <p>You must ensure that both original and new datasets are in the same workspace and have the same column names. When you enter a value for the Dataset Override field, you must also include the path to the dataset. You must provide the values for the Dataset Override field in the following format:</p> <pre>/project_name/ds_path/original_ds.csv:/project_name/ds_path/new_ds.csv</pre> <p>For example, <code>/proj1/Datasets/ds1.csv:/proj1023/Datasets/new_ds1.csv</code></p> <p>You can override multiple datasets that are saved in a different project by adding multiple paths to the dataset separated by a semicolon. For a <code>test</code> or <code>schedule</code>, the default value is the dataset specified in the test editor or schedule editor.</p>


Fields	Description
Duration	<p>Use this field to change the duration of the stages in the rate schedule.</p> <p>For example, <i>Stage1=10s, Stage2=3m</i></p> <p>The stage number specified must exist in the rate schedule.</p> <p> Note: The Duration field creates a new copy of the rate schedule that contains the specified number of duration.</p>
Exported HTTP Test log File	<p>Use this field to specify the path of the file directory to store the exported HTTP test log in a <code>.txt</code> format.</p> <p>For example, <i>C:/Users/Documents/Jenkins/logexp.txt</i></p> <p> Note: If you do not specify a value for the Exported HTTP Test log File, then no logs for the test run is stored in the Jenkins <code>workspace/temp</code> directory.</p>
Exported Statistical Report Data File	<p>Use this field to provide the complete path to a directory that you can use to store exported reports in a comma-separated values (CSV) format.</p> <p>For example, <i>C:/Users/Documents/tests</i></p> <p> Note: If you do not specify a value for the Custom Report Format Files field, then the reports that you selected in Preferences of Rational® Performance Tester (Window > Preferences > Test > Performance Test Reports > Export Reports) are exported.</p>
Exported Statistical Report Format	<p>Use this field to specify a format for the report that you want to export.</p> <p>You must use at least one of the following options along with the Exported Statistical Report Data File field:</p> <ul style="list-style-type: none"> ◦ <i>simple.csv</i> ◦ <i>full.csv</i> ◦ <i>simple.json</i> ◦ <i>full.json</i>


Fields	Description
	<ul style="list-style-type: none"> ◦ <i>csv</i> ◦ <i>json</i> <p>For example, <i>json</i>.</p> <p>You can add multiple formats for the report separated by a comma. When you want to export both simple and full type of report in a <i>json</i> or <i>csv</i> format, you can specify <i>json</i> or <i>csv</i> as the format in the field.</p> <p>The reports are saved to the location specified in the Exported Statistical Report Data File field.</p> <p> Note: The values provided in the Exported Statistical Report Format field always take precedence over the Export Reports options set in the product preferences (Window > Preferences > Test > Performance Test Reports > Export Reports).</p>
Exported Statistical Report in html	<p>Use this field to provide the complete path to a directory that you can use to export web analytic results.</p> <p>The results are exported to the specified directory. You can analyze the results on a web browser without using Rational® Performance Tester.</p> <p>For example, <i>C:/Users/Documents/Reports</i></p> <p> Note: When you do not use the Exported Statistical Report in html field, then the Performance report is stored in a <i>workspace/temp</i> directory.</p>
History	<p>Use this field when you want to view a record of all events that occurred during a test or schedule run.</p> <p>You can use any of the following options:</p>



Fields	Description								
	<ul style="list-style-type: none"> ◦ jaeger: To send test logs to the Jaeger UI during the <code>test</code> or <code>schedule</code> run. <p> Note: You must set the <code>JAEGER_AGENT_HOST</code> property as an environment variable by using the command line before you use the jaeger option to send test logs to the Jaeger UI.</p> <ul style="list-style-type: none"> ◦ testlog: To send test logs as traditional test logs in Rational® Performance Tester during the <code>test</code> or <code>schedule</code> run. ◦ null: To send no test logs either to the Jaeger UI or Rational® Performance Tester during the <code>test</code> or <code>schedule</code> run. <p>For example, <i>jaeger</i></p> <p>You can add multiple options separated by a comma to send test logs during the <code>test</code> or <code>schedule</code> run to Rational® Performance Tester and the Jaeger UI.</p> <p>For example, <i>jaeger,testlog</i></p> <p>For more information about how to view test logs in the Jaeger UI and Rational® Performance Tester, see the Related information section.</p>								
IMShared Location	<p>Enter the complete path to the <code>IMShared</code> directory, if it is not in the default location.</p> <p>For example, <i>D:\Testtool\IBM\IBMIMShared</i></p> <p>The default location of the <code>IMShared</code> directory is as follows:</p> <table border="1" data-bbox="511 1270 1421 1495"> <thead> <tr> <th data-bbox="511 1270 812 1325">Operating system</th> <th data-bbox="812 1270 1421 1325">The default path to the directory</th> </tr> </thead> <tbody> <tr> <td data-bbox="511 1325 812 1379">Windows®</td> <td data-bbox="812 1325 1421 1379">C:\Program Files\IBM\IBMIMShared</td> </tr> <tr> <td data-bbox="511 1379 812 1434">Linux™</td> <td data-bbox="812 1379 1421 1434">/opt/IBM/IBMIMShared</td> </tr> <tr> <td data-bbox="511 1434 812 1495">Mac</td> <td data-bbox="812 1434 1421 1495">/Application/IBM/IBMIMShared</td> </tr> </tbody> </table>	Operating system	The default path to the directory	Windows®	C:\Program Files\IBM\IBMIMShared	Linux™	/opt/IBM/IBMIMShared	Mac	/Application/IBM/IBMIMShared
Operating system	The default path to the directory								
Windows®	C:\Program Files\IBM\IBMIMShared								
Linux™	/opt/IBM/IBMIMShared								
Mac	/Application/IBM/IBMIMShared								
Labels	<p>Use this field to add labels to test results when the test run is complete.</p> <p>For example, <i>label1, label2</i></p> <p>You can add multiple labels to a test result separated by a comma. When you run test assets, then the labels that you added are displayed on the Performance Report in Rational® Performance Tester.</p>								


Fields	Description
	<p>The Results page of Rational® Test Automation Server displays the label that you added in the Labels field for the specific test asset in the following conditions:</p> <ul style="list-style-type: none"> ◦ When you set Publish result after execution as Always or Prompt in the Preferences of Rational® Performance Tester (Windows > Preferences > Test > Rational Test Automation Server). ◦ When you use the Labels field along with the Publish field. <p> Note: When you run tests by using the double quotation marks ("") for the Labels field, then the labels in the test result do not include double quotation marks.</p> <p>For example, if you provide the value for the Labels field as "100" users, then the labels display 100 users in the test result.</p> <p>To work around this problem, you must create a command-line config file, and then run the test by using the Config File field.</p>
Number of Virtual Users	<p>Use this field to override the default number of virtual users in the <code>test</code> or <code>schedule</code> run.</p> <p>For a <code>schedule</code>, the default is the number of users specified in the schedule editor and for a <code>test</code>, the default is one user.</p> <p> Note: The Number of Virtual Users field creates a new copy of the <code>schedule</code> that contains the specified number of users.</p>
Overwrite Results File	<p>Select or clear this field to determine whether a result file with the same name must be overwritten or not.</p> <p>The default status of the field is selected. Therefore, the file is overwritten and retains the same file name.</p>
Publish	<p>Use this field to publish test results to Rational® Test Automation Server from Jenkins.</p> <p> Remember: Before you use the Publish field, you must provide the offline user token of Rational® Test Automation Server by using any of the following methods:</p>

Fields	Description
	<div data-bbox="524 268 570 317" data-label="Image"> </div> <ul style="list-style-type: none"> ◦ Set the value of the RTCP_OFFLINE_TOKEN environment variable to the offline user token of Rational® Test Automation Server by using the command-line interface. ◦ Provide the offline user token of Rational® Test Automation Server in the preferences of Rational® Performance Tester (Window > Preferences > Test > Rational Test Automation Server) <p>You must use one of the following values in the Publish field:</p> <ul style="list-style-type: none"> ◦ <code>serverURL#project.name=name_of_the_project&teamspace.name=name_of_the_teamspace</code> ◦ <code>serverURL#project.name=name_of_the_project&teamspace.alias=name_of_the_teamspace_alias</code> ◦ no <p>The no option is useful if the product preferences are set to publish the results, but you do not want to publish them.</p> <p>For example, <code>URL_OF_RationalTest_Server#project.name=test&teamspace.name=ts1</code></p> <p>Where:</p> <ul style="list-style-type: none"> ◦ <code>URL_OF_RationalTest_Server</code> is the URL of Rational® Test Automation Server. ◦ <code>test</code> is the name of the project in Rational® Test Automation Server. ◦ <code>ts1</code> is the name of the team space. <div data-bbox="524 1346 570 1394" data-label="Image"> </div> <p>Notes:</p> <ul style="list-style-type: none"> ◦ If you have a project with the same name in different team spaces, then you must append either the &teamspace.name=name_of_the_teamspace or &teamspace.alias=name_of_the_teamspace_alias options. ◦ If the name of the project or team space contains a special character, then you must replace it with <code>%<Hexvalue_of_special_character></code>. <p>For example, if the name of the team space is <i>Initial Team Space</i>, then you must provide it as <code>Initial%20Team%20Space</code>.</p>

Fields	Description
	 <p>Where %20 is the Hex value of <code>space</code> character.</p> <ul style="list-style-type: none"> The values provided in the Publish field always take precedence over the Results options set in the product preferences (Window > Preferences > Test > Rational Test Automation Server > Results). <p>The Reports information section on the Console Output page displays the names of the report along with its corresponding URLs in the following conditions:</p> <ul style="list-style-type: none"> When you configured the URL of in Preferences of Rational® Performance Tester (Window > Preferences > Test > Rational Test Automation Server). When you set Publish result after execution as Always or Prompt in the Preferences of Rational® Performance Tester (Window > Preferences > Test > Rational Test Automation Server > Results). When you used the Publish field or the Publish field along with the Publishreports field.
Publish_for	<p>Use this field to publish test results to Rational® Test Automation Server from Jenkins based on the completion status of the tests.</p> <p>You must use the Publish_for field along with the Publish field. You can add multiple options separated by a comma. The following are the available options that you can use for the Publish_for field:</p> <ul style="list-style-type: none"> ALL: You can use this option irrespective of status of the test. PASS: You can use this option to publish test results for the tests that have passed. FAIL: You can use this option to publish test results for the tests that have failed. ERROR: You can use this option to publish test results for the tests that included errors. INCONCLUSIVE: You can use this option to publish test results for the inconclusive tests. <p>For example, <i>FAIL,ERROR</i></p>
Publishreports	<p>Use this field to publish specific test results to Rational® Test Automation Server.</p> <p>The options that you can use with Publishreports are as follows:</p>

Fields	Description
	<ul style="list-style-type: none"> ◦ STATS ◦ TESTLOG <p>For example, <i>STATS</i></p> <p>You must use the Publishreports field along with the Publish field. You can prefix the value of Publishreports with ! to publish the reports except the specified one.</p> <p>For example, <i>!STATS</i></p>
Rate	<p>Use this field to change the rate of the rate runner group.</p> <p>For example, <i>Rate Runner Group1=1/s, 3/m</i></p> <p>Where, <i>Rate Runner Group1</i> is the name of the Rate Runner group that has two stages. The desired rate for the first stage is one iteration per second and the rate for the second stage is three iterations per minute.</p> <p> Notes:</p> <ul style="list-style-type: none"> ◦ The name of the Rate Runner group must match with the name in the rate schedule. ◦ The Rate field creates a new copy of the rate schedule that contains the specified number of rate.
Resource Monitoring Labels Override	<p>Use this field to perform any of the following actions:</p> <ul style="list-style-type: none"> ◦ Enable the Resource Monitoring from Service option for a performance schedule if the Resource Monitoring from Service option is not enabled from the schedule editor in Rational® Performance Tester. ◦ Ignore Resource Monitoring sources that were set in the performance schedule and change for a label matching mode. ◦ Replace an existing set of Resource Monitoring labels that were set in the performance schedule and run the schedule with a new set of Resource Monitoring labels. <p>If you have added a label in Rational® Test Automation Server for a Resource Monitoring source as <i>rm1</i>, then you can provide value as shown in the following example to collect data from the source:</p> <p><i>rm1</i></p>

Fields	Description
	<p>If your label for resource monitoring contains a comma (,) then you must replace the single comma with the double comma while providing the value. For example, if you added a label to a Resource Monitoring source as <code>rm1,test</code>, then you must provide the value as shown in the following example to collect data from source:</p> <p><code>rm1,,test</code></p> <p> Notes:</p> <ul style="list-style-type: none"> ◦ You can use this field only when you want to run a Rate schedule or VU schedule. ◦ You can add multiple Resource Monitoring labels separated by a comma. ◦ You must add the Resource Monitoring labels to the Resource Monitoring sources on the Resource Monitoring page in the Rational® Test Automation Server project.
Results File	<p>Use this field to provide a different name to the result file.</p> <p>The results file is stored in the Results directory. The default name of the result file is the name of the <code>test</code> or <code>schedule</code> with a timestamp appended.</p>
User Comments	<p>Use this field to add text that you want to display in the user comments row of the report.</p> <p>For example, <i>test run with dataset</i></p> <p> Note: When you run tests by using the double quotation marks (") for the User Comments field, then the user comments row of a report does not contain double quotation marks.</p> <p>For example, if you provide the value for the User Comments field as <code>test run with "dataset"</code>, then the user comments row of a report displays the value as <code>test run with dataset</code>.</p>

Fields	Description
	 To work around this problem, you must create a command-line config file, and then run the test by using the Config File field.
Var File	Use this field to provide a complete path to an XML file that contains the variable name and value pairs.
VM Args	Use this field to specify the maximum heap size for the Java process. You can add multiple VM Arguments separated by a comma.

- Click **Generate Pipeline Script** and copy the script that you created for the build step to use it in the sample script.

**Note:**

You need this generated script to enter in step [5.c on page 183](#) in the following topic:

Configuring the Pipeline project by using the sample script

Results

You have created the pipeline script.

What to do next

You can configure the **Pipeline** project by using the **Pipeline script** option. See [Configuring the Pipeline project by using the sample script on page 182](#).

Configuring the Pipeline project by using the sample script

You can configure a **Pipeline** project directly from the Jenkins UI by using the sample pipeline scripts to run test assets from Jenkins.

Before you begin

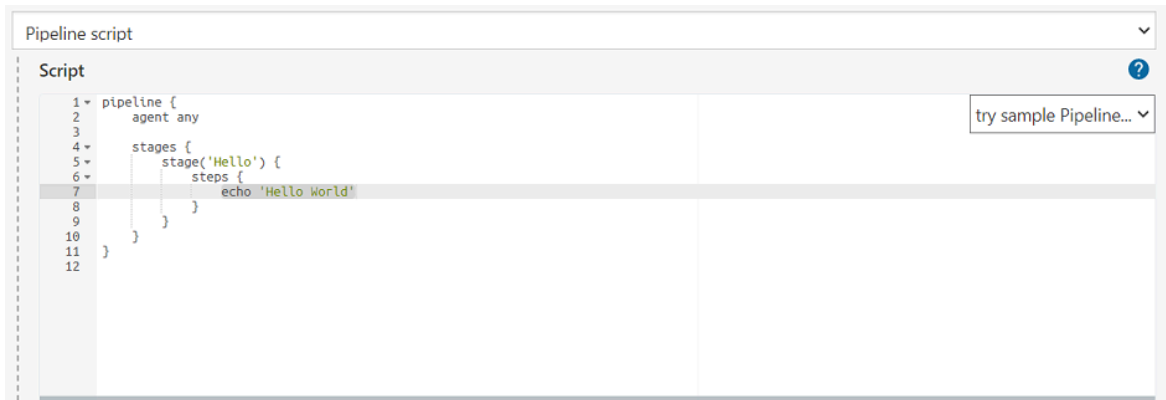
You must have completed the following tasks:

- Installed the Jenkins plugin on the Jenkins primary server. See [Installing the plugin on the Jenkins primary server on page 160](#).
- Created a pipeline script. See [Creating a pipeline script from Jenkins on page 171](#).

- Created an Agent in Jenkins. For more information about creating Agents, refer to the [Jenkins](#) documentation.
- Copied the name of the labels that you provided in the **Labels** field when you created the Agent.

1. Open your Jenkins **Pipeline** project from the list.
2. Click **Configure**, and then select the **Pipeline** tab.
3. Select the **Pipeline script** option in the **Definition** field.
4. Select the sample script from the drop-down list.

For example, if you select `Hello World` from the drop-down list, then the sample script is displayed as follows:



5. Perform the following steps to edit the sample script:
 - a. Replace the agent `any` in the script with the name of the label that you provided when you created the Agent.
For example, if you provided a name of the label as `win10_1022`, then edit the script as follows:


```
agent {label 'win10_1022'}
```
 - b. Provide a name for the `stage` in the script.
For example, `stage('Run Demo')`
 - c. Specify the script that you created in step 7 on page 182 in place of `echo 'Hello World'`.
6. Click **Save**.

Results

You have configured the **Pipeline** project by using the **Pipeline script** option.

What to do next

You can run test assets from the Jenkins server. See [Running tests from Jenkins on page 184](#).

Configuring the Pipeline project by using the script from SCM

You can configure the **Pipeline** project to use the pipeline script from the Source Code Management (SCM), and then run test assets from the Jenkins **Pipeline** project.

Before you begin

You must have completed the following tasks:

- Installed the Jenkins plugin on the Jenkins primary server. See [Installing the plugin on the Jenkins primary server on page 160](#).
- Created an Agent in Jenkins. For more information about creating Agents, refer to the [Jenkins](#) documentation.
- Copied the name of the labels that you provided in the **Labels** field when you created the Agent.
- Created a pipeline script and committed the script to the Git repository. For more information, refer to the [Jenkins](#) documentation.
- Been granted administrator permission of the Jenkins server to add credentials into Jenkins.
- Added your GitHub credentials into Jenkins. For more information about adding global credentials, refer to the [Jenkins](#) documentation.
- Created a Jenkins **Pipeline** project.

About this task

If your pipeline scripts are complex, you can then write the script outside of Jenkins UI. You must commit that script file into your Git repository. During the **Pipeline** build process, Jenkins checks out the script file from the Git repository and runs your **Pipeline** project.

1. Open your Jenkins **Pipeline** project from the list.
2. Click **Configure**, and then select the **Pipeline** tab.
3. Select the **Pipeline script from SCM** option in the **Definition** field.
4. Select the **Git** option from the drop-down list in the **SCM** field.
5. Enter the URL of the Git repository in the **Repository URL** field where you stored your pipeline script.
6. Select the credentials of the Git repository from the drop-down list.
7. **Optional:** Enter the branch name of the Git repository in the **Branch Specifier** field.
8. Enter the path of the script that you stored in the Git repository in the **Script path** field.
9. Click **Save**.

Results

You have configured the **Pipeline** project by using the **Pipeline script from the SCM** option.

What to do next

You can run test assets from the Jenkins server. See [Running tests from Jenkins on page 184](#).

Running tests from Jenkins

You can run test assets either from the Jenkins **Freestyle** or **Pipeline** project on the Jenkins server to test an application under test.

Before you begin

You must have completed the following tasks:

- Verified that you have test assets residing within .
- Configured the **Freestyle** project, if you want to run test assets from the **Freestyle** project. See [Configuring the Freestyle project on page 160](#).
- Configured the **Pipeline** project either by using the **Pipeline script** or **Pipeline script from SCM**, if you want to run test assets from the **Pipeline** project. See [Configuring the Pipeline project by using the sample script on page 182](#) or [Configuring the Pipeline project by using the script from SCM on page 183](#).
- Closed Rational® Performance Tester.

About this task

After you run the test assets from the Jenkins, the **Reports information** section on the **Console Output** page displays the names of the report along with its corresponding URLs if you publish test results to Rational® Test Automation Server.

1. Log in to the Jenkins server.

Result

The Jenkins dashboard is displayed.

2. Open your Jenkins **Freestyle** or **Pipeline** project from the list.
3. Perform one of the following steps to run the test assets from Jenkins:
 - a. Click **Build Now**.
 - b. Click **Build with Parameters** to provide the values for the defined parameters, and then click **Build**.

Results

You have run the test from the Jenkins server.

What to do next

You can view the build logs. To view the build logs, you can click the build number from the **Build History** pane, and then select the **Console Output** option.

Testing with Maven

Starting from 9.2.0, you can use the Maven plug-in that is provided with the testing product to run tests as part of your Maven build. Apache Maven is a software build tool based on the concept of a project object model (POM).

Before you begin

- You must have installed Rational® Performance Tester and set an environment variable that points to the installation location.

For Mac OS, add an environment variable that points to the installation directory of the product: `export TEST_WORKBENCH_HOME=/opt/IBM/SDP`

For Windows™ and Linux®, this environment variable is set when you install the product.

- You must have installed Maven from V3.2.0 and set up an environment variable that points to the `M2_HOME` installation directory.

Introduction

To automate testing with Maven, you must configure a `pom.xml` file and launch your tests from the command line using Maven command. You can either use your own `pom.xml` file, or one that is delivered with the product.

Three files are delivered with the product installation in the `<product install location>\SDP\maven2\` folder:

- `pomCustomSurefireSample.xml` for Windows, Linux and macOS.
- `pomMojoExecPluginSample_Linux.xml` for Linux and MacOS.
- `pomMojoExecPluginSample_Windows.xml` for Windows.

The files contain all types of dependencies as well as arguments required to execute the test scripts. There are two methods to run tests with Maven.

Method 1

With this method, you can run one or several tests. If you use your own `pom.xml` file, edit it with the following lines and indicate which test(s) must be executed, otherwise, use the `pomCustomSurefireSample.xml` file as follows:

- Copy the `pomCustomSurefireSample.xml` to a directory.
- Edit the file and update the lines, enter the name and location of the test(s) that must be run. If the product is installed on a different drive or a different location, or if location has been changed, enter the correct path to the `IBMIMShared` plug-in folder. For `aftsuite` attribute, you can input `aft.xml` file as the parameter value.

```
<!--test suite="testSources/Test1.testsuite"/-->
<!--test suite="Test2.testsuite"/-->
<test suite="C:/Runtimes/runtime-RptMvn/AA/testSources/Test2.testsuite" plugins="C:/Program
Files/IBM/IBMIMShared/plugins"/>
<!--test schedule="Schedule.testsuite" project="AA" workspace="C:/Runtimes/runtime-RptMvn"
plugins="C:/Program Files/IBM/IBMIMShared/plugins"/-->
<!--test suite="Test2.testsuite" project="AA" workspace="C:/Runtimes/runtime-RptMvn"/-->
<!--test aftsuite="Test1.xml" project="AA" workspace="C:/Runtimes/runtime-RptMvn"
plugins="C:/Program Files/IBM/IBMIMShared/plugins"/-->
```

- Run Maven to update the pom file version command and use the plug-in version currently available on delivered repositories.

```
mvn versions:update-properties -Dincludes=com.hcl.products.test.it -f pomCustomSurefireSample.xml
```

- Run the test(s).

```
mvn clean verify -f pomCustomSurefireSample.xml
```

Fail safe reports are generated in the target directory, especially in `target/failsafe-reports/<ProjectName>/<TestName>_<timestamp>.txt` that will contain the screen capture of the execution.

Method 2

With this method, no Maven report is generated. If you use your own pom.xml file, copy the following lines and provide your parameter values. Otherwise, you can use the `pomMojoExecPluginSample_Linux.xml` or `pomMojoExecPluginSample_Windows.xml` sample file. Example with the `pomMojoExecPluginSample_Windows.xml` sample file:

- Copy `pomMojoExecPluginSample_Windows.xml` to a directory.
- Edit the file and update the arguments to reflect which test to execute. If the product is installed on a different drive or a different location, or if `IBMIMShared` location has been changed, update the two last lines with the path to the `IBMIMShared` plug-in folder.

```
<argument>/C</argument>
    <argument>${pt-plugin-cmdline}</argument>
    <argument>-workspace</argument>
    <argument>C:\Runtimes\runtime-RptMvn</argument>
    <argument>-project</argument>
    <argument>AA</argument>
    <argument>-suite</argument>
    <argument>Test1.testsuite</argument>
    <argument>-plugins</argument>
    <argument>C:/Program Files/IBM/IBMIMShared/plugins</argument>
```

- In the argument tags, instead of the `-suite` option, you can use the `-aftsuite` option and input the aft xml file as the parameter value in the subsequent argument tag to run the AFT test. For example, in the preceding template, `<argument>-suite</argument> <argument>Test1.testsuite</argument>` can be replaced with `<argument>-aftsuite</argument> <argument>aftfile.xml</argument>`.
- Run the test.

For Windows:

```
mvn clean verify -f pomMojoExecPluginSample_Windows.xml
```

For Linux or MacOS:

```
mvn clean verify -f pomMojoExecPluginSample_Linux.xml
```

Related information

<https://maven.apache.org/index.html>

Integration with Micro Focus ALM

You can integrate IBM® Rational® Performance Tester with Micro Focus Application Lifecycle Management (ALM) to run performance test assets from Micro Focus ALM.

When you have Micro Focus ALM to manage the life cycle of your application under test, you can create test scripts to run it from Micro Focus ALM. You must use the content of the available template from the installation directory of Rational® Performance Tester to create test scripts. The template is based on Microsoft VBScript and supports VAPI-XP test scripts. You can then run those test scripts from Micro Focus ALM and analyze the test results. For information about Micro Focus ALM, refer to [ALM Help Center](#).

The following table lists the tasks that you must perform to run test assets from Micro Focus ALM:

Tasks	More information
Create any or all of the following types of tests in Rational® Performance Tester to test your application: <ul style="list-style-type: none"> • Compound tests • Performance tests • Schedules (Rate or VU Schedules) 	See Test Author Guide on page 204 .
Install Micro Focus ALM.	For more information about the installation of Micro Focus ALM, refer to the ALM Help Center . For more information about specific versions of software requirements, see System Requirements on page 27 .
Create a test script in Micro Focus ALM.	See Creating a test script in Micro Focus ALM on page 188 .
Copy the content of the template file and configure the test script in Micro Focus ALM.	See Configuring test scripts in Micro Focus ALM on page 189 .
Run performance test assets as test scripts from Micro Focus ALM.	See Running tests from Micro Focus ALM on page 202 .

Creating a test script in Micro Focus ALM

You must create a VAPI-XP-TEST type of test script on Micro Focus Application Lifecycle Management (ALM) to provide the details of the performance test runs.

Before you begin

- You must be familiar with working with Micro Focus ALM
- You must have performed the following tasks:
 - Installed Micro Focus ALM. For more information about installation of Micro Focus ALM, refer to the [ALM Help Center](#).
 - Been granted access to the Micro Focus ALM server.

1. Log in to the Micro Focus ALM portal, if you are not already logged in.

Result

The Micro Focus ALM dashboard is displayed.

2. Create a test by performing the following steps:
 - a. Expand **Testing** from the left pane, and then click **Test Plan**.
 - b. Select a folder from the available list to create a new test.
 - c. Click the **New Test** icon to create a new test.
 - d. Enter a name for the test in the **Test Name** field.
 - e. Select **VAPI-XP-TEST** as test type from the **Type** drop-down list, and then click **OK**.



Note: Rational® Performance Tester does not support other types of tests at this moment.

Result

The **VAPI-XP Wizard** is displayed.

3. Select **VBScript** from the **Script Language** drop-down list.
4. Enter a name for the script in the **Script Name** field.

The default name of the script is entered as *script*. You can change it by entering a new name.

5. Click **Next**, and then select **COM/DCOM Server Test** as a test type.
6. Click **Finish**.

Results

You have created the VAPI-XP-TEST test script in Micro Focus ALM.

What to do next

You must configure the test script to add the required parameter values of performance test. See [Configuring test scripts in Micro Focus ALM on page 189](#).

Configuring test scripts in Micro Focus ALM

You must configure the test script that you created in Micro Focus Application Lifecycle Management (ALM) to run the performance test assets.

Before you begin

You must have performed the following tasks:

- Created a test script in Micro Focus ALM. See [Creating a test script in Micro Focus ALM on page 188](#).
- Copied the content of the template file.

About this task

You can navigate to `alm` directory to copy the content of the template file. The `alm` directory resides within the installation directory of Rational® Performance Tester. The name of the template file is `PT_ALM_Windows.txt` and you can access the file from the following location:

```
Installation_dir\IBM\SDP\alm
```

For example, `C:\Program Files\IBM\SDP\alm`

You must provide the values for certain parameters in the test script to run tests from Micro Focus ALM. The minimum parameters that you need to configure are, **Workspace**, **Project**, and **TestsuiteName** or **ConfigFile**. The remaining parameters are optional.

For example, if you provide a complete path of a configuration file that contains the parameters for a `test Or schedule` run in the **Config File** parameter, then you need not be required to provide values for the **Workspace**, **Project**, and **TestsuiteName** parameters.



Note: If Rational® Performance Tester is not installed in default location, then you must provide the value for the **IMSharedLocation** parameter along with other mandatory parameters.



Remember: You must enter the parameter values within the double quotation marks.

1. Log in to the Micro Focus ALM portal, if you are not already logged in.

Result

The Micro Focus ALM dashboard is displayed.


2. Expand **Testing** from the left pane, and then click **Test Plan**.
3. Select a test script from the folder that you want to configure.
4. Click the **Test Script** tab.
5. Paste the content of the `PT_ALM_Windows.txt` file that you copied in the space provided in the **Test Script** tab.
6. Enter the parameter values that are required for your test run in the script by referring to the following table:


The following table lists the required parameters that you must provide to run the test from Micro Focus ALM:


Parameters	Description
ConfigFile	<p>Enter the complete path to a configuration file that contains the parameters for a <code>test</code> or <code>schedule run</code>.</p> <p>For example,</p> <pre>ConfigFile = "C:\Users\IBM\rational\workspace\Project1\config.txt"</pre> <p>For more information about creating a command line config file from Rational® Performance Tester, see the Related information section.</p>
Project	<p>Enter the name of the project that has performance test assets.</p> <p>For example,</p> <pre>Workspace = "C:/Users/IBM/rational\workspace1" Project = "proj1" TestSuiteName = "Tests/testHttp.testsuite"</pre>
TestsuiteName	<p>Enter the name of the test asset that you want to run.</p> <p>For example,</p> <pre>Workspace = "C:/Users/IBM/rational\workspace1" Project = "proj1" TestSuiteName = "Tests/testHttp.testsuite"</pre>
Workspace	<p>Enter the complete path of the Eclipse workspace.</p> <p>For example,</p> <pre>Workspace = "C:/Users/IBM/rational\workspace1" Project = "proj1" TestSuiteName = "Tests/testHttp.testsuite"</pre>


The following table lists the optional parameters that you can provide to run the test from Micro Focus ALM:


Parameters	Descriptions
DatasetOverride	<p>Use this parameter to replace the dataset values during a <code>test</code> or <code>schedule run</code>.</p> <p>You must ensure that both original and new datasets are in the same workspace and have the same column names. When you enter a value for the Dataset Override field, you must also include the path to the dataset. You must provide the values for the Dataset Override field in the following format:</p> <pre>/project_name/ds_path/original_ds.csv:/project_name/ds_- path/new_ds.csv</pre>


Parameters	Descriptions
	<p>For example,</p> <pre>Workspace = "C:/Users/IBM/rationalmdp/workspace1" Project = "proj1" TestSuiteName = "Tests/testHttp.testsuite" DatasetOverride = "/project_name/ds_path/ds_filename.csv:/project_name/ds_path/new_ds_file name.csv"</pre> <p>You can override multiple datasets that are saved in a different project by adding multiple paths to the dataset separated by a semicolon. For a <code>test</code> or <code>schedule</code>, the default value is the dataset specified in the test editor or schedule editor.</p>
Duration	<p>Use this parameter to change the duration of the stages in the rate schedule.</p> <p>For example,</p> <pre>Workspace = "C:/Users/IBM/rationalmdp/workspace1" Project = "proj1" TestSuiteName = "Schedule/ratesch.testsuite" Duration = "Stage1=10s, Stage2=3m"</pre> <p>The stage number specified must exist in the rate schedule.</p> <p> Note: The Duration parameter creates a new copy of the rate schedule that contains the specified number of duration.</p>
Exportlog	<p>Use this parameter to specify the path of the file directory to store the exported HTTP test log in a <code>.txt</code> format.</p> <p>For example,</p> <pre>Workspace = "C:/Users/IBM/rationalmdp/workspace1" Project = "proj1" TestSuiteName = "Tests/testHttp.testsuite" Exportlog = "C:\Users\Documents\1023\ALM\LogExp.txt"</pre>
ExportStatReportlist	<p>Use this parameter to specify IDs of reports that you want to export in place of the default reports.</p> <p>You can provide multiple report IDs separated by a comma. You can navigate to Preferences of Rational® Performance Tester (Window > Preferences > Test > Performance Test Reports > Export Reports), and then select Show Report IDs checkbox to view the report IDs.</p>

Parameters	Descriptions
	<p>You must use the ExportStatReportlist parameter along with the ExportStatsFile or ExportStatsHtml parameter.</p> <p>For example, to export a Performance Report, you can provide the value for the ExportStatReportlist parameter as follows:</p> <pre data-bbox="565 457 1133 604">Workspace = "C:/Users/IBM/rational_sdp/workspace1" Project = "proj1" TestSuiteName = "Tests/testHttp.testsuite" ExportStatsFile = "C:\Users\Documents\ALM" ExportStatReportlist = "http"</pre>
ExportStatsFile	<p>Use this parameter to provide the complete path to a directory that you can use to store the exported report in a comma-separated values (CSV) format.</p> <p>For example,</p> <pre data-bbox="565 810 1133 919">Workspace = "C:/Users/IBM/rational_sdp/workspace1" Project = "proj1" TestSuiteName = "Tests/testHttp.testsuite" ExportStatsFile = "C:\Users\Documents\ALM"</pre> <p> Note: If you do not specify a value for the ExportStatReportlist parameter, then the reports that you selected in Preferences of Rational® Performance Tester (Window > Preferences > Test > Performance Test Reports > Export Reports) are exported.</p>
Exportstatsformat	<p>Use this parameter to specify a format for the result that you want to export along with the ExportStatsFile parameter.</p> <p>You must use at least one of the following options with the Exportstatsformat parameter:</p> <ul style="list-style-type: none"> ◦ simple.csv ◦ full.csv ◦ simple.json ◦ full.json ◦ csv ◦ json <p>For example,</p> <pre data-bbox="565 1724 1133 1860">Workspace = "C:/Users/IBM/rational_sdp/workspace1" Project = "proj1" TestSuiteName = "Tests/testHttp.testsuite" ExportStatsFile = "C:\Users\Documents\ALM" Exportstatsformat = "simple.json"</pre>



Parameters	Descriptions
	<p>You can add multiple options separated by a comma.</p> <p>For example,</p> <pre data-bbox="565 380 1133 520">Workspace = "C:/Users/IBM/rationalsdp/workspace1" Project = "proj1" TestSuiteName = "Tests/testHttp.testsuite" ExportStatsFile = "C:\Users\Documents\ALM" Exportstatsformat = "simple.json,full.csv"</pre> <p>When you want to export both simple and full type of test results in a <code>json</code> or <code>csv</code> format, you can specify either <code>json</code> or <code>csv</code> as options.</p> <p>For example,</p> <pre data-bbox="565 716 1133 856">Workspace = "C:/Users/IBM/rationalsdp/workspace1" Project = "proj1" TestSuiteName = "Tests/testHttp.testsuite" ExportStatsFile = "C:\Users\Documents\ALM" Exportstatsformat = "json"</pre> <p>When the test run completes, the test result exports to <code>simple.json</code> and <code>full.json</code> files.</p> <p> Note: The values provided in the test script always take precedence over the Export Reports options set in the Preferences of Rational® Performance Tester (Window > Preferences > Test > Performance Test Reports > Export Reports).</p>
<p>ExportStatsHtml</p>	<p>Use this parameter to provide the complete path to a directory that you can use to export web analytic results.</p> <p>The results are exported to the specified directory. You can analyze the results on a web browser without using Rational® Performance Tester.</p> <p>For example,</p> <pre data-bbox="565 1507 1133 1612">Workspace = "C:/Users/IBM/rationalsdp/workspace1" Project = "proj1" TestSuiteName = "Tests/testHttp.testsuite" ExportStatsHtml = "C:/Users/Documents/Reports"</pre>
<p>IMSharedLocation</p>	<p>Use this parameter to enter the complete path to the IBMIMShared location, if it is not in the default location.</p> <p>The default location of IBMIMShared is <code>C:\Program Files\IBM\IB-MIMShared</code>.</p>


Parameters	Descriptions
	<p>For example,</p> <pre>Workspace = "C:/Users/IBM/rational\sdp/workspace1" Project = "proj1" TestSuiteName = "Tests/testHttp.testsuite" IMSharedLocation = "D:\Testtool\IBM\SDP"</pre>
<p>Labels</p>	<p>Use this parameter to add labels to test results when the test run is complete.</p> <p>For example,</p> <pre>Workspace = "C:/Users/IBM/rational\sdp/workspace1" Project = "proj1" TestSuiteName = "Tests/testHttp.testsuite" Labels = "label1, label2"</pre> <p>You can add multiple labels to a test result separated by a comma. When you run test assets, then the labels that you added are displayed on the Performance Report in Rational® Performance Tester.</p> <p>The Results page of Rational® Test Automation Server displays the label that you added in the Labels field for the specific test asset in the following conditions:</p> <ul style="list-style-type: none"> ◦ When you set Publish result after execution as Always or Prompt in the Preferences of Rational® Performance Tester (Windows > Preferences > Test > Rational Test Automation Server). ◦ When you use the Labels field along with the Publish field. <p> Note: When you run tests by using the double quotation marks ("") for the Labels parameter, then the labels in the test result do not include double quotation marks.</p> <p>For example, if you provide the value for the Labels parameter as "100 users", then the labels display 100 users in the test result.</p> <p>To work around this problem, you must create a command-line config file, and then run the test by using the ConfigFile parameter.</p>
<p>OverwriteResultsFile</p>	<p>Set this parameter value to <code>true</code> or <code>false</code> to determine whether a result file with the same name must be overwritten or not.</p> <p>The default value is <code>true</code>.</p> <p>For example,</p>



Parameters	Descriptions
	<pre>Workspace = "C:/Users/IBM/rationalmdp/workspace1" Project = "proj1" TestSuiteName = "Tests/testHttp.testsuite" OverwriteResultsFile = "false"</pre>
Publish	<p>Use this parameter to publish test results to Rational® Test Automation Server from Micro Focus ALM.</p> <p>You must provide the offline user token in the Preferences of Rational® Performance Tester (Window > Preferences > Test > Rational Test Automation Server) before you use the Publish parameter in the test script.</p> <p>You must use one of the following values in the Publish parameter:</p> <ul style="list-style-type: none"> ◦ <i>serverURL#project.name=name_of_the_project&team.space.name=name_of_the_team.space</i> ◦ <i>serverURL#project.name=name_of_the_project&team.space.alias=name_of_the_team.space_alias</i> ◦ no <p>The no option is useful if the product preferences are set to publish the results, but you do not want to publish them.</p> <p>For example,</p> <pre>Workspace = "C:/Users/IBM/rationalmdp/workspace1" Project = "proj1" TestSuiteName = "Tests/testHttp.testsuite" Publish = "URL_OF_RationalTest_Server#project.name=test&team.space.name=ts1"</pre> <pre>Workspace = "C:/Users/IBM/rationalmdp/workspace1" Project = "proj1" TestSuiteName = "Tests/testHttp.testsuite" Publish = "no"</pre> <p>Where:</p> <ul style="list-style-type: none"> ◦ <i>URL_OF_RationalTest_Server</i> is the URL of Rational® Test Automation Server. ◦ <i>test</i> is the name of the project in Rational® Test Automation Server. ◦ <i>ts1</i> is the name of the team space. <p> Notes:</p>


Parameters	Descriptions
	 <ul style="list-style-type: none"> ◦ If you have a project with the same name in different team spaces, then you must append either the &teamspace.name=name_of_the_teamspace or &teamspace.alias=name_of_the_teamspace_alias options. ◦ If the name of the project or team space contains a special character, then you must replace it with %<Hexvalue_of_special_character>. For example, if the name of the team space is <i>Initial Team Space</i>, then you must provide it as <i>Intial%20Team%20Space</i>. Where %20 is the Hex value of <i>space</i> character. ◦ The values provided in the Publish field always take precedence over the Results options set in the product preferences (Window > Preferences > Test > Rational Test Automation Server > Results). <p>The Reports information section on the Output window displays the names of the report along with its corresponding URLs in the following conditions:</p> <ul style="list-style-type: none"> ◦ When you configured the URL of in Preferences of Rational® Performance Tester (Window > Preferences > Test > Rational Test Automation Server). ◦ When you set Publish result after execution as Always or Prompt in the Preferences of Rational® Performance Tester (Window > Preferences > Test > Rational Test Automation Server > Results). ◦ When you used Publish parameter or Publishreports parameter.
Publish_for	<p>Use this parameter to publish test results to Rational® Test Automation Server from Micro Focus ALM based on the completion status of the tests.</p> <p>You must use the Publish_for parameter along with the Publish parameter. You must add multiple parameters separated by a comma. The available options that you can use for the Publish_for parameter are as follows:</p> <ul style="list-style-type: none"> ◦ ALL: You can use this option irrespective of status of the test. ◦ PASS: You can use this option to publish test results for the tests that have passed. ◦ FAIL: You can use this option to publish test results for the tests that have failed.

Parameters	Descriptions
	<ul style="list-style-type: none"> ◦ ERROR: You can use this option to publish test results for the tests that included errors. ◦ INCONCLUSIVE: You can use this option to publish test results for the inconclusive tests. <p>For example,</p> <pre>Workspace = "C:/Users/IBM/rationalmdp/workspace1" Project = "proj1" TestSuiteName = "Tests/testHttp.testsuite" Publish = "URL_OF_RationalTest_Server#project.nam=test&teamspace.name=ts1" Publish_for = "FAIL,ERROR"</pre>
Publishreports	<p>Use this parameter to publish specific test results to Rational® Test Automation Server.</p> <p>You must use the Publishreports parameter along with the Publish parameter. The options that you can use with Publishreports are as follows:</p> <ul style="list-style-type: none"> ◦ STATS ◦ TESTLOG <p>For example,</p> <pre>Workspace = "C:/Users/IBM/rationalmdp/workspace1" Project = "proj1" TestSuiteName = "Tests/testHttp.testsuite" Publish = "URL_OF_RationalTest_Server#project.nam=test&teamspace.name=ts1" Publishreports = "STATS"</pre> <p>You can prefix the Publishreports value with ! to publish the reports except for the specified one in the test script.</p> <p>For example,</p> <pre>Workspace = "C:/Users/IBM/rationalmdp/workspace1" Project = "proj1" TestSuiteName = "Tests/testHttp.testsuite" Publish = "URL_OF_RationalTest_Server#project.nam=test&teamspace.name=ts1" Publishreports = " !TESTLOG"</pre>
Rate	<p>Use this parameter to change the rate of the rate runner group.</p> <p>For example,</p>

Parameters	Descriptions
	<pre>Workspace = "C:/Users/IBM/rationalmdp/workspace1" Project = "proj1" TestSuiteName = "Schedule/ratesch.testsuite" Rate = "Rate Runner Group1=1/s, 3/m"</pre> <p>Where, <code>Rate Runner Group1</code> is the name of the rate runner group that has two stages. The desired rate for the first stage is one iteration per second and the rate for the second stage is three iterations per minute.</p> <p> Notes:</p> <ul style="list-style-type: none"> ◦ The name of the Rate Runner group must match with the name in the rate schedule. ◦ The Rate parameter creates a new copy of the rate schedule that contains the specified number of rate.
Reporthistory	<p>Use this parameter when you want to view a record of all events that occurred during a <code>test</code> or <code>schedule</code> run.</p> <p>You must use any of the following options:</p> <ul style="list-style-type: none"> ◦ jaeger: To send test logs to the Jaeger UI during the test or schedule run. <p> Note: You must set the <code>JAEGER_AGENT_HOST</code> property as System environment variable before you use the jaeger option to send test logs to the Jaeger UI.</p> <ul style="list-style-type: none"> ◦ testlog: To send test logs as traditional test logs in Rational® Performance Tester during the <code>test</code> or <code>schedule</code> run. ◦ null: To send no test logs either to the Jaeger UI or Rational® Performance Tester during the <code>test</code> or <code>schedule</code> run. <p>For example,</p> <pre>Workspace = "C:/Users/IBM/rationalmdp/workspace1" Project = "proj1" TestSuiteName = "Tests/testHttp.testsuite" Reporthistory = "jaeger"</pre> <p>You can add multiple options separated by a comma to send test logs during the <code>test</code> or <code>schedule</code> run to Rational® Performance Tester and the Jaeger UI.</p> <p>For example,</p> <pre>Workspace = "C:/Users/IBM/rationalmdp/workspace1" Project = "proj1"</pre>

Parameters	Descriptions
	<pre data-bbox="565 260 1052 317">TestSuiteName = "Tests/testHttp.testsuite" Reporthistory = "jaeger,testlog"</pre> <p data-bbox="545 348 1403 422">For more information about how to view test logs in the Jaeger UI and Rational® Performance Tester, see the Related information section.</p>
ResourceMonitoring-LabelsOverride	<p data-bbox="545 485 1170 516">Use this parameter to perform any of the following actions:</p> <ul data-bbox="610 548 1403 905" style="list-style-type: none"> ◦ Enable the Resource Monitoring from Service option for a performance <code>schedule</code> if the Resource Monitoring from Service option is not enabled from the schedule editor in Rational® Performance Tester. ◦ Ignore Resource Monitoring sources that were set in the performance schedule and change for a label matching mode. ◦ Replace an existing set of Resource Monitoring labels that were set in the performance schedule and run the schedule with a new set of Resource Monitoring labels. <p data-bbox="545 947 1422 1052">If you have added a label in Rational® Test Automation Server for a Resource Monitoring source as <code>rm1</code>, then you can provide value as shown in the following example to collect data from the source:</p> <pre data-bbox="565 1079 1133 1188">Workspace = "C:/Users/IBM/rationalldap/workspace1" Project = "proj1" TestSuiteName = "Tests/testHttp.testsuite" ResourceMonitoringLabelsOverride = "rm1"</pre> <p data-bbox="545 1230 1403 1377">If your label for resource monitoring contains a comma (,) then you must replace the single comma with the double comma while providing the value. For example, if you added a label to a Resource Monitoring source as <code>rm1,test</code>, then you must provide the value as shown in the following example to collect data from source:</p> <pre data-bbox="565 1404 1133 1514">Workspace = "C:/Users/IBM/rationalldap/workspace1" Project = "proj1" TestSuiteName = "Tests/testHttp.testsuite" ResourceMonitoringLabelsOverride = "rm1,,test"</pre> <p data-bbox="565 1556 691 1591"> Notes:</p>

Parameters	Descriptions
	 <ul style="list-style-type: none"> ◦ You can use this field only when you want to run a Rate schedule or VU schedule. ◦ You can add multiple Resource Monitoring labels separated by a comma. ◦ You must add the Resource Monitoring labels to the Resource Monitoring sources on the Resource Monitoring page in the Rational® Test Automation Server project.
ResultsFile	<p>Use this parameter to provide a different name to the result file.</p> <p>The default name of the result file is the name of the <code>test</code> or <code>schedule</code> with a time-stamp appended.</p>
UserComments	<p>Use this parameter to add text that you want to display in the user comments row of the report.</p> <p>For example,</p> <pre style="background-color: #f0f0f0; padding: 5px;">Workspace = "C:/Users/IBM/rationalsdp/workspace1" Project = "proj1" UserComments = "test run with dataset"</pre> <p> Note: When you run tests by using the double quotation marks ("") for the UserComments parameter, then the user comments row of a report does not contain double quotation marks.</p> <p>For example, if you provide the value for the UserComments parameter as <code>test run with "dataset"</code>, then the user comments row of a report displays the value as <code>test run with dataset</code>.</p> <p>To work around this problem, you must create a command-line config file, and then run the test by using the ConfigFile parameter.</p>
Users	<p>Use this parameter to overrides the default number of users, if required.</p> <p>For a <code>schedule</code>, the default is the number of users specified in the schedule editor and for a <code>test</code>, the default is one user.</p>

Parameters	Descriptions
	 Note: The Users parameter creates a new copy of the <code>schedule</code> that contains the specified number of users.
VarFile	Use this parameter to provide a complete path to an XML file that contains the variable name and value pairs.
VMArgs	Use this parameter to specify the maximum heap size for the Java process. You can add multiple VM Arguments separated by a comma.

7. Click the **Save** icon.

Results

You have configured the test script by adding the required parameter values for the test run.

What to do next

You can run test assets from Micro Focus ALM. See [Running tests from Micro Focus ALM on page 202](#).

Related information

[Creating a command-line config file on page 370](#)

[Viewing test logs in Jaeger on page 156](#)

[Viewing test logs on page 467](#)

[Generating offline user tokens](#)

Running tests from Micro Focus ALM

You can run test assets from Micro Focus Application Lifecycle Management (ALM) to test the performance of your application under test.

Before you begin

You must have completed the following tasks:

- Created a test script in Micro Focus ALM. See [Creating a test script in Micro Focus ALM on page 188](#).
- Configured the test script in Micro Focus ALM. See [Configuring test scripts in Micro Focus ALM on page 189](#).

- **Optional:** Generated offline user token to publish test results to Rational® Test Automation Server. For more information refer to [Managing access to the server](#) in the Rational® Test Automation Server Documentation.
- Closed Rational® Performance Tester.

1. Log in to the Micro Focus ALM portal, if you are not already logged in.

Result

The Micro Focus ALM dashboard is displayed.

2. Expand **Testing** from the left pane, and then click **Test Plan**.
3. Open the test script that you want to run.
4. Click the **Test Script** tab.

Result

The test script is displayed.

5. Click the **Execute Script** icon to run the test script.

Results

You have run the performance test scripts in Micro Focus ALM.

What to do next

You can view the test result details that are displayed in the **Output** window of Micro Focus ALM.

The **Reports information** section on the **Output** window displays the names of the report along with its corresponding URLs if you publish test results to Rational® Test Automation Server.

Chapter 6. Test Author Guide

This guide describes how to create test scripts in Rational® Service Tester for SOA Quality and enhances tests by applying different test elements such as dataset, variables, and verification points. This guide is intended for testers.

Creating tests

To create a test, you record representative interactions with an application.

After you record a test, you can play it back to confirm that the recorded actions do what you expect.



Note: When you record a test that includes a file download, the file is not physically saved to disk. However, you can confirm that the file was retrieved from the server by looking in the response of the request that asked for the file. One method to locate the request for large downloaded files is to look for a request with a large response size.

Creating a project

The tests that you create, and the assets associated with the tests, reside in a project on your desktop. You can create the project separately, or you can simply record a test, which automatically creates a project named `testproj`.

1. Select .

Result

The **Create a Project** window opens.

2. In the **Project Name** field, enter a name for the project.
3. Select **Use default location**.
4. **Optional:** Click **Next** and select the folders to create in the new project.
These folders organize your files by asset (`Tests`, `Results`, and so on).
5. Click **Finish**.

Result

After you click finish, you are prompted to record a test.

What to do next

You can create a test from a new recording or from an existing recording, or you can click **Cancel** to create a test project without recording a test.

Recording service tests

When you record a test, the test creation wizard records your interactions with the service, generates a test from the recording, and opens the test for editing. You can record a test session by invoking service calls with the generic service client or by using an existing client. You can also create a service test manually or from a Business Process Execution Language (BPEL) model.

Service testing guidelines

Before you can test a service, you must set up your test environment and incorporate these guidelines in order to produce reliable tests.

Test prerequisites

Before creating service tests, you might need to perform some initial tasks. These tasks depend on the transport and security protocols that are implemented by the web service under test.

- **HTTP:** This transport method is checkmark_perf.jpg by default; no additional configuration is required.
- **SSL:** The workspace must contain the certificate keystore (JKS) files that are required for single or double authentication.
- **Java™ Message Service (JMS):** The Web Services Description Language (WSDL) syntax must be compatible with the requirements of the product. Refer to [Verifying WSDL syntax compliance for JMS services on page 207](#).

Test generation

When the test is generated, message call envelopes are created according to the XML schema definition (XSD). During this process, mandatory fields are created, and default choices are assumed. You can modify these elements in the test editor.



Note: During recording, you might supply authentication details which are not relevant for the actual application under test. To exclude such actions from the generated test, in **Window > Preferences > Test > Test editor > Service test** ensure that the **Display the 'Skip if Empty' column in XML tree viewer** check box is selected. To select the empty XML elements that you want to skip, in the test editor, select the elements in the **Skip if empty** column.

Encryption and security

The Java™ Runtime Environment (JRE) that the product uses must support the level of encryption required by the digital certificate that you select. For example, you cannot use a digital certificate that requires 256-bit encryption with a JRE that supports only 128-bit encryption. By default, the product is configured with restricted or limited strength ciphers. To use less restricted encryption algorithms, you must download and apply the unlimited jurisdiction policy files (`local_policy.jar` and `US_export_policy.jar`).

For Oracle Java, download the files from this site: <http://www.oracle.com/technetwork/java/javase/downloads/jce8-download-2133166.html>.

Before installing these policy files, back up the existing policy files in case you want to restore the original files later. Then overwrite the files in `/jre/lib/security/` directory with the unlimited jurisdiction policy files.

SSL Authentication

Service tests support simple or double SSL authentication mechanisms:

- Simple authentication (server authentication): In this case, the test client needs to determine whether the service can be trusted. You do not need to setup a key store. If you select the **Always trust** option, you do not need to provide a server certificate key store.

If you want to really authenticate the service, you can configure a certificate trust store, which contains the certificates of trusted services. In this case, the test will expect to receive a valid certificate.

- Double authentication (client and server authentication): In this case, the service needs to authenticate the test client according to its root authority. You must provide the client certificate keystore that needs to be produced to authenticate the test as a certified client.

When recording a service test through a proxy, the recording proxy sits between the service and the client. In this case, you must configure the SSL settings of the recording proxy to authenticate itself as the actual service to the client (for simple authentication), and as the client to the service (for double authentication). This means that you must supply the recording proxy with the adequate certificates.

When using stub services, you can also configure the SSL settings of the stub service to authenticate itself as the actual server. This means that you must supply the service stub with the adequate certificate.

NTLM and Kerberos Authentication

The product supports Microsoft™ NT LAN Manager (NTLMv1 and NTLMv2) and Kerberos authentication. The authentication information is recorded as part of the test during the recording phase.

To enable NTLMv2 support, you must add a third party library to the workbench. For more information, see [Configuring the workbench for NTLMv2 authentication on page 341](#).

Digital certificates

You can test services with digital certificates for both SSL and SOAP security protocol. Digital certificates must be contained in Java™ Key Store (JKS) keystore resources that are accessible in the workspace. When dealing with keystore files, you must set the password required to access the keys both in the security editor and the test editor. For SOAP security you might have to provide an explicit name for the key and provide a password to access the private keys in the keystore.

Limitations

Arrays are not checkmark_perf.jpg.

Because of a lack of specification, attachments are not checkmark_perf.jpg with the Java™ Message Service (JMS) transport. The envelope is directly sent using UTF-8 encoding.

All security algorithms are not always available for every Java™ Runtime Environment (JRE) implementation. If a particular security implementation is not available, add the required libraries to the class path of the JRE that this product uses.

The generic service tester displays the envelope as reflected in the XML document. However, security algorithms consider the envelope as a binary. Therefore, you must set up the SOAP security configuration so that incoming and outgoing messages are correctly encrypted but remain decrypted inside the test.

Performance

Virtual user performance depends on the implementation of the container application. For an HTTP transport, the product has been tested with a maximum of 900 concurrent virtual users under Windows™ and 600 under Linux™. For JMS, the maximum is 100 concurrent virtual users, although this number can vary due to the asynchronous implementation of JMS. Beyond these values, connection errors might occur and the transaction rate will decrease.

Verifying WSDL syntax compliance for JMS services

Various Java™ Message Service (JMS) providers vary in the syntax used for describing services. Before testing JMS services, you must ensure that Web Services Description Language (WSDL) files comply with the requirements of the tool.

1. In the project explorer or test explorer, locate and open the WSDL file for the JMS service that you want to test. If necessary, you can import a WSDL file from the file system by clicking **File > Import > File System**.
2. Ensure that the following criteria are met in the syntax of the WSDL file that you use.

- Namespace: `xmlns:jms="http://schemas.xmlsoap.org/wsdl/jms/"`
- SOAP bindings are set to: `transport="http://schemas.xmlsoap.org/soap/jms"`
- JMS transports are defined either as a URL or as `jms:address` element

3. If the WSDL file is not compliant, edit the file so that it meets the criteria, and then save and close the file.

Example

For example, a JMS defined as a URL looks like this:

```
<soap:address location="jms:/queue?jndiConnectionFactoryName=UIL2ConnectionFactory;
  jndiDestinationName=queue/testQueue;
  initialContextFactory=org.jnp.interfaces.NamingContextFactory;
  jndiProviderURL=9.143.104.47"/>
```

A JMS defined as an address looks like this:

```
<jms:address destinationStyle="queue"
  jndiConnectionFactoryName="myQCF"
  jndiDestinationName="myQ"
  initialContextFactory="com.ibm.NamingFactory"
  jndiProviderURL="iiop://something:900/">
</jms:address>
```

Configuring the environment for SOAP security

SOAP security profiles require access to the libraries that implement encryption, signature, and other security algorithms that transform the XML messages before sending and after receiving them. You must prepare an environment with these libraries to use SOAP security, set the class path of the Java™ Runtime Environment (JRE) that Eclipse uses, and set the class path of the virtual machine that the Agent Controller uses.

Before you begin

Before you can test SOAP-based services that use security algorithms, you must obtain a set of security libraries and configuration files for SOAP.

BouncyCastle (<http://www.bouncycastle.org>) is a provider of such security libraries. Use of these security libraries is optional for the Rational® test product.

1. Copy the library files into the `jre/lib/ext` of the JRE installation.

By default, this is the following directory: `C:\Program Files\IBM\SDP\jdk\jre\lib\ext`

2. Add the following VM argument either to the Eclipse launch command line or to the `eclipse.ini` file:

```
-vmargs-Dosgi.parentClassLoader=ext
```

The `eclipse.ini` file is located in the same directory as the `eclipse.exe` launcher binary that is used to run the product.

What to do next

To configure a remote computer that uses only the Agent Controller and does not require access to the workbench, perform only step 1 and restart the Agent Controller service.

After configuring the environment, you must import a Web Services Description Language (WSDL) file and use the **WSDL security editor** to set up a security profile for the WSDL file.

Recording a service test with the generic service client

You can record a service test by invoking service requests with the generic service client. After you have sent the requests and received the responses from the service, select the results in the History section of the generic service client to generate a test. If you do not have access to a dedicated client for the service calls, the generic service client is the easiest way to generate the calls and to record a test.

Before you begin

If you are testing a SOAP-based web service, ensure that you have access to a valid Web Services Description Language (WSDL) file. The wizard can import WSDL files from the workspace, the file system, a remote repository, or from a URL. Ensure that the WSDL files use the correct syntax for the test environment. The generic service client might not work with some WSDL files.

If you are using Secure Sockets Layer (SSL) authentication, ensure that you have the required key files in your workspace.

If you are using SOAP security, ensure that you have configured the environment with the correct libraries and configuration files. See [Configuring the environment for SOAP security on page 208](#) for more information.

If the response in a recording or test generation is in XML and the size of the XML data is higher than the value set in the **XML Message Received maximum length** field, the response is automatically converted to text to avoid any memory issues. To convert the full response to text, the tool checks the value set for **Text Message Received maximum length**. If the value is lesser than the size of the response, the response is truncated. If you want the response to be in XML when the response size exceeds the value set in **XML Message Received maximum length**, you can manually increase the value for both recording and test generation. To change the value for recording, click **Window > Preferences > Generic Service Client > Message Edition**. To change the value for test generation, click **Window > Preferences > Test > Test Generation > Service Test Generation**.

About this task



You can record and generate a test by using REST APIs. The API documentation to record a test is located at `Install_directory\IBM\IBMIShared\plugins\com.ibm.rational.test.lt.server.recorder.jar`. The API documentation to generate a test after the recording completes is located at `C:\Program Files\IBM\IBMIShared\plugins\com.ibm.rational.test.lt.server.testgen.jar`.

To use a WS-SecurityPolicy that is included in a WSDL or an external XML file, you need to configure the security policy as described in [Using a security policy on page 259](#). If a recording contains the Security Assertion Markup Language (SAML) token, the WS Security policy file must rely on the Service Token Service (STS) that produces the token. This token can then be used for encryption or other purposes as was designed.

Sample policy file that relies on SAML token:




```
<sp:SupportingTokens xmlns:sp="http://schemas.xmlsoap.org/ws/2005/07/securitypolicy">
  <wsp:Policy>
    <sp:IssuedToken
      sp:IncludeToken="http://schemas.xmlsoap.org/ws/2005/07/securitypolicy/IncludeToken/AlwaysToRecipient">
      <sp:Issuer>
        <Address
          xmlns="http://www.w3.org/2005/08/addressing">http://9.143.105.204:8080/axis2/services/STS</Address>
        </sp:Issuer>
        <sp:RequestSecurityTokenTemplate>
          <t:TokenType
            xmlns:t="http://schemas.xmlsoap.org/ws/2005/02/trust">http://
docs.oasis-open.org/wss/oasis-wss-saml-token-profile-1.1#SAMLV2.0</t:TokenType>
          <t:KeyType
            xmlns:t="http://schemas.xmlsoap.org/ws/2005/02/trust">http://
schemas.xmlsoap.org/ws/2005/02/trust/SymmetricKey</t:KeyType>
          <t:KeySize xmlns:t="http://schemas.xmlsoap.org/ws/2005/02/trust">256</t:KeySize>
          </sp:RequestSecurityTokenTemplate>
        </wsp:Policy>
        <sp:RequireInternalReference/>
      </wsp:Policy>
    </sp:IssuedToken>
```

```
</wsp:Policy>
</sp:SupportingTokens>
```

1. In the Performance Test perspective, click the **New Test from Recording** toolbar button  or click **File > New > Test from Recording**.
2. In the **New Test from Recording** wizard, click **Create a test from a new recording**, select **Service Test**, and click **Next**.
If you are recording sensitive data, you can select a **Recording encryption level**.
3. On the **Select Location** page, select the project and folder where you want to create the test, type a name for the test, and click **Next**.
If necessary, click **Create Parent Folder**  to create a project or folder.
4. On the **Select Location** page, select **Generic Service Client**.
This option uses the generic service client if you do not have access to a dedicated client for the service calls. See [Recording a service test through a client program on page 211](#) for information about using other client programs to record the test.
5. Click **Next**. If this is the first time you are recording a web service test, read the Privacy Warning, select **Accept**, and click **Finish** to proceed.

Result


The generic service client opens.

6. If your service uses a transport or authentication protocol that requires overriding the default settings, then click the **Transport** tab and create an HTTP, Java™ Message Service (JMS), IBM® WebSphere® MQ, IBM® WebSphere® Java MQ, or Java MQ transport.
7. Click the **Requests** tab.
Choose from:
 - Right-click **WSDLs**  and select one of the options to get the WSDL file.
 - Right-click **WADLs**  and select one of the options to get the WADL file.
 - Right-click **Endpoints**  and select one of the options to send the request.

See [Sending service requests with the generic service client on page 338](#) for more information about using the generic service client.
8. After creating the call, click the **Edit Data** arrow to change the details of the call if necessary.
9. Click the **Invoke** arrow to invoke the service call.

Result

If the call was successful, the response is displayed under the **View Response** arrow.


10. To record a test with multiple calls, repeat steps 6 through 9.
11. When you have finished sending service requests, stop the recorder. You can do this by closing the generic service client or by clicking the **Stop** push button  in the **Recorder Control** view.
If you changed the network settings of the client program as described in step 8, you can revert to the default settings before closing the program.

Result

The **Generate Service Test** wizard opens.

12. Click **Finish**.

What to do next

Alternatively, you can use the generic service client to create, edit, and invoke the calls without recording. Successful responses are added to the **Request History** list. You can select calls in the **Request History** list, and click the **Generate Test Suite** icon .

 Related information

[Sending service requests with the generic service client on page 338](#)

[Recording a service test through a client program on page 211](#)

[Recording sensitive session data on page 239](#)

[Sending service requests with WSDL files on page 350](#)

Recording a service test through a client program

You can record tests for SOAP-based, XML, plain text, or binary services with any client program that uses the HTTP protocol. To record the test, the recorder intercepts the service calls and message returns between the client and the service. You can choose between an HTTP or SOCKS proxy recorder or a low-level socket recorder, depending on the capabilities of the client program.

Before you begin

The following recorders are available for recording traffic from an application:

- **SOCKS proxy recorder:** Use this recorder when no proxy connections are required.
- **HTTP proxy recorder:** Use this recorder when a proxy connections is required to connect to the network or when the client program does not support SOCKS.
- **Socket recorder:** Use this recorder for low-level network traffic when the client does not support proxies. This recorder does not support SSL authentication or encryption of any kind and is only available if the IBM® Rational® Performance Tester Extension *for Socket Protocols* is installed.

Regardless of the recorder that you use, the client program must use the HTTP network protocol. For recording Java™ Message Service (JMS) or IBM® WebSphere® MQ tests, see [Recording a service test with the generic service client on page 208](#).


If you are using Secure Sockets Layer (SSL), the HTTP or SOCKS proxy can cause authentication problems because the proxy recorder relays traffic between the client and the server. Depending on the authentication method in place, the client might require that the proxy recorder authenticate itself as the server and the server might require that the proxy recorder authenticate as the client. If the client program requires an authenticated server, you must either have access to the server certificate keystore and provide it to the proxy recorder or configure the client to accept the default certificate from the proxy recorder instead of the certificate from the actual server.

If you are testing a SOAP-based web service, ensure that you have access to a valid Web Services Description Language (WSDL) file. The wizard can import WSDL files from the workspace, the file system, a remote repository,

or from a URL. Ensure that the WSDL files use the correct syntax for the test environment. The generic service client might not work with some WSDL files.

If you are using SOAP security, ensure that you have configured the environment with the correct libraries and configuration files. See [Configuring the environment for SOAP security on page 208](#) for more information.

To record a service test with a client program:

1. In the Performance Test perspective, click the **New Test from Recording** toolbar button  or click **File > New > Test from Recording**.
2. In the **New Test from Recording** wizard, click **Create a test from a new recording**, select **Service Test**, and click **Next**.

If you are recording sensitive data, you can select a **Recording encryption level**.

3. On the **Select Location** page, select the project and folder to create the test in, type a name for the test, and click **Next**.

If necessary, click **Create Parent Folder**  to create a project or folder

4. On the **Select Client Application** page, select the type of client program to use.

The program type defines the recorder that can be used. The following client program types are checkmark_perf.jpg for recording a service test:

Choose from:

- **Managed Application:** This option starts a specified program and uses a proxy or socket recorder to record the traffic.

On the **Managed Application Options** page, click **Browse** to specify the **Program path**. If necessary, specify the **Working directory**, and type the command line **Arguments** that the program requires.

If the program requires user input from a command-line interface, select **Open console for user input**.

- Choose a web browser to record traffic that is sent and received with the web browser.
- **Unmanaged Application:** This option enables you to record traffic from one or multiple client programs that use a proxy. You must manually start the client programs and the proxy recorder records all traffic that is sent and received through the specified network port.
- **Generic Service Client:** This option uses the generic service client if you do not have access to a dedicated client for the service calls. See [Recording a service test with the generic service client on page 208](#) for using the generic service client to record service tests.

5. On the **Recorder Settings** page, depending on the type of client program you selected, specify these details:

- a. If you selected **Managed Application**, specify the recording method.
 - Select **Record traffic with the proxy recorder** to record HTTP or SOCKS traffic through a proxy.
 - Select **Record traffic with the socket recorder** to record low-level network traffic for applications where a proxy cannot be used. This recorder does not support SSL authentication or encryption.



Note: When using proxy recording, you can filter out HTTP or HTTPS requests to a specific endpoints so that any requests to those endpoints are not recorded. See Proxy recording preferences

b. If you selected **Record traffic with the proxy recorder**, specify whether the proxy recorder uses HTTP or SOCKS. Select **HTTP** if a connection to proxy is required or if your application does not support SOCKS.

c. If you are using SSL authentication, specify the authentication settings for the proxy recorder.

During the recording, the proxy recorder is between the client and the server.

- If the server requires client SSL authentication, provide the client certificate for the proxy recorder to be authenticated by the server as though the proxy recorder were the client. Select **The server requires a specific client certificate**.

To provide single certificate keystore, specify the file name and password of the server certificate keystore. If multiple certificates are required, click **Multiple certificates**, and click **Add** to specify a certificate keystore file name and password for each host name and port.

- To record a secured site using Microsoft Edge or Google Chrome on Windows, install the recorder certificate by selecting **Register the recorder root certificate authority**. Before the recording starts, the browser prompts you to install the certificate. After the recording is stopped, the browser prompts you to uninstall the certificate. To avoid multiple prompts for each recording, select **Keep the recorder root certificate authority after recording**.



Note: If you already had the certificate from a version prior to 9.2.1 and then install the latest version of the product, you might have to install the certificate again.

This option is not available when you record by using the Mozilla Firefox or Safari browser.

To record a secured site on these browsers, manually import the certificate in the browser from the default location `C:\Program Files\IBM\IBMIMShared\plugins\com.ibm.rational.test.lt.recorder.proxy_version\SSLCertificate`. For information about how to import the certificates, see the browser's documentation.

- If the client requires server authentication, you must provide the server certificate keystore for the proxy recorder to be authenticated by the client as though the proxy recorder were the server. Select **The client requires a specific server certificate**, and click **Add** to specify a

certificate keystore filename and password for each hostname and port. If you do not select this option, the proxy recorder provides its own default certificate.



Note: The keystore must contain the private certificate of the server.

- d. If you selected to use the HTTP proxy recorder, specify how to connect to the network. If necessary, specify an HTTP or SOCKS proxy or point to a proxy auto-configuration (PAC) file.
Use this option if you are connecting to the service through a corporate proxy or firewall.

6. Click **Next**. If this is the first time you record a service test and you did not select a web browser for the client application, read the Privacy Warning, select **Accept**, and click **Finish** to proceed.
7. If you selected a proxy recorder with a managed or unmanaged application, change the network settings of the client program to use the proxy recorder.







The method for changing the network settings depends on the client program. However, you must be able to set the following proxy settings in the program:


- SOCKS or HTTP proxy: Specify the protocol that you selected for the proxy recorder in the wizard.
- Host name: Set to `localhost`.
- Port: Specify the port number that you selected for the proxy recorder in the wizard.

To avoid unexpected results, revert to the previous proxy settings before you stop the recording.

8. Use the client program to perform the actions to test.

You can use the **Recorder Test Annotations** toolbar to add comments, record synchronizations, or take screen captures during the recording.

- To add a comment to the recorded test, click the **Insert comment** icon .
- To add a screen capture to the recorded test, click the **Capture screen** icon . Screen and window captures make your tests easier to read and help you visualize the recorded test. You can change the settings for screen captures and add a comment to the image.
- To manually add a synchronization point to the recording, click the **Insert synchronization** icon .
- To manually add a transaction folder to the recording, click the **Start Transaction** icon  and **Stop Transaction**  icon to start and stop the transaction.
- To insert a split point into the recorded test, click the **Split point** icon . With split points, you can generate multiple tests from a single recording, which you can replay in a different order with a schedule.

9. After you finish the user tasks in the client program, stop the recorder. You can do this by closing the client program or by clicking the button **Stop**  in the **Recorder Control** view.

If you changed the network settings of the client program as described in step 8, you can revert to the default settings before closing the program.

Result

The **Generate Service Test** wizard opens.

10. If you inserted a split point during the recording, on the **Destination** page, specify the location for the split test or merge the split recordings together.

See [Splitting a test during recording on page 240](#) for more information about splitting tests.

11. On the **Service Test Generation Options** page, if you are testing a SOAP-based web service, specify a Web Services Description Language (WSDL) file from the workspace or click **Add** to import a WSDL or to link to a remote WSDL file and click **Next**.
12. Select the domains to include in the test and click **Finish**. The domains that are not selected are not included in the test. You can add them back by generating the test again from the recording.
To include all the domains for all of the recordings, click the **Select all and remember my decision** check box. To enable the filter again for HTTP tests, click **Window > Preferences > Test > Test Generation > HTTP Test Generation**, and, for Service tests, click **Service Test Generation** and then click the **Enable domain review before test generation** check box.
13. Click **Finish**.

Results

A progress window opens while the test is generated. On completion, the **Recorder Control** view displays the `Test generation completed` message, the test navigator lists your test, and the test opens in the test editor.

Related information

[Recording a service test with the generic service client on page 208](#)

[Sending service requests with the generic service client on page 338](#)

[Recording sensitive session data on page 239](#)

[Sending service requests with WSDL files on page 350](#)

Preparing to record a test for the HTTP/2 service

To test a web service that is based on the HTTP/2 protocol, record a test by using the SOA extension of Rational® Performance Tester. Before recording the HTTP/2 service, follow the procedure in this topic to configure your computer.

About this task

This configuration is required because this feature is released as Beta and is intended for use in a non-production environment only.

Use Mozilla Firefox or Google Chrome when recording on servers that support the HTTP/2 service.

1. Download the following Application Layer Protocol Negotiation (ALPN) boot jar file <https://mvnrepository.com/artifact/org.mortbay.jetty.alpn/alpn-boot/8.1.8.v20160420>
2. Create or rename the `productInstallDir\jdk` folder to `.\jdk.ibm`. You can rename the folder back to `jdk` later to test with the IBM JDK.
3. Download Oracle Java 1.8.0u92 from <http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>. You can then either extract the compressed file or install Java at `productInstallDir\jdk`.
4. Copy the ALPN jar file to `productInstallDir\majordomo\lib`.
5. From `productInstallDir`, open `eclipse.ini` and add the following flags:

```
-Xbootclasspath/p:<productInstallDir>\majordomo\lib\alpn-boot-8.1.8.v20160420.jar
```



Note: If there are any other flags starting with -X, delete those flags.

6. Configure Rational® Performance Tester Agent to use Oracle Java.

- a. Stop the Majordomo process.

On Windows systems, run the following command: `cd "c:\program files\ibm\sdp\majordomo" ngastop`

On Linux systems, run the following command: `cd /opt/IBM/SDP/Majordomo ./MDStop.sh`

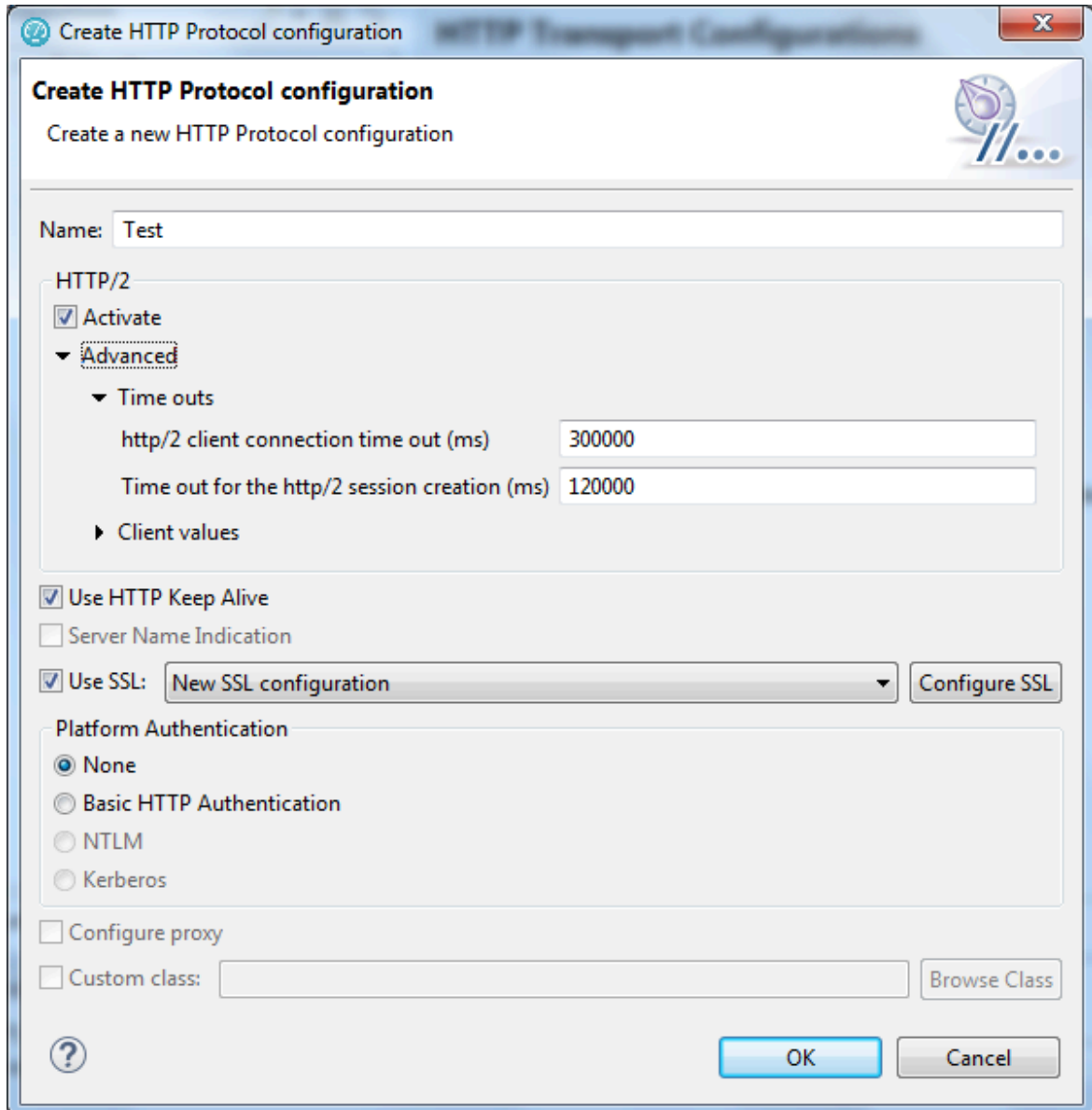
- b. Set the environment variable *RPT_JAVA* to the Oracle Java binary or executable file.

On Windows systems, run the following command: `set RPT_JAVA=c:\program files\java\jdk1.8.0_92\bin\java.exe`

On Linux systems, run the following command: `export RPT_JAVA=/root/jdk1.8.0_92/jre/bin/java`


- c. Start the Majordomo process.

7. When you record a service by using a web browser against an HTTP/2 client, the HTTP/2 traffic is automatically captured. But, to record an HTTP/2 service by using the GSC client, in the HTTP transport protocol configuration dialog, you must select the **Activate** check box.



8. Specify the following configuration options for HTTP/2:

HTTP/2

 **Note:** Testing HTTP/2 service is in the Beta mode. For more information, see [Preparing to record a HTTP/2 service on page 215](#).

To test a service that uses the HTTP/2 protocol, select the **Activate** check box. This check box is automatically selected when you record a service by using a browser. If you use the Generic Service Client component to create a HTTP/2 test, you have to manually select the check box.

HTTP/2 client connection timeout

Specifies the time limit for the HTTP/2 client to connect to the HTTP/2 server.

Time out for the HTTP/2 session creations

Specifies the time limit to create the HTTP/2 session. This time starts after the connection is established.

Enable HTTP/2 Push

The Push functionality of HTTP/2 automatically identifies and passes the related objects or requests to the client when a request is sent to the server. Clear the check box to not use the functionality.

Initial session window

Specifies the buffer size on the sessions.

Initial stream window

Specifies the window size for buffer on each stream after the connection is established.

HTTP/2 Client Input Buffer Size

Specifies the buffer size that is used to read the network traffic.

Maximum Quantity of Messages that can be queued

Specifies the maximum number of messages that can be queued for the HTTP/2 client on a thread.

Maximum Quantity of HTTP/2 thread pool

Specifies the maximum number of thread pools that will be used by the HTTP/2 client to distribute the workload.

Minimum Quantity of HTTP/2 thread pool

Specifies the minimum number of thread pools that will be used by the HTTP/2 client to distribute the workload.

HTTP/2 client bytearray pool size

Specifies the buffer size to receive the unciphred values.

Server Name Indication



Note: Not applicable for HTTP/2.

Clear this check box if you do not want to connect to the host computer by using the Server Name Indication protocol. If the host computer is already configured with Server Name Indication protocol, you should keep this check box selected.

Use HTTP Keep Alive

Select this option to keep the HTTP connection open after the request. This option is not available if you are using IBM® Rational® AppScan®.

Use SSL

Select this option to use an SSL configuration. Click **Configure SSL** to create an SSL configuration or select an existing configuration.

Platform Authentication

In this section, specify the type of authentication that is required to access the service. Select **None** if no authentication is required.

Basic HTTP authentication

Select this option to specify the **User Name** and **Password** that are used for basic authentication.

NTLM authentication

Note: Not applicable for HTTP/2.

Select this option to use the Microsoft™ NT LAN Manager (NTLM) authentication protocol. NTLM uses challenge-response authentication. This view lists what is negotiated (checkmark_perf.jpg by the client and requested of the server) and what is authenticated (the client reply to the challenge from the server).

Kerberos authentication

Note: Not applicable for HTTP/2.

Select this option to use the Kerberos authentication protocol between the client and server.

Connect through proxy server

Note: Not applicable for HTTP/2.

If the HTTP connection needs to go through a proxy server or a corporate firewall, specify the **Address** and **Port** of the proxy server. If the proxy requires authentication, select either **Basic proxy authentication** or **NTLM proxy authentication**.

Proxy authentication

In this section, specify the type of authentication that is required to access the proxy. Select **None** if no authentication is required.

Basic proxy authentication

Select this option to specify the **User Name** and **Password** that are used for basic authentication.

NTLM proxy authentication

Select this option to use the Microsoft™ NT LAN Manager (NTLM) authentication protocol. NTLM uses challenge-response authentication. This view lists what is negotiated (checkmark_perf.jpg by the client and requested of the server) and what is authenticated (the client reply to the challenge from the server).

Custom class



Note: Not applicable for HTTP/2.

Select this option if the communication protocol requires complex, low-level processing with a custom Java™ code to transform incoming or outgoing messages. Click **Browse** to select a Java™ class that uses the corresponding API. This option is not available in IBM® Security AppScan®.

9. Click **OK**. You have configured the workbench to test an HTTP/2 service.

What to do next

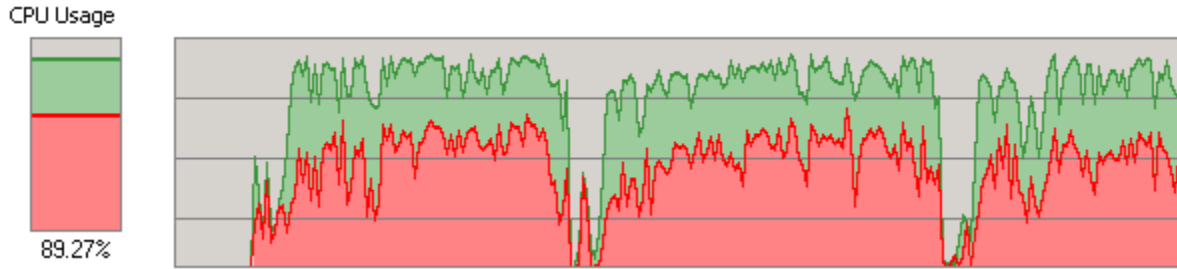
You can now record a regular SOA test for the HTTP/2 service. After the recording, in the Version field of request details, the requests are marked with HTTP/2 indicating that the HTTP/2 traffic is captured. If the test playback fails, check if all the steps are correctly followed.

Optimizing HTTP/2 tests for SOA

HTTP/2 tests require a lot of CPU and memory resources. When you apply load on HTTP/2 tests using computers that do not have enough resources, the tests might fail. You might want to configure or tune the computers that run HTTP/2 tests.

CPU Usage

Ensure that the HTTP/2 tests get adequate CPU resources to run. If there are other processes running on the computer and they are not required, you can stop them. For example, the CPU usage statistic in the image below indicates that the other processes (shown in red) on the computer are consuming a lot of resources whereas the test execution process (shown in green) is getting less resources.



Memory usage and garbage collection

Ensure that enough memory is available for the test execution. You can configure the garbage collector and adjust the memory heap size.

Garbage Collection - Consider using the following values so that the garbage collector does not allocate large amount of temporary memory. By doing so, you are tuning the number of threads allocated for the garbage collector according to the capability of the computer. You apply the values for each location asset of the schedule.

RPT_VMARGS

`-XX:MaxGCPauseMillis=250 -XX:ParallelGCThreads=6 -XX:ConcGCThreads=3 -XX:GCTimeRatio=19`

General Properties

ROOTDIR	=	/tmp/cloud_agent
OPERATING_SYSTEM	=	LINUX
CLOUD_ROLE	=	CLOUD_AGENT
RPT_VMARGS	=	-XX:MaxGCPauseMillis=250...
LOCATION_TEMPLATE	=	/myProj/Cloud-SL.loctemp...
RPT_ENABLE_IP_ALIASING	=	FALSE
RPT_IPA_ENABLE_ALL_INTE...	=	TRUE

Memory heap - Consider using the following values for memory heap:

RPT_VMARGS

`-Xms11024m`

RPT_DEFAULT_MEMORY

`22412m`

General Properties

ROOTDIR	=	/tmp/cloud_agent
OPERATING_SYSTEM	=	LINUX
CLOUD_ROLE	=	CLOUD_AGENT
RPT_VMARGS	=	-Xms11024m
LOCATION_TEMPLATE	=	/myProj/Cloud-SL.loctemp...
RPT_ENABLE_IP_ALIASING	=	FALSE
RPT_IPA_ENABLE_ALL_INTE...	=	TRUE
RPT_DEFAULT_MEMORY	=	22412m

Thread Usage

Ensure that you start load testing with fewer virtual testers and gradually ramp up the workload. This practice helps in observing the changes in the workload, that is, the number of calls per second. In the graph, when the number of calls per second is flat, it indicates that the maximum capacity of the computer is reached and there is no need to add more virtual testers.

Platform tuning

Configure the TCP/Socket capabilities of your system by following these two links:

<https://www.eclipse.org/jetty/documentation/jetty-9/index.html#high-load>

<https://www.microsoft.com/en-us/download/details.aspx?id=56495>

Creating a service test from a BPEL model

You can use Business Process Execution Language (BPEL) resources from your workspace to automatically generate a set of service tests that corresponds to the paths that are run in a synchronous BPEL model.

Before you begin

Tests are stored in test projects. If your workspace does not contain a test project, the test creation wizard creates one, enabling you to change its name. To store a test in a specific project, verify that the project exists before you record the test.

If you are using Secure Sockets Layer (SSL) authentication, ensure that you have any required key files in your workspace.

If you are using Java™ Message Service (JMS), ensure that you have configured the environment with the correct libraries and configuration files. Ensure that the WSDL files use the correct syntax for the test environment.

If you are using SOAP security, ensure that you have configured the environment with the correct libraries and configuration files.

BPEL models must be synchronous. Asynchronous BPEL models are not checkmark_perf.jpg.

Ensure that the BPEL models refer to the WSDL files in a valid import statement, for example:

```
<bpws:import importType="http://www.w3.org/2001/XMLSchema" location="foo.wsdl" namespace="http://foo"/>
```

Relative file paths, such as: `../../foo.wsdl` are not checkmark_perf.jpg.

Ensure that you have one or more valid Web Services Description Language (WSDL) files and the associated BPEL model in your workspace. Only the calls to services with a valid web service binding are taken into account. For example, if the BPEL model was produced in IBM® Websphere Integration Developer, then services must be exported with the following web service bindings:

```
<bpws:invoke name="myOperation" operation="myOperation" partnerLink="IServicePartner"
portType="ns3:IService" wpc:displayName="myOperation" wpc:id="20">
```

Only BPEL *invoke* activities are considered for generating tests. Any BPEL *receive* and *reply* activities are ignored.

Websphere Integration Developer does not generate the required `soapAction` attributes for the soap operations in the WSDL files. Please edit the generated WSDL files, as follows for every operation: `<soap:operation soapAction="" />`.

To create a service test from a BPEL model:

1. In the Performance Test perspective, click **File > New > Other > Test > Test Assets > BPEL to Web service test**, and then click **Next**.
2. Click **Browse** to select a BPEL file from the workspace, and click **Next**.
3. On the **Web service test generation** page, change the number of paths by specifying how activities and sequences from the flow of the BPEL model are processed. Each path generates one test.
 - a. In the **Flow** section, select how any concurrent sequences that are found in the flow will be converted into paths.
 - b. In the **Switch** section, select whether to test *otherwise* activities from the flow.
 - c. In the **Throw** section, select how *throw* activities from the flow are converted into paths.
 - d. In the **Invoke** section, select whether to test inline catches inside *invoke* activities from the flow.
 - e. Select **Enable data correlation in generated tests** to automatically create references in the generated test elements by propagating variables to the parameters of the web service call and message return elements.
4. Click **Recount paths** to update the number of paths to test, and click **Next**.
One test is generated for each path.
5. For WSDL operations that are bound to multiple ports, you must select one port that is to be used for the test.

Under each test that will be generated, the **Operations** list displays the WSDL operations that are bound to multiple ports.

If no WSDL operations are displayed under the tests, this means that all operations are bound to a single port. In this case, skip step 6.

- a. In the **Operations** list, expand a test and select a WSDL operation that requires binding.
- b. In the **Binding ports** list, select the port that you want to use to test the selected WSDL operation.
- c. Repeat steps a and b for each WSDL operation that requires binding.

6. Click **Next**.
7. Select a location and a name for the new folder where the tests generated from the BPEL model are created, and click **Finish**.

Results

A new folder is created in the Test Navigator containing the generated service tests. These tests are generated with default message content and must be edited with valid input values.

Creating a service test manually

You can create a service test without recording by simply adding the test elements as required and manually editing the test element details in the test editor.


Before you begin

Tests are stored in test projects, which are test projects that include a source folder. You must create a test project before creating a test.

Ensure that you have a valid WSDL file in your workspace. Ensure that the WSDL files use the proper syntax for the test environment.

If you are using Secure Sockets Layer (SSL) authentication, ensure that you have any required key files in your workspace.

If you are using SOAP security, ensure that you have configured the environment with the proper libraries and configuration files.

1. In the workbench, click **File > New > Other > Test > Test Assets > Web service test** or click the **New Service Test**  toolbar button.
2. Select a project and, in **Name**, type a name for the test, and then, click **Next**.
The name that you type is the base name for the recording, test, and other required files. You see these files in standard Navigator or the Java™ Package Explorer with their distinguishing suffixes, but you see only the simple (test) name in the Test Navigator.
3. Select a web service request to create the test for.
If you select **Web service request** or one of the options in **Specification-based structure**, specify a WSDL port and then configuration properties for the HTTP protocol. If you select, **XML request** and **Text request**, specify the configuration properties for the HTTP, JMS, WebSphere MQ, WebSphere Java MQ, and Microsoft.Net protocols.

For information about the configuration properties of each protocol, see the topics in [Sending service requests with the generic service client on page 338](#).
4. Click **Finish**. The service test is created.

Creating a service test for WebSphere® MQ

You can create an IBM® WebSphere® MQ test by adding the test elements as required and editing the test element details in the test editor.


Before you begin

Tests are stored in test projects, which are Java™ projects that include a source folder. You must create a test project before creating a test.



Ensure that you have a valid Web Services Description Language (WSDL) file for a WebSphere® MQ service in your workspace.

If you are using Secure Sockets Layer (SSL) authentication, ensure that you have any required key files in your workspace.

If you are using SOAP security, ensure that you have configured the environment with the correct libraries and configuration files.

1. In the workbench, click **File > New > Other > Test > Test Assets > Web service test** or click the **New Service Test**  toolbar button.
2. Select a project, and then, in **Test file name**, type a name for the test and click **Next**.
The name that you type is the base name for the recording, test, and other required files. You see these files in the standard Navigator or the Java™ Package Explorer with their distinguishing suffixes, but you see only the simple (test) name in the Test Navigator.
3. In the **Select a service request interface** page, complete one of the following steps:
 - a. To test a service that use a WSDL file, select **Web service request** or **Specification-based structure**, click **Next**, and select a WSDL file.
 - b. To test a service that does not use a WSDL file, select **XML Request**, **Text Request**, **Binary Request** or an **Empty test**.
4. Click **Next** and select the **WebSphere MQ** protocol.
5. In **SOAP Action**, specify the SOAP action to be used to invoke the MQ request.
6. To override the message header and descriptor that was specified in WebSphere MQ transport configuration, click **Override MQ Protocol Configuration values** and specify the customize header and message descriptor.
7. Click **Finish**. The service test is created.
8. On the web service call, click **Update Response**.
This opens the **Response Preview** window, displaying the data that will be used to perform the call.
9. Click **Update Test**.
This action calls the web service and creates a message return element with the return data. If a message return element already exists, then it is updated with latest return data. With the message return test element, you can implement data correlation and content-based verification points.

Creating a service test for WebSphere Java MQ


To test Java-based applications, create a service test and add the WebSphere Java MQ messages. You can create a service test by using Generic Service Client option  or the New Service Test wizard .

Before you begin

Connect to a WebSphere MQ server.

If you are using Secure Sockets Layer (SSL) authentication, ensure that you have any required key files in your workspace.

If you are using SOAP security, ensure that you have configured the environment with the correct libraries and configuration files.

1. In the workbench, click **File > New > Other > Test > Test Assets > Web Service Test** or click **Create a Service Test** .
2. Select a project, and then, in **Test file name**, type a name for the test.
The name that you type is the base name for the recording, test, and other required files. You see these files in the standard Navigator or the Java™ Package Explorer with their distinguishing suffixes, but you see only the simple test name in the Test Navigator.
3. In the **Select a service request interface** page, complete one of the following steps:
 - a. To test a service that use a WSDL file, select **Web service request** or **Specification-based structure**, click **Next**, and select a WSDL file.
 - b. To test a service that does not use a WSDL file, select **XML Request**, **Text Request**, **Binary Request** or an **Empty test**.
4. Click **Next**, select the **WebSphere Java MQ** protocol, and specify a transport configuration. If necessary, click **New** to create the transport configuration for the call. See [Creating a WebSphere Java MQ transport configuration on page 344](#).
5. Complete the following information in the **General** tab:



Learn more about the UI elements in the General tab:

Queue

Name of the queue as defined on the WebSphere MQ server.

Message type

The types of messages are these:

- *Datagram* means that the message does not require a reply.
- *Request* means that the message requires a reply.
- *Reply* means that the message is a reply to an earlier request message.
- *Report* means that the message is reporting on some expected or unexpected occurrence, usually related to some other message. An example is a request message that contained data that was not valid.

**Message Persistence**

This value indicates whether the message is persistent or not. If the message is persistent, it survives the system failures and restarts of the queue manager. If the message is not persistent, it survives a restart if it is present on a queue having the NPMCLASS(HIGH) attribute. However, even with the NPMCLASS(HIGH) attribute a message does not survive a QMGR class. Nonpersistent messages on queues having the NPMCLASS(NORMAL) attribute are discarded at queue manager restart, even if the message is found on the auxiliary storage during the restart procedure.

Dynamic Reply

Select this check box for the WebSphere MQ server to dynamically create a temporary queue as a reply. If this check box is not selected, the message in Reply Queue is used.

Reply Queue

This is the name of the message queue to which the application that issued the get request for the message should send the reply and report messages.

Reply Manager

This is the name of the queue manager on which the reply-to queue is defined.

Additional properties

Specify the additional properties for the queues.

6. **Optional:** If necessary, complete the following information on the **Config** tab:

**Learn more about the UI elements in the Config tab:****Message Priority**

This is the priority of the message. The lowest priority is 0.

Encoding

This is the numeric encoding of numeric data in the message. This value does not apply to numeric data in the MQMD structure itself.

Expiry Interval

This is the period of time, in tenths of a second, after which the message becomes eligible to be discarded if it has not already been removed from the target queue. The expiry interval is set by the application that put the message.



Character set

This is the character set identifier of the character data in the application message data.

7. **Optional:** In the **Report** tab, select the report messages to receive.
8. **Optional:** If necessary, complete the following information in the **Context** tab:



Learn more about the UI elements in the Context tab:

Application Identity Data

This information is defined by the application suite. Use it to provide information about the message or its originator.

Application Origin Data

This information is defined by the application suite. Use it to provide additional information about the origin of the message.

Accounting Token

This information is needed by the application to appropriately charge for the work that is done as a result of the message.

User ID

This is the user identifier of the application that originated the message.

9. **Optional:** In the **Identifiers** tab, for the messages that require binary input, specify the ID in the string format in the second column. The first column is filled automatically in the hexadecimal format.
10. **Optional:** In the **Segmentation** tab, select the segment of the message and click **Next**.
11. If you had selected **XML Request**, click **Next**, select a XSD file and click **Finish**.

Result

The new service test is created.

What to do next

You can now enhance the test and run it.

Creating a service test for a plain XML call


You can create a test for a plain XML call over HTTP, JMS, or IBM® WebSphere® MQ, by simply adding the test elements as required and editing the test element details in the test editor.

Before you begin

Tests are stored in test projects, which are Java™ projects that include a source folder. You must create a test project before creating a test.

If you are using Secure Sockets Layer (SSL) authentication, ensure that you have any required key files in your workspace.

If you are using SOAP security, ensure that you have configured the environment with the correct libraries and configuration files.

1. In the workbench, click **File > New > Other > Test > Test Assets > Service Test** or click the **New Service Test**  toolbar button.
2. Select a project, and then, in **Test file name**, type a name for the test and click **Next**.
The name that you type is the base name for the recording, test, and other required files. You see these files in the standard Navigator or the Java™ Package Explorer with their distinguishing suffixes, but you see only the simple (test) name in the Test Navigator.
3. On the **Select Service Call Interface** page, select whether you want to create a test using a plain **XML call** interface or a **Web service call** interface.
If you select web service call interface, select or add a WSDL file and then, select port to which the call will be binded. Click **Next**.
4. On the **Configure Protocol** page, select either **HTTP**, **JMS** or WebSphere® **MQ** as the protocol and then, specify the options for the selected **Protocol configuration**.
5. On the **Select Root Element** page, you can select an XSD and then, select a root element for the call.
6. Click **Finish**.

Changing service test generation preferences

You can change default test generation values by changing the preference settings. The default settings, however, are appropriate for recording in most cases.

1. Click **Window > Preferences > Test > Web Services Test Generation**
2. Select the setting to change.

Time out delay used for call

This is the default time out for web service calls. If the web service does not respond within this period, an error is produced.

Think time default value

This is the default think time for generated tests.

3. After changing a setting, click **Apply**.

Digital certificates overview

The digital certificates feature enables you to run tests against servers that use Secure Sockets Layer (SSL) for applications that require client-side digital certificates to authenticate users.

A *digital certificate* is a file that binds a public cryptographic key with an identity (a user or an organization). Trusted certificate authorities issue digital certificates, which are then used to authenticate users and organizations for

access to websites, email servers, and other secure systems. A *certificate store* is an archive file that contains almost any number of digital certificates, possibly certificates that are issued from different certificate authorities.

To use digital certificates in tests:

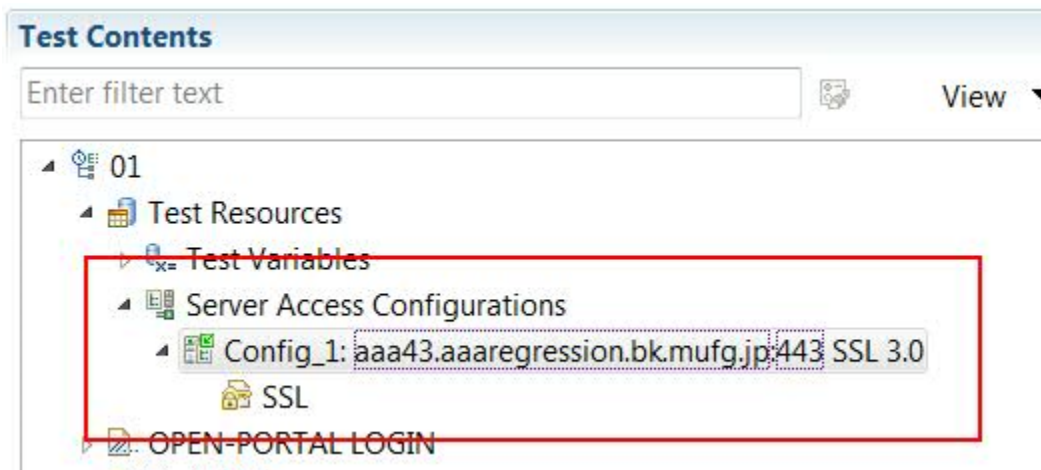
1. Create a digital certificate store. For more information about this subject, see [Digital certificate creation overview on page 231](#) and [Creating a digital certificate store on page 234](#).
2. Associate a digital certificate with a test for playback. For more information about this subject, see [Playing back a test with a digital certificate on page 237](#).
3. Optionally, you can associate the digital certificates in one or more digital certificate stores with a dataset. For more information about this subject, see [Using a digital certificate store with a dataset](#).

Create a certificate store by running the supplied KeyTool command-line program. The program creates a certificate store that contains digital certificates.

Record a test that requires using a digital certificate. Specify the certificate and password that to use, and then begin recording the test. Browse the website as you typically would to record a test.

After you have finished recording, open the test for editing. On the Common Options page, under Digital Certificates, click **Add**. Type the name of the certificate store that you created previously; then select the certificate that you want to use. Save the test. When you run this test, the digital certificate from the certificate store is submitted to the server.

If you have recorded a test that does not use SSL, you can convert that test to be secure by adding an SSL object to the corresponding Server Access Configuration in the test.



To use a certificate store with a dataset, open the test for editing. On the **Common Options** page, click **Add Dataset**. Create a dataset with two columns that contains a list of the certificates in the certificate store and a list of passphrases for the certificates. Select **Fetch only once per user**. Save the dataset. On the Common Options page, under Digital Certificates, click **Add**. Select the certificate store that you created previously from the **Certificate Store** column. Insert a **Certificate Name** for the digital certificate. Highlight this name, and then select **Substitute from dataset**. Choose the dataset added previously, and then choose the column with the certificate name. Repeat this

process to substitute passphrases from the dataset column containing passphrases. Save the test. Add the test to a schedule. When you run this schedule, the certificates from the certificate store are submitted to the server.

Digital certificate creation overview

If you want to use digital certificates to run tests against applications that require client-side digital certificates to authenticate users, work with the appropriate server administrators to determine the types of certificates that you need to create.

In cryptography, a public key certificate is a document that uses a digital signature to bind a public key with a physical identity. These certificates are often referred to generically as digital certificates or client digital certificates. The most common standard for digital certificates is the X.509 standard.

In public key cryptography, each certificate has two associated keys: a public key and a private key. The public key is incorporated into the X.509 certificate and is always available with the certificate itself. The private key is always kept private (meaning, it is never transmitted). For ease of portability, the two keys (and the certificate) can be included in one, encrypted and passphrase-protected, format known as PKCS#12.

In order to verify the authenticity of a certificate, it is digitally signed by another certificate, known as a Certificate Authority (CA). This CA certificate may be one created (and kept secure) by a company hosting a secure application, or it could be created by a company such as Verisign.

When a web application requires digital certificates, an administrator typically creates digital certificates for each authorized user. The administrator digitally signs each certificate using the system CA certificate. These certificates, along with the public and private keys, are distributed to users. Often these keys will be distributed in the PKCS#12 format. Users then import these certificates into their web browsers. When the browser is challenged by the server, it will produce its certificate.

When importing certificates for web applications, select the check box that indicates that the keys be exportable. With this indication, the certificate can be exported to a PKCS#12 formatted file for later use by other programs.

Do not use certificates that are assigned to actual users for performance testing purposes. Use test certificates that do not correspond to actual users.

There are four types of certificates that can be used in testing:

- Self-signed certificates
- Signed certificates
- Certificate authority (CA) certificates
- Unsigned certificates (rarely used)

Self-signed certificates are used when no entity needs to vouch for the authenticity of the certificate. These are the simplest certificates to create and use. Typically, however, a signed certificate is used to represent a particular user.

Signed certificates are used when a certificate needs to be created for and issued to one, and only one, user. Signed certificates are signed by a certificate authority (CA).

Certificate authority (CA) certificates are self-signed certificates used to sign (certify) certificates.

Unsigned certificates are certificates that are neither signed by a CA nor self-signed. Most web applications do not use unsigned certificates.

When you create a self-signed or signed certificate (including CA certificates) you can specify a *subject*. The subject of a certificate is the set of attributes of an X.500 Distinguished Name that is encoded in the certificate. The subject enables the recipient of a certificate to see information about the owner of the certificate. The subject describes the certificate owner, but is not necessarily unique. Think of subjects as entries in a telephone book; there can be multiple entries for Patel Agrawal, but each entry refers to a different person.

The subject can contain many different types of identifying data. Typically, the subject includes the following:

Attribute	Example
COMMON NAME (CN)	CN=Patel Agrawal
ORGANIZATION (O)	O=XYZ Corporation
ORGANIZATIONAL UNIT (OU)	OU=XYZ Software Group
COUNTRY (C)	C=IN
LOCALITY (L)	L=Bangalore
STATE or PROVINCE (ST)	ST=Karnataka
E-MAIL ADDRESS (emailAddress)	emailAddress=agrawal@xyz.com

This information can be typed as one string, using forward slashes to separate the data.

For example, the above subject would be typed as follows:

```
/CN=Patel Agrawal/O=XYZ Corporation/OU=XYZ Software Group/C=IN/L=Bangalore/ST=Karnataka/  
emailAddress=agrawal@xyz.com
```

To learn more about using the supplied command-line program to create certificates, see [Creating a digital certificate store on page 234](#).

Creating a digital certificate with OpenSSL

You can use the OpenSSL program to create digital certificates for use with tests.

Before you begin

OpenSSL is available from the OpenSSL Project at <http://www.openssl.org/>.

1. Create a certificate authority (CA).

For the purposes of testing, this CA takes the place of a recognized CA on the Internet, such as VeriSign. You use this CA to digitally sign each certificate that you plan to use for testing.

- a. Create a certificate request (CSR) file. The "subject" (-subj) describes the user of the certificate. Enter dummy values as shown. The following command line sets the password for the certificate to `abcdefg`.

Example

```
openssl req -passout pass:abcdefg -subj "/C=US/ST=IL/L=Chicago/O=IBM Corporation/OU=IBM Software
Group/CN=Rational Performance Tester CA/emailAddress=rpt@abc.ibm.com" -new > waipio.ca.cert.csr
```

- b. Create a key file, `waipio.ca.key`, to store the private key.
This removes the password protection from the certificate request file so that you do not have to type the password every time you sign a certificate. Because the password protection has been removed, use the certificate request file for testing purposes only.

Example

```
openssl rsa -passin pass:abcdefg -in privkey.pem -out waipio.ca.key
```

- c. Create an X.509 digital certificate from the certificate request. The following command line creates a certificate signed with the CA private key. The certificate is valid for 365 days.

Example

```
openssl x509 -in waipio.ca.cert.csr -out waipio.ca.cert -req -signkey waipio.ca.key -days 365
```

- d. Create a PKCS#12-encoded file containing the certificate and private key. The following command line sets the password on the P12 file to `default`. Rational® Performance Tester uses password of `default` for all PKCS#12 files by default.

Example

```
openssl pkcs12 -passout pass:default -export -nokeys -cacerts -in waipio.ca.cert -out
waipio.ca.cert.p12 -inkey waipio.ca.key
```

Result

You now have a CA certificate (`waipio.ca.cert`), which can be installed into the web server under test and a private key file (`waipio.ca.key`) that you can use to sign user certificates.

2. Create a digital certificate for a user.
 - a. Create a CSR file for the user. Set the initial password to `abc`. Optionally, provide an appropriate subject.

Example

```
openssl req -passout pass:abc -subj "/C=US/ST=IL/L=Chicago/O=IBM Corporation/OU=IBM Software
Group/CN=John Smith/emailAddress=smith@abc.ibm.com" -new > johnsmith.cert.csr
```

- b. Create a private key file without a password.

Example

```
openssl rsa -passin pass:abc -in privkey.pem -out johnsmith.key
```

- c. Create a new X.509 certificate for the new user, digitally sign it using the user's private key, and certify it using the CA private key. The following command line creates a certificate which is valid for 365 days.

Example

```
openssl x509 -req -in johnsmith.cert.csr -out johnsmith.cert -signkey johnsmith.key -CA
waipio.ca.cert -CAkey waipio.ca.key -CAcreateserial -days 365
```

- d. **Optional:** Create a DER-encoded version of the public key. This file contains only the public key, not the private key. Because it does not contain the private key, it can be shared, and does not need to be password protected.

Example

```
openssl x509 -in johnsmith.cert -out johnsmith.cert.der -outform DER
```

- e. Create a PKCS#12-encoded file. The following command line sets the password on the P12 file to default.

Example

```
openssl pkcs12 -passout pass:default -export -in johnsmith.cert -out johnsmith.cert.p12 -inkey
johnsmith.key
```

Repeat this step to create as many digital certificates as needed for testing. Keep the key files secure, and delete them when they are no longer needed. Do not delete the CA private key file. You need the CA private key file to sign certificates.

Results

Now you can install the CA certificate (`waipio.ca.cert`) into WebSphere®. Optionally, create a user certificate specifically for your web server, and install it into WebSphere®.

You can use user certificates individually to record tests. To use the user certificates (`johnsmith.cert.p12`) during test editing and playback, compress them in ZIP format to a file with the `.rcs` extension. This creates a digital certificate store. To learn more digital certificate stores, see [Creating a digital certificate store on page 234](#). You can also import user certificates into your web browser to interactively test them in your environment.

Creating a digital certificate store

The KeyTool command-line program enables you to create a Rational® Certificate Store (RCS) file that contains digital certificates for use with tests. A Rational® Certificate Store (RCS) file is a compressed archive file that contains one or more PKCS#12 certificates. You can also use the KeyTool program to remove certificates from a certificate store.

About this task

Rational® Performance Tester acts as a proxy between the browser and the server application to record the data exchange. When a secured page is recorded using Rational® Performance Tester, the proxy certificate of the product is presented to the browser.

1. In the command line tool, navigate to the directory that contains the Keytool utility. By default, the utility is located at `C:\Program Files\IBM\SDP\jdk\jre\bin`.
2. Type the following command:

Example

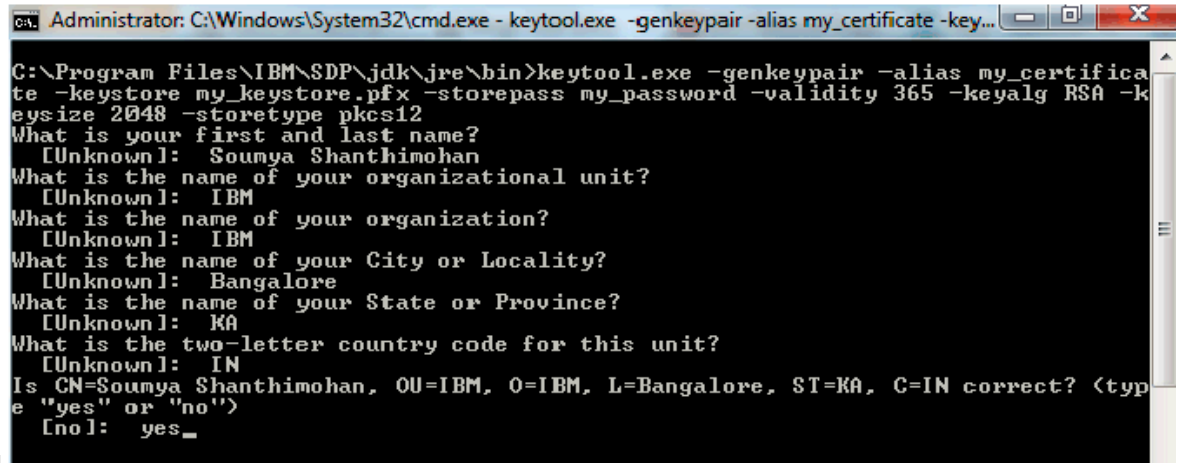
```
keytool.exe -genkeypair -alias certificateName -keystore keystoreName -storepass password -validity 365
-keyalg RSA -keysize 2048 -storetype pkcs12
```

For additional information about parameters by certificate generation, review the official [keytool documentation](#).

Option	Description
<code>-genkeypair</code>	Generate public and private keys for key pair.
<code>-alias</code>	Alias for your certificate in the key store. You may never use it, but every new certificate in your key store must have its own alias.
<code>-keystore</code>	Name of the key store file, which will be generated as the result of the command. It holds your certificate and a corresponding private key. You can reuse this key store for next certificates that you might generate. One key store can contain many certificates.
<code>-storepass</code>	Password that protects your key store file. You will have to enter it every time you want to sign a document.
<code>-validity</code>	Number of days the certificate is valid. You can enter more than 365.
<code>-keyalg</code>	Algorithm to generate the cryptographic keys that is corresponding to your certificate. You can use RSA or DSA.
<code>-keysize</code>	Length of the cryptographic keys. The more the length the stronger the signature.
<code>-storetype</code>	Format of the key store file. PKCS#12 (a.k.a PFX) key stores can be understood by a lot of different programs and you can also import a PKCS#12 file in

Option	Description
	your Windows key store (just double click it and follow the instructions).

3. The certificate generation process prompts you to enter some information about you. Enter the information as



```

Administrator: C:\Windows\System32\cmd.exe - keytool.exe -genkeypair -alias my_certificate -keystore my_keystore.pfx -storepass my_password -validity 365 -keyalg RSA -keysize 2048 -storetype pkcs12
C:\Program Files\IBM\SDP\jdk\jre\bin>keytool.exe -genkeypair -alias my_certificate -keystore my_keystore.pfx -storepass my_password -validity 365 -keyalg RSA -keysize 2048 -storetype pkcs12
What is your first and last name?
  [Unknown]: Soumya Shanthimohan
What is the name of your organizational unit?
  [Unknown]: IBM
What is the name of your organization?
  [Unknown]: IBM
What is the name of your City or Locality?
  [Unknown]: Bangalore
What is the name of your State or Province?
  [Unknown]: KA
What is the two-letter country code for this unit?
  [Unknown]: IN
Is CN=Soumya Shanthimohan, OU=IBM, O=IBM, L=Bangalore, ST=KA, C=IN correct? (type "yes" or "no")
  [no]: yes_

```

prompted.

4. If prompted for a password when using the keystore, enter the same password as provided on the command line.

Result

The key store file (.pfx) is stored in your current directory.

Results

You now have a digital certificate store that you can use with tests. Because the KeyTool program has many options, you might want to create an alias or script file to use to invoke KeyTool. Use KeyTool to create and add as many digital certificates as you want. If you want to create a dataset of the names of certificates in the certificate store, run KeyTool again with the `-list` option. This option writes a list of names that can then be imported to a dataset.

What to do next

Before you start recording the application that requires client certification, import the certificate to the Rational® Performance Tester project. For information about how to import the certificate and record a test, see the Recording a test topic.

You do not have to use the KeyTool command-line program to create a certificate store. It is possible to use existing PKCS#12 certificates with Rational® Performance Tester. PKCS#12 certificates can be exported from a web browser. PKCS#12 certificates encode the private key within the certificate by means of a password.



Note: Do not use certificates associated with real users. Certificates associated with real users contain private keys that should not become known by or available to anyone other than the owner of the certificate. An intruder who gained access to the certificate store would have access to the private keys of all certificates



in the store. For this reason, you must create, or have created for you, certificates that are signed by the correct certificate authority (CA) but that are not associated with real users.

Playing back a test with a digital certificate

After you create a digital certificate store and record a test using a digital certificate, you must associate the digital certificate with the test for playback.

Before you begin

You need to record a test using a digital certificate, and you need a digital certificate store file containing one or more PKCS#12 certificates.

If your certificate extension is not .rcs, then you need to zip the certificate, rename the extension to .rcs, and copy it to the root directory of the project.

To associate a digital certificate with a test for playback:

1. Open the test for editing.
2. On the Security tab, under Digital Certificates, click **Add**.
3. Select or type the name of the certificate store file that you created previously.
You must type or select the file name. You cannot browse to locate the file. The certificate store must be a Rational® Certificate Store (RCS) file. A Rational® Certificate Store file is a compressed archive file that contains one or more PKCS#12 certificates.
4. Select the digital certificate that you want to use, and then click **Select**.
5. When prompted to place the digital certificate in a dataset, click **No**. To learn more about substituting digital certificates, see Using a digital certificate store with a dataset.

Result



Note: If you add multiple certificates to the **Digital Certificates** list on the Common Options page, the first certificate that satisfies the request from the server (in the order by which the certificates were entered) is used during playback.

6. Save the test.

Results

When you run this test, the digital certificate from the certificate store is submitted to the server.

Entrust TruePass authentication overview

Entrust provides digital identity and encryption technologies to governments and private industry. With Entrust TruePass software users can authenticate with secure web applications without installing a digital certificate in their browsers. This makes it convenient for use in kiosks and other public user environments.

You can now run tests against servers that require Entrust TruePass authentication. Roaming mode with TruePass applet version 7.0 and later are checkmark_perf.jpg. Local mode, and versions of the TruePass applet prior to 7.0, are

not checkmark_perf.jpg. Recording tests with Entrust TruePass applications works just as regular HTTP recording does.

The Entrust TruePass Authentication object is displayed in the test editor for tests that you record with Entrust TruePass applications. The **Version** field displays the recorded version number of the Entrust TruePass applet. The **Server Name** and **Port** are correlated fields. Click **Substitute** to use the **Data Sources** view to change the server or port number for playback. The **Application Context** displays where the Entrust application is mapped to in the application server. The **User Name** and **Passphrase** fields can be substituted with values from a dataset.



The screenshot displays the Test Editor interface. On the left, the **Test Contents** pane shows a tree view of the test structure. The selected element is **Entrust Authentication (Mode: Roaming)**. To the right of the tree are buttons for **Add**, **Insert**, **Remove**, **Up**, **Down**, **Prev**, and **Next**. Below the tree is a **Common properties** section. On the right, the **Test Element Details** pane shows the configuration for the selected element. It includes a **Version** field set to 8.0, a **Connection Information** section with **Server Name** (www.entrust.com) and **Port** (443), and an **Authentication Information** section with **User Name** (testuser) and **Passphrase** (testPassphrase99). A **Substitute** button is located next to the Port field.





Annotating a test during recording

You can add comments, add transactions, or change a page name while you record a test. The advantage of adding these elements during (rather than after) recording is that you can place the annotations in the test exactly where you want. In addition, because annotations are part of the recorded test, they are regenerated when you regenerate the test. You can also insert split points into a test during record.

1. Start recording the test. The **Recorder Test Annotations** toolbar opens near the top of the screen.
2. Click the appropriate icon.

You can use the **Recorder Test Annotations** toolbar to add comments, record synchronizations, or take screen captures during the recording.

- To add a comment to the recorded test, click the **Insert comment** icon . You are prompted for a comment.
- To add a screen capture to the recorded test, click the **Capture screen** icon . Screen and window captures make your tests easier to read and help you visualize the recorded test. You can change the settings for screen captures and add a comment to the image.

- To manually add a transaction folder to the recording, click the **Start Transaction** icon  and **Stop Transaction** icon  to start and stop the transaction. Transactions can be nested.
 - To insert a split point into the recorded test, click the **Split point** icon . With split points, you can generate multiple tests from a single recording, which you can replay in a different order with a schedule. See [Splitting a test during recording on page 240](#) for more information about splitting a test.
 - When recording an HTTP test, to change the page name, click the **Change page name** icon . In the resulting test, the page element in the test editor uses the new name, however the original name is preserved in the **Page Title Verification Point** area so that page title verification points still work correctly.
3. Close the client program to stop the recording.
 4. If you inserted a split point during the recording, on the **Destination** page, in the **Test Generation** wizard, specify the location for the split test or merge the split recordings together.


Results

The test is generated with the comments, transactions, and page names that you added.

Recording sensitive session data

You can keep recording session (`.recsession`) files to view the contents of a recording or to regenerate tests. However, if a recorded test contains sensitive information, you can choose to obfuscate, or encrypt, text strings in the `recsession` file.

To protect test data in a recording session file:

1. In the Performance Test perspective, click the **New Test from Recording** toolbar button  or click **File > New > Test from Recording**.
2. In the **New Test from Recording** window, select **Create a Test from a New Recording**, and select the type of test to create.
3. In **Recording encryption level**, select one of these options:

Choose from:

 - **Obfuscated:** This setting hides text strings to prevent viewing the raw data in `recsession` files with a text editor outside of the workbench. You can still use `recsession` file to generate tests and to view recording information.
 - **Passphrase:** This setting uses an AES-128-bit algorithm to encrypt text strings in the `recsession` files. The encryption strength depends on the length of the passphrase. The recording session file is unrecoverable if the passphrase is lost.
4. On the **Select Location** page, select the project and folder locations to contain the new test, type a name for the test, and click **Next**.
5. If you selected **Passphrase**, on the **Passphrase Protection** page, type the passphrase twice in **Passphrase** and **Confirm passphrase**.



For solid protection, make the passphrase longer than 24 characters if using English words or at least 12 random characters.

6. Click **Next**, and continue the recording session for the type of test that you selected.

Splitting a test during recording

You can insert split points when you record a test. Split points allow you to generate multiple tests from a single recording that you can replay in a different order with a schedule.

To split a test during recording:

1. Start recording the test. The **Recorder Test Annotations** toolbar opens near the top of the screen.
2. To insert a split point into the recorded test, click the  **Split point** button. . The **Insert Split Point** window is displayed.
3. Type a name for this section of the test and click **OK**. You are naming the previous section of the test, not the upcoming section of the test.
Repeat this step between recorded user actions as needed to split tests.
4. After you finish performing the user tasks in the client program, stop the recorder. You can do this by closing the client program or by clicking the **Stop**  button in the **Recorder Control** view.
If you changed the network settings of the client program as described in step 8, you can revert them to the default settings before closing the program.

Result

The **Generate Service Test** wizard opens.

5. On the **Destination** page, specify the location for the split test or merge the split recordings together:
 - In **Location**, click **Browse** to specify the folder where the split tests are generated.
 - Type a **Test prefix** that will be appended to the name of each split test. Leave blank if you do not want the split test names to have a prefix.
 - In the split test list, mark the split tests that you want to generate. Click **Select All** to generate all split tests or **Unselect All** to clear the list.
 - To merge several split tests into a single test, multi-select the tests that you want to merge by holding the **Shift** key and click the **Merge** button.
6. Click **Finish**.

Results

The tests are generated using the test names that you specified.

Generating a new test from a recorded session

You can generate a new test from a recorded session. For example, if you accidentally damage a test during editing, or if you want to change a test preference, you can regenerate the test instead of re-recording it. If split points were inserted in the recording, you can choose to generate a single test without split points.

To regenerate a complete test from a recording that contains split points:

1. In the test navigator, select the .recession file of the test recording to regenerate.
2. Right-click, and then select **Generate Test**.

Result


The **Generation Test** wizard is displayed.

3. If the .recession file is compatible with multiple test types, select the type of test that you want to generate and click **Next**.

Example

For example, select **HTTP Test** to generate an HTTP performance test.

4. On the **Select Location** page, select the project and folder where you want to create the test, type a name for the test, and click **Next**.

If necessary, click  **Create the parent folder** to create a new performance test project or a folder

5. If the .recession file contains split points, on the **Options** page, select **Generate test without split points** if you want to regenerate the test as a single test.
6. Click **Finish**.

Results


The test is regenerated and opened in the test editor.

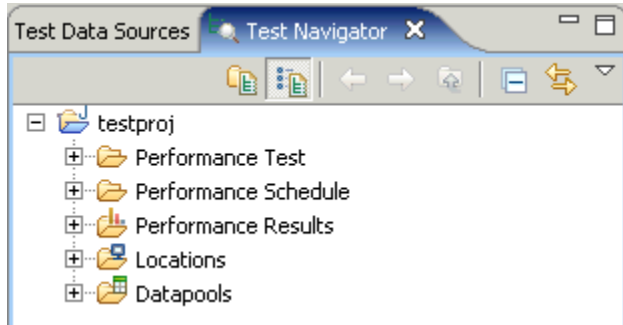
Organizing test assets by type

By clicking an icon, you can view your test assets in a logical order, in separate folders for tests, schedules, results, locations, and datasets.

About this task

In the Test Navigator view, you can click **Show Java Content** to see custom code that you created, click **Show Missing Resources** to view unresolved references, and click **Show File Extensions** to view file extensions of test assets.

1. On the Test Navigator toolbar, click the **Show the logical test navigator**  icon.
To see the Test Navigator view, click **Windows > Show view** and click **Test Navigator**.
2. Your assets are now grouped logically. To see them, open the appropriate folder.



Note: In the Logical view, only if the asset is available the appropriate folder is displayed. For example, if the Results folder is displayed only after you have executed a test.

Editing tests

After you record a test, you can edit it to include datasets (to provide variable data rather than the data that you recorded), verification points (to confirm that the test runs as expected), and data correlation (to ensure that returned data is appropriate for the corresponding request). You can also add protocol-specific elements to a test. When you edit a test, the modified items appear in italic type. The italic type changes to regular type after you save the test.

Editing service tests

After you record a service test, you can edit the calls and message returns to include variable data (rather than the data that you recorded). You can add verification points (to confirm that the test runs as expected), transactions, conditional processing, and custom code.

Web service test editor overview

With the test editor, you can inspect or customize a test that you recorded.

The test editor lists the web service call elements for a test, in sequential order.

There are two main areas in the test editor window. The area on the left, **Test Contents**, displays the chronological sequence of test elements in the test. The area on the right, **Test Element Details**, displays details about the currently selected item (test, call, message return, or verification point) in the test hierarchy.

Window events are the primary test elements in a Citrix test and represent graphic objects that are drawn by the Citrix server, such as actual window, dialog boxes, menus, or tooltips. A Window event is recorded each time a window is created, destroyed, moved, or resized. The first occurrence of a window, a create window event, is displayed in bold. Window objects are typically identified by their title. If there is no window title, for example on menus or tooltips, then the test editor uses the window ID number.

A service request node name can be updated automatically or you can use custom code or dataset to supply different names. To apply a dataset to a node name, in the test editor, select the node name. In the Request Details area of the test editor, clear the **Update node name automatically**, select the name and substitute it with dataset.

Web service calls can contain web service message return elements, which display the results of the web service call. The XML message content can be displayed either in Form, Tree or Source view. Each of these views displays the same message content in different forms:

- **Form** view provides a simplified view of the call elements focused on editing the values of the XML message content.
- **Tree** view provides a hierarchical view of the XML structure, including elements, namespaces, and the associated values. Tree view also allows you to manipulate XML fragments.
- **Source** view displays the XML contents of a web service or XML call or the plain text contents of a simple text message.

Message return elements can contain verification point elements that check that the actual return results match expected criteria.

Some actions contain data that is highlighted. This highlighting indicates that the data can be used as a dataset candidate or as a reference. See [Data correlation overview on page 315](#) for more information.

In service calls and message returns, you can use datasets and data correlation on values contained in the XML or on XML fragments. To use data correlation on XML fragments, switch to the Tree view, right-click the XML element and select **Create XML Fragment**.

To view or modify the color coding in web service tests, click **Window > Preferences > Test > Test Editor**, and then click the **Fonts and Colors** tab.

Click **Add** to add elements to the selected test element. Alternatively, you can right-click a test element and select an action from a menu.

The choices that you see depend on what you have selected. For example, inside a web service call, you can add a web service message return. The **Insert** button works similarly. Use it to insert an element before the selected element. The **Remove** button allows you to delete an item.

Verifying application behavior

To check the expected behavior of the application during a service test, you can add verification points after a message return. During the run, verification points produce a pass, fail, error, or inconclusive status in the Web Service Verification Point report.

Adding equal verification points

Equal verification points enable you to check that the contents returned by a service match exactly the contents specified in the verification point.

About this task

When you add verification points, the results from a service response are compared with the expected data specified in the verification point test element. *Equal* or *contain* verification points enable you to directly compare the XML document that the service returns.

- Contain verification points return a Pass status when the response XML document contains the expected XML data.
- Equal verification points return a Pass status when the response XML document matches exactly the expected XML data.

Complex service requests or verification points might have empty XML elements that are not needed in a test script. When playing back the test, you can skip such empty XML elements. In **Window > Preferences > Test > Test editor > Service test** ensure that the **Display the 'Skip if empty' column in XML tree viewer** check box is selected. This option displays a **Skip if empty** column in the tree view of the request. You can then choose the XML elements to skip.

1. Open the test editor, and right click a response element and select **Add > Equal Verification Point**.
2. Select the verification point, and in the **Test Element Details** area of the test editor, type a name for the verification point.
3. Select the verification options:
 - Select **Test using XML namespaces** to perform the verification on the qualified structure of the XML document, including the namespace tagging, instead of the simple name. Disable this option to check only the simple name of the element and the final return value.
 - Select **Text XML text nodes** to include the content of text elements in the verification.
 - Select **Text XML attributes** to include the content of attributes in the verification.
4. On the Message page, select the **Form**, **Tree**, or **Source** view to specify the expected XML data.

For an equal verification point, the expected XML data contains the XML document from the response test element. If necessary, you can edit the expected XML data.

You can specify standard Java™ regular expressions in the **Tree** view. To do this, select the **Regular expression** column on the line of an attribute or text value and type the regular expression in the **Value** column. For example, the following regular expression checks for a correctly formatted email address: `/^[a-zA-Z0-9_\.\\-]+@[a-zA-Z0-9_\.\\-]+([a-zA-Z0-9]{2,4})+$/`

When using regular expressions, the number of XML nodes or XML fragments in the verification point must match the quantity of expected nodes.

What to do next

You can enable or disable each verification point by right-clicking the verification point in the test editor and clicking **Enable** or **Disable**.

Adding contain verification points

With contain verification points, you can check that one or several elements of the XML content returned by a service match the XML fragment that is specified in the verification point.

About this task

When you add verification points, the results from a service response are compared with the expected content that is specified in the verification point test element. *Equal* or *contain* verification points enable you to directly compare the XML contents that the service returns.

- Contain verification points return a Pass status when the response XML contents contain the expected XML fragment.
- Equal verification points return a Pass status when the response XML contents match exactly the entire expected XML content.

Complex service requests or verification points might have empty XML elements that are not needed in a test script. When playing back the test, you can skip such empty XML elements. In **Window > Preferences > Test > Test editor > Service test** ensure that the **Display the 'Skip if empty' column in XML tree viewer** check box is selected. This option displays a **Skip if empty** column in the tree view of the request. You can then choose the XML elements to skip.

1. Open the test editor, and select a service response element.
2. In the **Test Element Details** area, click the **Message** tab and select the **Form** or **Tree** view.
3. Expand the envelope line, right click the element that you want to check, and then click **Create Contain Verification Point**. This action creates a contain verification point that includes the XML element from the recorded response.



Note: You can also create a contain verification point with the message response by selecting the message response in the **Test Contents** pane and clicking **Add > Contain Verification Point**. However, the result is effectively the same as an equal verification point because the verification point contains the entire XML content of the message response.

4. Select the verification point, and in the **Test Element Details** pane, type a name for the verification point.
5. Select the verification options:
 - Select the **Test using XML namespaces** check box to perform the verification on the qualified structure of the XML document, including the namespace tagging, instead of the simple name. Disable this option to check only the simple name of the element and the final return value.
 - Select the **Test XML text nodes** check box to include the content of text elements in the verification.
 - Select the **Test XML attributes** check box to include the content of attributes in the verification.
6. If necessary, select the **Form**, **Tree**, or **Source** views to edit the expected XML fragment.

For an equal verification point, the expected XML data contains the XML document from the response test element. If necessary, you can edit the expected XML data.

You can specify standard Java™ regular expressions in the **Tree** view. Select the **Regular expression** column on the line of an attribute or text value and type the regular expression in the **Value** column. For example, the following regular expression checks for a correctly formatted email address: `/^([a-zA-Z0-9_\.\\-])+@(([a-zA-Z0-9\\-])+\.)+([a-zA-Z0-9]{2,4})+$/`

When using regular expressions, the number of XML nodes or XML fragments in the verification point must match the number of expected nodes. The verification point returns a Pass status when all regular expressions in the XML fragment are matched.

Example

You can use a contain verification point to check that the message response contains only a specific element with a specific value. For example, consider the following message response:

```
<s:Envelope
  xmlns:a="http://www.w3.org/2005/08/addressing"
  xmlns:s="http://www.w3.org/2003/05/soap-envelope">
  <s:Header>
    <a:Action
      s:mustUnderstand="1">http://www.w3.org/2005/08/addressing/soap/fault</a:Action>
    <a:RelatesTo>uuid:ed9bc447-d739-452f-989d-cd48344d494a</a:RelatesTo>
  </s:Header>
  <s:Body>
    <s:Fault>
      <s:Code>
        <s:Value>s:Sender</s:Value>
        <s:Subcode>
          <s:Value
            xmlns:a="http://schemas.xmlsoap.org/ws/2005/02/sc">a:BadContextToken</s:Value>
          </s:Subcode>
        </s:Code>
        <s:Reason>
          <s:Text
            xml:lang="en-US">The message could not be processed. This is most likely because the action
            &apos;http://Samples.ICalculator/Add&apos; is incorrect or because the message contains an invalid or
            expired security context token or because there is a mismatch between bindings. The security context
            token would be invalid if the service aborted the channel due to inactivity. To prevent the service
            from aborting idle sessions prematurely increase the Receive timeout on the service endpoint&apos;s
            binding.</s:Text>
          </s:Reason>
          <s:Node>http://www.w3.org/1999/xlink</s:Node>
          <s:Role>http://www.w3.org/1999/xlink</s:Role>
          <s:Detail
            xmlns:tns0="http://schemas.com/2003/10/Serialization/"
            xmlns:tns15="http://Samples.Windows"
            tns0:Id="id"
            tns0:Ref="idref">
            <tns15:GetCallerIdentityResponse>
              <tns15:GetCallerIdentityResult>str</tns15:GetCallerIdentityResult>
            </tns15:GetCallerIdentityResponse>
          </s:Detail>
        </s:Fault>
      </s:Body>
    </s:Envelope>
```

To check for the `Subcode` element, the expected content of the contain verification point is the following XML fragment:

```
<s:Subcode
  xmlns:a="http://www.w3.org/2005/08/addressing"
  xmlns:s="http://www.w3.org/2003/05/soap-envelope">
  <s:Value
    xmlns:a="http://schemas.xmlsoap.org/ws/2005/02/sc">a:BadContextToken</s:Value>
</s:Subcode>
```

By default, the contain verification point checks whether an element named `Subcode` contains one element named `Value`. You can use the following options:

- **Test using XML namespaces:** With this option, the verification point checks whether an element named `"http://www.w3.org/2003/05/soap-envelope":SubCode` contains one element named `"http://www.w3.org/2003/05/soap-envelope":Value`.
- **Test XML text node:** With this option, the verification point also checks whether the element named `Value` contains the text `a:BadContextToken`.
- **Test XML attributes:** With this option, the verification point also checks that the attributes match the expected XML fragment. In this example, the **Test XML attributes** option is not necessary because the `Subcode` element does not have any attributes.

To check that the `Detail` element properly returns a specific value for `GetCallerIdentityResult`, the expected content of the contain verification point is the following XML fragment:

```
<s:Detail
  xmlns:a="http://www.w3.org/2005/08/addressing"
  xmlns:s="http://www.w3.org/2003/05/soap-envelope"
  xmlns:tns0="http://schemas.com/2003/10/Serialization/"
  xmlns:tns15="http://Samples.Windows"
  tns0:Id="regular_expression"
  tns0:Ref="idref">
  <tns15:GetCallerIdentityResponse>
    <tns15:GetCallerIdentityResult>IdentityValue</tns15:GetCallerIdentityResult>
  </tns15:GetCallerIdentityResponse>
</s:Detail>
```

You can use the following options:

- **Test XML text node:** With this option, the verification point also checks whether the element named `GetCallerIdentityResult` contains the text `IdentityValue`.
- **Test XML attributes:** With this option, the verification point also checks that the attribute `Id` referred to by `tns0:Id` has the expected value. You can specify a regular expression for this value by using the **Regular expression** column in the **Tree** view of the verification point. For example, `tns0:Id="[a-zA-Z]"` checks that the value does not contain numbers.

What to do next

You can enable or disable each verification point by right-clicking the verification point in the test editor and clicking **Enable** or **Disable**.

Adding Xpath query verification points

With service query verification points, you can check that a response matches an XPath query.

Before you begin

When you add verification points, the results from a service response are compared with the expected data that is specified in the verification point test element. With *query* verification points, you can check that the number of nodes returned by an XML Path language query matches the expected number of nodes specified in the verification point.

Refer to the XPath specification for details on expressing an XPath query: <http://www.w3.org/TR/xpath>.

You can use the test editor to create or edit verification points.

1. Open the test editor, and select a web service response element.
2. Click **Add**, and select **Query verification point**.
3. In the **Test Element Details** area of the test editor, type a name for the verification point.
4. Type a valid **XPath expression** or click **Build Expression** to open the **XPath Expression Builder**.
The **XPath Expression Builder** helps you build and evaluate XPath expressions based on the recorded contents of the response.
5. Specify a **Comparison operator** (=, >, or <), and the expected number of nodes that the query should return.
Click **Evaluate** to update the Expected Count with the actual result based on the recorded contents of the response.

What to do next

You can enable or disable each verification point by right-clicking the verification point in the test editor and clicking **Enable** or **Disable**.



Note: Because XPath expressions require that the qualified name have a prefix, XPath expressions will return null for the default namespace declared with *xmlns*.

Adding attachment verification points

Service attachment verification points enable you to check that the attachment of a service response matches the specified criteria.

Before you begin

When you add verification points, the results from a service response are compared with the expected data that are specified in the verification point test element. *Attachment* verification points enable you to verify that an expected attachment is delivered with the response.

Attachment verification points return a Pass status when all the criteria of an attachment match the expected criteria specified in the verification point test element. If any of the criteria do not match, the verification point returns a Fail status.

You can use the test editor to create or edit verification points.

To add attachment verification points to a performance test:

1. Open the test editor and select a service response element.
2. Click **Add** and select **Attachment Verification Point**.
3. In the **Test Element Details** area of the test editor, type a name for the verification point, and specify the criteria to be verified. All criteria must match in order for the verification point to pass.

- a. In the case of multiple attachments, set the **Index of attachments** to the index number of the attachment to be checked. Type `1` if there is only one attachment in the response.
- b. Specify the expected size in bytes of the attachment.
- c. Specify the MIME type and encoding of the attachment.

What to do next

You can enable or disable each verification point by clicking **Enable verification point** in the test editor.

Adding Text verification points

To check the text content that is returned by the service response, you can add a text verification point in the service test. When you add the verification point, you can check whether the text matches equally with the response or whether the response contains the text.

1. Open the Test editor, right-click a response element and select **Add > Text Verification Point**.
2. In **Verification Point Name**, specify a name for the verification point.
3. In the **Operator** field, select the basis of comparison between the text to be verified and the response content.
4. To search between the offset values, select **From Offset** and **To Offset** check boxes and specify the offset values.
5. To search between two string values, select **From String** and **To String** check boxes and specify the strings. You must also specify the number where the strings occurred.

For example, if there are four occurrences of 'My Text' in the content and you want to verify the text that is between second and third occurrence, you should specify 2 and 3 in **From String** and **To String** respectively.

6. To do a case-sensitive match, select the **Case sensitive** check box.
7. To ignore carriage return/ line feed in the response, select the **Ignore CL/LF when matching** check box.
8. Save the test and run it.

Results

The Service Verification Point Report shows the number of Text Verification Points that passed or failed.

Adding properties verification points to a test response

You can add verification points for the properties in a service test so that these properties in the test response are verified and validated when you play back the test.

Before you begin

You must have recorded or created a service test using the test editor.

About this task

When you add verification points, results from a service response are compared with the expected data specified as the verification point test element. You can add the verification point for the properties to an existing test response when the test is manually created or recorded. After you add the verification point for the properties to a test response, you can verify the selected response properties during the test run.

1. Identify the service test from **Test Navigator** and double-click the service test to open it in the test editor.
2. Select a service response for a service request from the service test.
3. Right-click the service response, click **Add > Properties Verification Point**.


Result

The **Properties Verification Point** is added based on the existing properties of the service response.



Note: You can add multiple verification points for the properties, if required.

4. Perform any of the following on the verification points for the properties in the **Properties Verification Point Details** pane.

To Do...	Do This...
To add a new property and its value	Click Add .
To edit the value of an existing property	Click Edit .
To remove the property that you do not want to verify during the test run	Click Remove .  Note: You can remove multiple properties in a group at the same time.

5. Select or clear the **Apply And Operator** check box based on the requirement as follows:
 - To verify all the listed properties, select the **Apply And Operator** check box.
 - To verify one of the listed properties, clear the **Apply And Operator** check box.
6. Optionally, you can substitute the value of one or more properties in the verification point by using a test variable, data set, custom java code, or built-in variables.
7. Verify all the verification points for the properties that you entered, and then click **Save**.

Results

The verification points that you added for the response properties are added to the service test.

What to do next

You can run the test and after the test run, you can view and analyze the `properties verification point` details from the following page and reports:

- **Verdict List** pane in the **Test Log** page. Click any of the verification point from the list and view the details.
- **Response Properties Verification Points** tab in the **Service Verification Point Report**. Click the **Response Properties Verification Points** tab and view the verification point details.
- **Verification points verdicts** pane from the **Functional Test** report page. Click any of the verdict status to verify the expected value and actual value of the verification point for the properties.

Related reference

[Web Service Verification Points report on page 517](#)

Adding XSD verification points

XSD verification points enable you to check that the XML content of a service response comply with the rules defined in an XML Schema Definition (XSD) file.

Before you begin

When you add verification points, the results from a service response are compared with the expected data that are specified in the verification point test element. XSD verification points return a Pass status when the XML contents of the response are compliant with the associated XSD or a Web Service Description Language (WSDL) file that contains XSD information.

If you add multiple XSD files to the verification, then the XML content of the response must comply with all of the XSD files.

You can use the test editor to create or edit verification points.

To add an XSD verification point to a test:

1. Open the test editor and select a service response element.
2. Click **Add** and select **XSD Verification Point**.
3. In the **Test Element Details** area of the test editor, type a name for the verification point.
4. Click **Add XSD** to add a an XSD file to the validation list or **Add WSDL** to add a WSDL that contains XSD information.
Click **Open** to display the XSD or WSDL contents.

What to do next

You can enable or disable each verification point by right-clicking the verification point in the test editor and clicking **Enable** or **Disable**.

Working with Server Name Indication (SNI) recordings

If you have recorded against a server that supports Server Name Indication (SNI), an extension of the TLS protocol, the recording session file displays `true` for the **SNI Extension** field. There might be a need for you to access both SNI and non-SNI applications from the same server. To run the same test without using the SNI extension, you can manually change the value to `false`.

About this task

The screenshot displays the 'Packets' and 'Packet Details' panels. The 'Packets' panel shows a list of network events with columns for Packet, Start Time, and End Time. The 'Packet Details' panel shows the configuration for a 'Proxy Open Secured' connection, including Recorder, Connection Id, Parent Connection Id, Connection Type Id, Server Host, Server Port, Client Host, Client Port, Cipher Suites, SSL Protocols, HTTP Version Protocol, and SNI Extension. The 'SNI Extension' field is highlighted with a red box and contains the value 'true'.

Packet	Start Time	End Time
Proxy connection type 0	00:00.069	00:00.069
Local addresses	00:00.308	00:00.308
Proxy starts	00:00.310	00:00.310
Proxy initiates connection 1	00:06.271	00:06.271
Proxy opens connection 1	00:06.271	00:06.611
Proxy opens secured connection 2 based on 1	00:06.758	00:07.357
Proxy close connection 2	00:07.551	00:07.551
Proxy close connection 1	00:07.551	00:07.551
Proxy initiates connection 3	00:10.195	00:10.195
Proxy opens connection 3	00:10.195	00:10.259
Proxy opens secured connection 4 based on 3	00:10.268	00:10.408
Proxy close connection 4	00:10.703	00:10.703
Proxy close connection 3	00:10.703	00:10.703
Proxy initiates connection 5	00:10.788	00:10.788
Proxy opens connection 5	00:10.788	00:11.066

Field	Value
Recorder:	HTTP Proxy Recorder
Connection Id:	2
Parent Connection Id:	1
Connection Type Id:	0
Server Host:	tiles.services.mozilla.com
Server Port:	443
Client Host:	127.0.0.1
Client Port:	54790
Cipher Suites:	SSL_ECDHE_RSA_WITH_AES_128_GCM_SHA256
SSL Protocols:	TLSv1.2
HTTP Version Protocol:	http/1.1
SNI Extension:	true

The **Server Access Configurations** resource of the test script also have SSL entries. Each SSL entry displays which TLS version and Cipher value was used. To edit multiple SSL entries, select them and in the Detail area, right-click the entries and click **Edit multiple SSLs**.

Adding elements to a socket test

A socket test provides the flexibility to add close, send, and receive elements to the test.

Adding a service request

You can use service request elements in tests to send a request to the service.

About this task

Complex service requests or verification points might have empty XML elements that are not needed in a test script. When playing back the test, you can skip such empty XML elements. In **Window > Preferences > Test > Test editor > Service test** ensure that the **Display the 'Skip if empty' column in XML tree viewer** check box is selected. This option displays a **Skip if empty** column in the tree view of the request. You can then choose the XML elements to skip.

1. Open the test in the test editor, and select the first element in the test.
2. Click **Add** and select a service request.
3. If you selected WSDL service request, select one or several WSDL files in your workspace for the web service that you want to test and click **Next**.
If necessary, you can import a WSDL file into the workspace with the **Add** button.
4. Select either **HTTP**, **JMS**, or **WebSphere MQ** depending on the transport protocol used by the web service, and provide the correct transport protocol configuration to perform the call.
You can create a **New** transport configuration or reuse an existing one.
5. Click **Finish**.

Result

This creates the web service request in the test editor.

6. On the **Message** page of the request, select the **Form**, **Tree**, or **Source** views to edit the service request contents.
7. If any resource files are to be attached to the request, select the **Attachment** tab. Use **Add**, **Remove**, or **Edit** to specify the resources that are to be attached to the request.
8. If the service uses encryption, signature or other security protocols, select the **Security for Request** and **Security for Response** pages to configure the security for this particular service request or to open the WSDL security editor.

What to do next

After creating elements, you can use the test editor to edit service requests. You can create a service response element to test the performance and behavior of the service. You can also replace some content values with dataset variables or a references.

Updating a service response from the service

While you are developing a service test, you can send a request from the test editor to record or update the response element.

Before you begin

Service response elements are children of service request elements. Service tests use response elements to measure the response time between a call and the corresponding response. Response elements can also contain verification points.

You can click **Update Response** in the request element to complete one of the following actions:

- Record a response from the service: This method sends the request and records the actual response from the service. For services that use the IBM® WebSphere® MQ or JMS transport protocols, multiple responses can be recorded.
- Update the current response content: If a response exists, its contents are replaced. If multiple responses are received, the number and order of the responses are updated.



Important: After updating the response content, data correlation or verification points that referred to replaced content might no longer work.

You can use the test editor to create or edit response elements in a service test. There are three methods of adding a service response:

- Generate a response from Web Services Description Language (WSDL): If the service uses WSDL, then the response is created with the content structure that the WSDL specifies.
- Add a text response: In this response type, you specify free formatted content for the response.
- Record a response from the service: This method sends the request and records the actual response from the service.

WebSphere® MQ and JMS requests can contain multiple response elements.

To add a response element to a service test:

1. Open the test in the test editor, and select a service request element.
2. On the **Test Element Details** page, click **Update Response**.
Alternatively, right-click the service request element, and click **Add > Response from Request**.
Result
This action performs the service request. If the request is valid, the **Update Response** window opens and displays the response data.
3. In the **Return Preview** window, review the content of the response to ensure that it is correct.
For the WebSphere® MQ and JMS protocols, if multiple responses are received, then click the arrows to view each response.
 - a. Click the **Message** tab to view the contents of the response in the **Form**, **Tree** or **Source** view.
 - b. Click the **Attachment** tab to view any resource files that were attached to the response.
 - c. Click the **Response Properties** tab to view the properties of the response.
4. To use the received response in the test, click **Update Test**.
This creates the response elements as a child of the request element or updates the existing response elements with the new data.

What to do next

After creating or updating response elements, you can create verification points on the response contents to test the behavior of the service.

Related information

[Manually adding a response element on page 254](#)

[Verifying application behavior on page 243](#)

Manually adding a response element

You can add service response elements to specify the received content of a service request. You can use the test editor to create or edit response elements in an existing service test.

Before you begin

Service response elements are children of service request elements. Service tests use response elements to measure the response time between a call and the corresponding response. Response elements can also contain verification points. IBM® WebSphere® MQ and JMS requests can contain multiple response elements.

Depending on the type of request, you can manually create several types of response elements:

- Response from Web Services Description Language (WSDL): For web services, this response type uses the WSDL file to create the specified XML structure of the response.
- XML response: This response type creates an empty response element in which you must manually create the expected XML structure. You can use an XML Schema Definition (XSD) document from the XSD catalog to assist you.
- Text response: This response type creates an empty response element, which can contain freely formatted text.

Alternatively, you can automatically create and update response content by recording the actual response content that the service returns. See [Updating a service response from the service on page 253](#) for more information.

To add a response element to a service test:

1. Open the test in the test editor, and select a service request element.
2. Create one of these elements:

Choose from:

- For web service requests, click **Add > Response from WSDL**.
- If the expected response contains XML content, click **Add > XML Response**.
- If the expected response contains plain text, click **Add > Text Response**.

Result

This action creates the corresponding response element in the test. If the request uses the WebSphere® MQ or JMS format, then you can create multiple responses.

3. Edit the message content of the response element to reflect to actual content that the service returns.
 - a. Click the **Message** tab to view the contents of the response in the **Form**, **Tree** or **Source** view.
 - b. Click the **Attachment** tab to view any resource files that were attached to the response.
 - c. Click the **Response Properties** tab to view the properties of the response.

What to do next

After creating a message return, you can create verification points on the contents to test the behavior of the service.

Related information

[Updating a service response from the service on page 253](#)

[Verifying application behavior on page 243](#)

Managing JMS/MQ connections in a service test

When you run a service test that includes JMS or MQ protocol, the socket connections are created and closed in the background. When you include multiple tests in a compound test or a schedule, in this context, is used to refer to VU Schedule and Rate Schedule, multiple connections are created and closed. Starting from 9.2, when you run service tests in a schedule, you can select a pooling strategy for these JMS/MQ connections so that when the connections are created, they do not close and are reused subsequently for the other JMS/MQ calls, if required.

About this task

You can set the scope of JMS/MQ connections to a test, compound test, or schedule. When you set the scope to test, existing behavior comes into play wherein duplicate connections could be created and closed. When you set the scope to a compound test or a schedule, the connections are reused for JMS/MQ calls within a compound test or a schedule.

1. In the Test Navigator, browse to the schedule and double-click it. The schedule is displayed.
2. Select a schedule. In the VU Schedule Details area, click the **Advanced** tab and under **Protocol-specific options**, click **Edit Options**.
3. In **JMS/MQ connections scope**, select the scope of the connections.
4. Use the following options to control the underlying MQ Connection Manager to create only the specified number of connections. These options are generally used by the MQ expert:
 - a. In **Maximum quantity of connections**, specify a number to ensure that a certain number of connections are open at a time only for MQ Java.
 - b. In **Maximum quantity of unused connections**, specify the maximum number of connections that should be unused among the open connections.
 - c. In **Connection timeout (ms)**, specify a time after which there is no attempt to establish the connection.

Results

When you run a schedule, the JMS/MQ connections are reused.

Editing WSDL security profiles

To ensure that your service test uses the correct security protocols to access a SOAP-based service, you must specify a security profile for the (Web Service Description Language) WSDL file. After a security profile is set up, it can be reused in multiple web service calls.

WSDL security editor overview

With the WSDL security editor you can create the SOAP algorithm stacks that are associated with a web service operation. Algorithm stacks contain digital certificate information and the security algorithms that are applied to messages to perform secure communication with a web service.

After you create an algorithm stack, you associate it with an operation that is specified in the Web Services Description Language (WSDL) file of the web service. Algorithm stacks remain available in the workspace and you can reuse them with other WSDL files. You can also edit a test to make the same web service call several times with different security configurations.

You use the **WSDL security editor** to create and edit security configurations. The WSDL security editor contains two pages that correspond to the steps of setting up a security configuration:

- Describing a security stack
- Associating a security stack with each WSDL operation

Algorithm stacks

Algorithm stacks contain one or several algorithm blocks that are arranged in a sequence of steps. Each algorithm block modifies or transforms the message content. Algorithm blocks can add timestamps to, add tokens to, encrypt, or sign messages.

Use the **Algorithm Stacks** page of the WSDL security editor to create stacks for service requests and responses. When a message is sent or received, each algorithm block in the stack is executed in the specified order. For example, you can define a request stack for outgoing requests that adds a timestamp, signs, and then encrypts the message content, and you can define a response stack that decrypts incoming responses. You can create as many algorithms as your application requires.

You can edit algorithm blocks and move them up and down in the stack. Encryption and signature blocks can use keystores for digital certificates. Some algorithm blocks display messages that help you enter correct information. If the contents of the algorithm block are invalid, an error icon is displayed.

Raw transaction data view

When a stack is associated with a service request or response, viewing the results of each transformation step that is applied to the XML message content can be useful. You can use the **Raw Transaction Data** view to look at the message content before and after each algorithm in the stack.

Digital certificate keystores

You can add digital certificate keystores to a security stack to use with encryption or signature algorithms. Keystores must be declared with their associated passwords before the algorithms that use them. Digital certificates are contained in Java™ keystore files (KS, JKS, JCEKS, PKCS12, and PEM) that must be located in your workspace.

Associating stacks with WSDL operations

Use the **Algorithms by WSDL operations** page of the WSDL security editor to associate a security algorithm stack with each web service call and message return in the WSDL file.

Creating security profiles for WSDL files

You can create SOAP security profiles for the web service calls or message returns that require message encryption, signature or other advanced security algorithms.

Before you begin

You must have a Web Services Description Language (WSDL) file in your workspace.

If the security profile uses digital certificates for encrypting or signing requests or responses, you must have the corresponding keystore files (KS, JKS, JKECS, PKCS12, or PEM) in your workspace.

About this task

If the WSDL is simple and you want to check its security, in the **Request Stack** tab of the test editor, click **Override Stack > Tools > Analyze Security from Pasted Content**. Paste the SOAP XML message and click **Next**. The next page shows the different security algorithms used in the XML. Click **Finish** to add the security algorithms to the editor.



Note: When you add a secured SOAP XML message in **Message > Source** tab of the test editor, certain security related warnings are displayed in the **Error Message** view. If you are aware of the secured SOAP XML message and do not want to view the warnings, click **Window > Preferences > Generic Service Client > Message Edition** and select the **Analyze pasted SOAP content** check box.

If the WSDL uses WS-Policy, you must configure security as follows:

1. In the test navigator or project explorer, right-click the WSDL file and select **Edit WSDL Security**.

Result

The WSDL security editor is displayed.

2. Click the **Security Algorithms** tab.

Security profiles are described by adding elements to a stack. When a service request is sent or a response is received, each element in the stack is applied to the message in a specified order. If necessary, create one security profile for outgoing requests and one for incoming responses.

3. In the **Security Algorithms** area, click **Add** to create a new algorithm stack, and click **Rename** to change the default name.
4. In the **Algorithm Stack Details** area, click **Add** to add a new algorithm element to the stack. You can add time stamps, username tokens, encryption, or signatures.
5. Edit each element in the stack according to the requirements of the web service.

You can apply encryption and signature stack elements to portions of the web service call or message return document by specifying an Xpath query in **User Xpath part selection**. For example, you can encrypt one XML element with one encryption stack element, and another element with another stack element. You can use the **Web Service Protocol Data** view to help identify the correct Xpath query for this option.

You can check whether the security stack is valid by clicking **Tools > Validate Selected Algorithm**.

6. When all the stack elements are complete, ensure that the execution order is correct. If necessary, use the **Up** and **Down** buttons to change the order of elements in the stack.
7. Repeat steps 4 through 7 to create as many algorithms as are required for security profile.
8. Click the **Algorithms by WSDL Operations** tab. This page enables you to associate a security profile with each request or response operation in the WSDL.
9. In the **WSDL Contents** column, select a service request or response.
10. In the **Algorithm Stack** column, select a security profile from the list. If necessary, click **<<** to open the stack on the Security Algorithms page.

Results

After saving the security profile, the **Web Service Protocol Data** view displays the effect of the security profile on the XML data of the web service.

Related reference

[WSDL security editor reference on page 900](#)

Related information

[Using a security policy on page 259](#)

[Adding WS-Addressing to a security configuration on page 270](#)

[Implementing a custom security algorithm on page 268](#)

Using a security policy

The WS-Policy specification enables web services to use XML to publish their security policies either as part of the Web Services Description Language (WSDL) file (compliant with the WS-PolicyAttachment specification) or as a separate XML document. With the WSDL Security Editor, you can create a security profile that uses a policy that complies with the WS-Policy specification.

Before you begin

Before creating a security configuration, you must have a WSDL file in your workspace.

If the security policy uses digital certificates for encrypting or signing requests or responses, you must have the corresponding keystore files (KS, JKS, JKECS, PKCS12, or PEM) in your workspace.

When you import a WSDL that contains a policy (with WS-PolicyAttachment), a security profile is automatically generated for each operation in the WSDL security editor.

1. In the test navigator or project explorer, right-click the WSDL file, and select **Configure WSDL Security**.

Result

This opens the WSDL security editor.

2. Click the **Security Algorithms** tab.
Security profiles are described by adding elements to a stack. When a service request is sent or a response is received, each element in the stack is applied to the message in the specified order.
3. In the **Security Algorithms** area, click **Add** to create a profile, and click **Rename** to change the default name.
4. In the **Algorithm Stack Details** area, click **Add > WS-Policy** to add the WS-Policy element to the stack.
You can also add time stamps, user-name tokens, encryption, or signatures.
5. If the policy is included in the WSDL file, click **Use policy included in WSDL (WS-PolicyAttachment)**, and edit the WS-Policy settings as required:

Policy

If you are not using the WS-PolicyAttachment specification, specify the XML policy file. Click **Browse** to add a policy file from the workspace or to import a policy file.

Signature configuration

Select this option to specify a keystore for any signature that is specified in the policy. Click **Edit Security** to add a keystore from the workspace or to import a keystore.

Encryption configuration

Select this option to specify a keystore for any encryption that is specified in the policy. Click **Edit Security** to add a keystore from the workspace or to import a keystore.

Decryption configuration

Select this option to specify a keystore for any decryption that is specified in the policy. Click **Edit Security** to add a keystore from the workspace or to import a keystore.

Retrieve token from security token server (WS-Trust and WS-SecureConversation)

Select this option, and click **Configure** to specify a Security Token Server (STS) to use with the policy.

Additional properties

Use this table to specify settings for the advanced properties or specific implementations of the WS-Security specification. Click **Add** to add a property name and to set a value.

6. Check that the security profile is valid by clicking **Tools > Validate Selected Algorithm**.
7. Click the **Algorithms by WSDL Operations** tab.
On this page, you can associate a security profile with each request or response operation in the WSDL.
8. In the **WSDL Contents** column, select a web service request or response operation.
9. In the **Algorithm Stack** column, select a security profile from the list.
If necessary, click << to open the stack on the **Security Algorithms** page.

What to do next

After saving the security profile, the **Web Service Protocol Data** view displays the result of the security profile on the XML data of the web service.

Related information

[Creating security profiles for WSDL files on page 257](#)

[Adding WS-Addressing to a security configuration on page 270](#)

[Implementing a custom security algorithm on page 268](#)

Adding security stacks

To provide better WSDL security, you can make use of many security algorithms in the service test.

About this task

1. From the Test Navigator view or from the Request Library section of Generic Service Client, right-click the WSDL file and select **Edit WSDL Security**.
2. In the **Security Algorithms** area of **Algorithm Stacks** tab, click **Add** to create a profile.
3. In the **Stack Contents** area, click **Add** and add any of the following security algorithms:

Custom Security Algorithm

If you want to use a Java™ class as a custom security algorithm, then use this stack element to apply the custom algorithm to the service.

Java™ Project

If you have not implemented a custom Java™ class, select **Java Project**, type a name for the new project, and click **Generate** to create a new Java™ class with the default structure for custom security implementations.



Note: If you are using IBM® Security AppScan®, this field is not available.

Implementation class

Specify the name of the class that implements the custom security algorithm.

Click **Browse Class** to select an existing Java™ class from the workspace.

Properties

Use this table to send any specific properties and associated values to the custom security algorithm.

WS-Addressing Algorithm

Use this block if your service uses either WS-Addressing 2004/08 or the WS-Addressing 1.0 Core standard.

Namespace

Specify the namespace for either WS-Addressing 2004/08 or WS-Addressing 1.0 Core.

Action if request uses WS-Addressing

Select the action to complete if WS-Addressing is already in the request.

Replace anonymous address in Reply-to with:

Select this option to generate the specified address in the Reply-to header instead of an anonymous address.

Remove WS-Addressing from response

Select this option to strip any WS-Addressing headers from the response.

Encrypted Key

This block defines an encrypted key that can be used in an XML signature or XML encryption block. The encrypted key block must be before a block that uses the encrypted key.

Actor / Role name

Specify the name of the recipient of the algorithm header element, if required.

Must understand

Select whether it is mandatory that the algorithm header is processed by the recipient, if required. The recipient is either the Actor name or the server.

Key name

Specify the name of the encrypted key.

Identifier type

Select the type of key identifier to be used for the key. The following key identifiers are available, as defined in the the Web Service Security (WSS) specification X509 profile and OASIS WSS 1.1 specification:

- ISSUER_SERIAL
- BST_DIRECT_REFERENCE
- X509_KEY_IDENTIFIER
- THUMBPRINT_IDENTIFIER
- SKI_KEY_IDENTIFIER

Key size

Specify the size of the key in bits.

Key encoding algorithm name

Specify the algorithm to be used for encoding the key.

Keystore

Select a keystore or click **Edit Security** to define a new keystore or to manage the existing keystores.

Name

Select a key contained in the specified keystore.

Password

Type the password for the selected key name.

XML Signature

The XML signature security algorithm specifies how the XML document is signed. For details on security algorithms, refer to the web service security specification.

Actor / Role name

Specify the name of the recipient of the algorithm header element, if required.

Must understand

Select whether it is mandatory that the algorithm header is processed by the recipient, if required. The recipient is either the Actor name or the server.

Security token

Select the type of key identifier to be used for the signature. The following key identifiers are available, as defined in the the Web Service Security (WSS) specification X509 profile and OASIS WSS 1.1 specification:

- ISSUER_SERIAL
- BST_DIRECT_REFERENCE
- X509_KEY_IDENTIFIER
- SKI_KEY_IDENTIFIER
- KEY_VALUE
- USER_NAME_TOKEN
- CUSTOM_SYMM_SIGNATURE

In addition, the following identifiers are available when the signature is based on a UsernameToken profile:

- USER_NAME_TOKEN
- CUSTOM_SYMM_SIGNATURE

User XPath part selection

Specify an XPath query that describes parts of the XML document that can be the subjects of the algorithm. By default, the body is the subject. Click the **XPath Helper** button to build the Xpath expression.

Key

Select the key used for the encryption. The details of each key vary.

- **x509 key**: This key specifies the name and password of the x509 key and the keystore where it is located.
- **User name token key**: This specifies a user name and password for the signature.
- **Encrypted key**: This specifies a reference to an encrypted key that was previously defined in the security stack. Click **Insert a new encrypted key** to create a new encrypted key definition block.

Signature algorithm name

Specify the signature method algorithm as described in the XML Signature Syntax and Processing specification.

Canonicalization

Specify the canonicalization method to be used as described in the XML Signature Syntax and Processing specification.

Digest algorithm method

Specify which digest method to be used based on the algorithm method used on the server side.

Inclusive namespaces

Specify whether the canonicalization is exclusive as described in the Exclusive XML Canonicalization specification.

XML Encryption

The XML encryption security algorithm specifies how the XML document is encrypted. For details on security algorithms, refer to the web service security specification.

Actor / Role name

Specify the name of the recipient of the algorithm header element, if required.

Must understand

Select whether it is mandatory that the algorithm header is processed by the recipient, if required. The recipient is either the Actor name or the server.

Identifier type

Select the type of key identifier to be used for the encryption. The following key identifiers are available, as defined in the Web Services Security (WSS) specification X509 profile and the OASIS WSS 1.1 specification:

- ISSUER_SERIAL
- BST_DIRECT_REFERENCE
- X509_KEY_IDENTIFIER
- SKI_KEY_IDENTIFIER
- EMBEDDED_KEYNAME
- THUMBPRINT_IDENTIFIER
- ENCRYPTED_KEY_SHA1_IDENTIFIER

User XPath part selection

This enables you to specify an XPath query that describes parts of the XML document that can be subjects of the algorithm. By default, the body is the subject.

Key

Select the key used for the encryption. The details of each key vary.

- **x509 key**: This specifies the name and password of the x509 key and the keystore where it is located.
- **Raw key**: This specifies the name and the byte value of your SecretKey in hexadecimal.
- **Encrypted key**: This specifies a reference to an encrypted key that was previously defined in the security stack. Click **Insert a new encrypted key** to create a new encrypted key definition block.

Encoding Algorithm Name

Specify the encryption method to be used as defined in the XML Encryption Syntax and Processing specification.

Key Encoding Algorithm

Specify the standard algorithm for encoding the key as defined in the XML Encryption Syntax and Processing specification.

User name token

The user name token security algorithm adds a user name token to the XML document in the message. For details on security algorithms, refer to the web service security specification.

Actor / Role name

Specify the name of the recipient of the algorithm header element, if required.

Must understand

Select whether it is mandatory that the algorithm header is processed by the recipient, if required. The recipient is either the Actor name or the server.

Name

Type the name of the user.

Password

Type the password of the user.

Password type

Specify the password type for the security algorithm as defined in the Web Services Security UsernameToken profile.

Use nonce

Select this check box to add the Nonce element to the User Name Token XML code. In most cases, the Nonce ID is required.

Use created

Select this check box to add current timestamp to the Created XML element in the User Name Token XML.

Time Stamp

The time stamp security algorithm adds time stamp information to the XML document in the response. For details on security algorithms, refer to the web service security specification.

Actor / Role name

Specify the name of the recipient of the algorithm header element, if required.

Must understand

Select whether it is mandatory that the algorithm header is processed by the recipient, if required. The recipient is either the Actor name or the server.

Expiration delay

Specify the delay after which the time stamp expires.

Millisecond precision

Select this option to produce a time stamp that uses millisecond precision instead of the default (1/100th second).

SAML Assertion Block

To use the self-signed SAML assertion security algorithm, add the SAML Assertion stack to the request or WSDL files.

User XPath part selection

Specify an XPath query that describes parts of the XML document that can be the subjects of the algorithm. By default, the body is the subject. Click the **XPath Helper** button to build the Xpath expression.

Key

Select the key used for the encryption. The details of each key vary.

- **x509 key**: This key specifies the name and password of the x509 key and the keystore where it is located.
- **User name token key**: This specifies a user name and password for the signature.
- **Encrypted key**: This specifies a reference to an encrypted key that was previously defined in the security stack. Click **Insert a new encrypted key** to create a new encrypted key definition block.

Signature algorithm name

Specify the signature method algorithm as described in the XML Signature Syntax and Processing specification.

Canonicalization

Specify the canonicalization method to be used as described in the XML Signature Syntax and Processing specification.

Digest algorithm method

Specify which digest method to be used based on the algorithm method used on the server side.

Inclusive namespaces

Specify whether the canonicalization is exclusive as described in the Exclusive XML Canonicalization specification.

Signed Assertion

Select this check box to self-sign the SAML Assertion.

Issuer

Specify the description of the issuer of the SAML Assertion or protocol message.

Subject

Specify the principal that is the subject of all of the statements in the assertion. It might contain an identifier or a series of one or more subject confirmations.

Subject Qualifier

Specify the Name Qualifier of the Subject

Subject Format

Specify the format used for the Subject.

Subject Locality DNS

Specify the DNS domain name for the system from which the assertion subject was authenticated.

Subject Locality IP

Specify the IP address for the system from which the assertion subject was authenticated.

Statement Type

Specify the authentication method to use for the assertion.

Authentication: The assertion subject was authenticated

Attribute: The specified subject is associated with the supplied attributes.

Authorization decision: Permission to allow a subject to access the specified resource.

Requested Resource

When **Authorization decision** option is used, specify the resource for which you need access.

Action

Specify what action to take to access the resource.

Confirmation number

Confirmation methods define the mechanism by which an entity provides evidence (proof) of the relationship between the subject and the claims of the SAML assertions.

Sender vouches: Select this option when a server needs to share the client identity with SOAP messages on behalf of the client. This method is similar to identity assertion, but it has the added flexibility of using SAML assertions to share not only the client identity, but also client attributes.

Holder of key: Select this option when the proof of the relationship between the subject and claims is established by signing part of the SOAP message with the key specified in the SAML assertion. Because there is key material associated with a holder-of-key token, this token can be used to provide a message-level protection (signing and encryption) of the SOAP message.

Bearers: Select this option when the proof of the relationship between the subject and claims is implicit. No specific steps are taken to establish the relationship. Because there is no key material associated with a bearer token, protection of the SOAP message, if required, must be performed using a transport-level mechanism or another security token, such as an X.509 or Kerberos token, for message level protection.

Version

Specify the SAML version used.

4. **Optional:** To verify simple SAML code, use the **Analyze Security from Pasted Content** option. For more information about that option, see [Creating security for WSDL profiles on page 257](#).

Implementing a custom security algorithm

You can define your own security algorithms for SOAP security profiles by implementing custom security Java™ interfaces that can be used in the WSDL security editor. With custom security algorithms, you can implement proprietary security algorithms that transform the XML before sending and after receiving message content.

Before you begin

The custom security interface and the JAR file that contains it are provided with the product in the `customsecuritydefinition` folder of the `com.ibm.rational.ttt.common.models.core` plugin. You need these interfaces to create your own algorithms. If you are using IBM® Rational® Performance Tester or IBM® Rational® Service Tester for SOA Quality, see [Extending test execution with custom code on page 402](#) for more information about extending test capabilities with Java™ code.

1. In the test navigator or project explorer, create a new Java™ class in your web service test project folder.
2. Implement a security algorithm in Java™ using the following interface:

```
/**
 * *****
 * IBM Confidential
 *
 * (c) Copyright IBM Corporation. 2008. All Rights Reserved.
 *
 * The source code for this program is not published or otherwise
 * divested of its trade secrets, irrespective of what has been
 * deposited with the U.S. Copyright Office.
 * *****
 */

package com.ibm.rational.test.lt.models.wscore.datamodel.security.xmlsec;

import java.util.Properties;
import org.w3c.dom.Document;

public interface ICustomSecurityAlgorithm {
```

```

/**
 * The following methods can be used in both case:
 * Execution in the workbench and execution of the test.
 */

/**
 * Called to process de Document that is sent over a transport.
 * @param subject
 */
void process(Document subject);
/**
 * Called to un process a document that is received from a server.
 * @param subject
 */
void unProcess(Document subject);

/**
 * Properties defined in the UI of the CustomSecurityAlgorithm.
 * @param map
 */
void setProperties(Properties map);

/**
 * The following methods can only be used in terms of cast to test service interface,
 * or in terms of access to the previous XML information, when the jar containing
 * the custom security algorithm is deployed in the performance test project. In
 * this case you cannot use the algorithm directly from the workbench.
 */

/**
 * This object corresponds to the ITestExecutionService object.
 * This applies only to an algorithm that must link to the execution of the test.
 * If you plan to use this object you will need to deploy the jar containing the
 * implementation into your performance test project and not directly into the JRE.
 *
 * In case of a need of the previous xml document received from the execution you can
 * obtain the value using:
 * IDataArea area =
 * ((ITestExecutionService)executionObject).findDataArea(IDataArea.VIRTUALUSER);
 * String previousXML = (String) area.get("PREVIOUS_XML"); //$NON-NLS-1$
 *
 */
void setExecutionContext(Object executionObject);

```

The `process` method modifies the XML before it is sent to the server.

The `unprocess` method modifies the XML after it is received from the server.

The `setProperties` method retrieves any properties that are defined in the security editor for this custom security interface.

The `setExecutionContext` method is called during test with the object `ITestExecutionServices` that corresponds to the message using this custom security interface.

3. The custom security interface can be used either in the **WSDL security editor** for web services or in XML call elements in the **Local XML security** tab.

Choose from:

- If you are testing a WSDL-based web service, right-click the WSDL file in the test navigator or project explorer to open the WSDL security editor, select the **Security Algorithms** page; then, under **Details of selected security algorithm stack**, click **Add > Custom Security Algorithm**.
 - If you are testing an XML call, open the XML call element in the test editor, select the **Local XML Security** tab, and then, click **Add > Custom Security Algorithm**
4. In custom security, click **Browse Class** to select the class name of the custom security algorithm, for example: `ICustomSecurityAlgorithm`.
 5. Type an **Algorithm name** for the custom security algorithm.
 6. In the properties list, use **Add**, **Remove**, or **Edit** to specify any properties that are used by the `setPropertyies` method in your custom security algorithm.

What to do next

After saving the security configuration or the call element, the **Web Service Protocol Data** view displays the effect of the security algorithm on the XML data of the web service.

Related reference

[WSDL security editor reference on page 900](#)


Adding WS-Addressing to a security configuration

The WS-Addressing specification provides transport-neutral mechanisms that enable SOAP-based web services to communicate addressing information. You can use WSDL security algorithms to add WS-Addressing to your service tests.

Before you begin

Before adding WS-Addressing to a security configuration, you must have a service test with requests and responses that are related to a valid WSDL.

To add WS-Addressing to a WSDL security algorithm:

1. Open the test, select a service request, and in the **Raw Transaction Data** view, select **Enable the display of the XML document after the security processing**.
2. On the **Request Stack** page, click **Edit WSDL Security** .



Tip: If you need to edit separate security or processing algorithms for incoming responses, click **Show Response Stack** to add a **Response Stack** page to the editor.

Result

The **WSDL security editor** opens.

3. Select the **Algorithm Stacks** page of the WSDL security editor, and in the **Security Algorithm** list, select or create a security algorithm.
4. In the **Stack Contents** list, click **Add > WS-Addressing** and specify the settings that are implemented by the service.

WS-Addressing Algorithm

Use this block if your service uses either WS-Addressing 2004/08 or the WS-Addressing 1.0 Core standard.

Namespace

Specify the namespace for either WS-Addressing 2004/08 or WS-Addressing 1.0 Core.

Action if request uses WS-Addressing

Select the action to complete if WS-Addressing is already in the request.

Replace anonymous address in Reply-to with:

Select this option to generate the specified address in the Reply-to header instead of an anonymous address.

Remove WS-Addressing from response

Select this option to strip any WS-Addressing headers from the response.

5. Save the WSDL security algorithm, and select the test editor.

Result

The WS-Addressing namespace and header XML content is displayed in the **Raw Transaction Data** view.

Related reference

[WSDL security editor reference on page 900](#)

Related information

[Creating security profiles for WSDL files on page 257](#)

[Implementing a custom security algorithm on page 268](#)

Testing asynchronous services

Use the asynchronous callback services for inter-object communications in a service test.

Asynchronous service testing overview

Asynchronous services use a callback interaction pattern for inter-object communications. Asynchronous services can be used, for example, in publish-subscribe systems that are provided by message-oriented middleware vendors or in system and device management domains.

WS-Notification services

Asynchronous services are standardized in the *WS-Notification* specifications:

- *WS-BaseNotification* defines the web services interfaces for *NotificationProducers* and *NotificationConsumers*. This specification includes standard message exchanges that are implemented by service providers that want to act in these roles, along with the associated operational requirements.
- *WS-BrokeredNotification* defines the web services interface for a *NotificationBroker*. A *NotificationBroker* is an intermediary which, among other things, enables entities that are not service providers themselves to publish messages. It includes standard message exchanges that are implemented by *NotificationBroker* service providers, along with the associated operational requirements of service providers and requestors that participate in brokered notifications.
- *WS-Topics* defines a mechanism to organize and categorize items of interest for subscription known as *topics*. These are used in conjunction with the notification mechanisms defined in *WS-BaseNotification* and *WS-BrokeredNotification*.

You can test web services and XML services that implement the *WS-Notification* specification by creating an asynchronous request inside a test. The asynchronous request contains the interfaces for the corresponding *WS-Notification* specification, along with a callback structure.

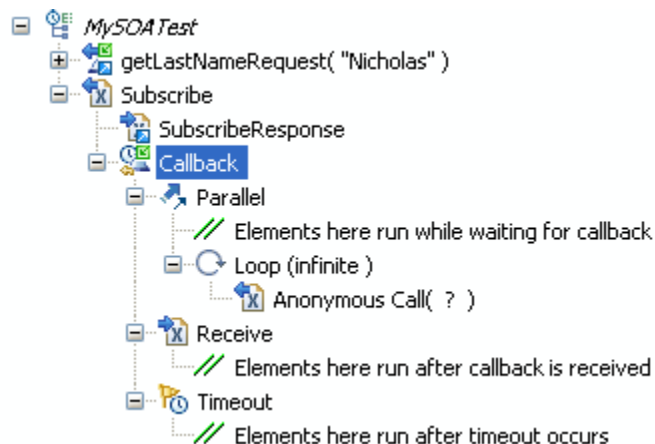
Proprietary asynchronous services

You can test proprietary asynchronous services that do not implement *WS-Notification* specifications. To test these services, you manually create a service request that contains the interfaces for the service, and then, you can add the asynchronous callback structure to the request.

The XML data of the asynchronous request must contain an endpoint that specifies the URL of the callback receiver. During the test, this endpoint is used to redirect the callback to the tester instead of the real receiver.

Callback structure

To test asynchronous services, you must create an asynchronous request structure in your test as shown in the following diagram:



A web service request or a plain XML request provides the subscription action and contains a callback element, which describes the behavior of the test in three states:

- *Parallel* contains test elements that are run after the subscription request and while waiting for the notification response.
- *Receive* contains test elements that are run when the notification response has been received from the service.
- *Timeout* contains test elements that are run if the notification response is not received after a delay that is specified in the callback element.

When everything contained in the parallel, receive, and timeout elements have finished running, the run continues with the next element in the test after the asynchronous request.

The method for generating the asynchronous callback structure in the test depends on whether the asynchronous service uses the WS-Notification specification:

- WS-Notification services: Create the asynchronous request in the test.
- Proprietary services: Manually create a web service request or XML request in the test, and then add the asynchronous callback structure to the request.

Creating an asynchronous request structure

You can create an asynchronous request based on the *WS-Notification* specification, which contains an callback structure.

1. In the test editor, select the test, and click **Add**, and then click **Specification-based Structure**.

Result

The **New Web Service Test** wizard opens.

2. On the **Web Services Specification Selection** page, Select **WS-Notification**, and click **Next**.
3. On the **WS-Notification Details** page, if the service has a Web Services Description Language (WSDL) file, click **Add** to associate it with the call.
4. Specify the **Subscription identifier**.
You can select default identifiers for Websphere Application Server or Apache Muse; or if your service does not use a standard identifier, you can select **Custom**, and type the **Name** and **Namespace** of the identifier.
5. In the **Topic** area, replace the default **Name** and **Namespace** values with those of topic of your service.
6. Specify the **Subscription duration**.
Because this is a test environment, the subscription expires after the specified delay to save server resources.
7. If this is a WS-BrokeredNotification service, which implements a notify call when the subscription is received, you can select **Add notify call**, and type the message to be sent.
8. Click **Next**.
9. On the **Configure Protocol** page, select a **Protocol configuration**, and specify the options of the configuration.
Select **Generate SOAP 1.2 envelope** if you are testing a SOAP 1.2 web service.
10. Click **Finish**.

Result

This action generates in the test editor a web service call or an XML request with a callback structure that contains a parallel, a receive, and a timeout element.

What to do next

In the callback structure, add test elements to the parallel, receive, and timeout elements to specify the behavior of the test:

- *Parallel* contains test elements that are run after the asynchronous call has been sent.
- *Receive* specifies the message return of the callback and contains test elements that are run after the callback is received.
- *Timeout* contains test elements that are run if the callback is not received after a specified delay.

Adding an asynchronous callback to a service request

To test a proprietary asynchronous service that does not implement the *WS-Notification* specification, you can add an asynchronous callback to a service request or XML request.

Before you begin

Manually create a web service call or XML call that invokes the asynchronous service. The call must contain an endpoint that specifies the URL of the callback receiver. This endpoint is used to redirect the callback to the tester.

If the service implements the WS-Notification specification, create the asynchronous call structure with the **Create New WS-Notification Request and Callback** wizard instead. See [Creating an asynchronous request structure on page 273](#).

1. In the test editor, select a web service or XML request, click **Add**, and then click **Asynchronous Callback**.

Result

The **Create New Asynchronous Callback** wizard opens.

2. On the **Select Callback Endpoint** page, select the XML element of the request where the endpoint URL of the callback is located.
3. If you have a web Services Description Language (WSDL) file for the web service, click **Next**. Otherwise, skip to step 5.
4. On the **Bind Message to WSDL Port** page, select a port from the WSDL file. If the WSDL file for the service is not listed, click **Add** to add a WSDL file from the workspace or to import a WSDL file.
5. Click **Finish**.

Result

This generates a callback structure that contains a parallel, a receive, and a timeout element, in the test editor.

What to do next

In the callback structure, you can add test elements to the parallel, receive, and timeout elements to specify the behavior of the test:

- *Parallel* contains test elements that are run after the asynchronous request has been sent.
- *Receive* specifies the message return of the callback and contains test elements that are run after the callback is received.
- *Timeout* contains test elements that are run if the callback is not received after a specified delay.

Creating a reliable messaging call structure

You can create a test structure dedicated to testing service calls based on the *WS-ReliableMessaging* specification.

Before you begin

The WS-ReliableMessaging specification provides for a series of SOAP messages to be delivered reliably between distributed applications in the presence of software component, system, or network failures. In the context of a service test, a reliable messaging call structure consists of a series of calls that conform to the specification. The structure can be created either as a sequential list of unique service calls or a loop that contains a call element and uses a dataset to identify the unique calls.

1. In the test editor, select the test, and click **Add**, and then click **Specification-based Structure**.

Result

The **New Web Service Test** wizard opens.
2. On the **Web Service Specification Selection** page, Select **WS-ReliableMessaging**, and click **Next**.
3. Select one or several Web Services Description Language (WSDL) files in your workspace for the web service that you want to test, and click **Next**.

If necessary, you can import a WSDL file into the workspace with the **Import** push button.
4. On the **Configure Protocol** page, select an existing HTTP transport configuration, or click **New** to create a new configuration.
 - a. Specify the **URL** of the service, the HTTP **Method**, and **Version**.
 - b. In the **Header** table, click **Add** to specify any specific headers that need to be added to the call.
 - c. In the **Cookies** table, click **Add** to specify any specific cookies that need to be used by the call.
 - d. Click **Next**.
5. On the **Sequence Options** page, specify how the sequence structure will be created in the test.
 - a. In **Message count**, specify the number of calls in the list or the number loop iterations.
 - b. Select **Create service call list** to generate a list of calls with the number of messages or **Create loop with dataset** to generate a loop with a dataset.

The dataset defines the call number for each call in the loop.
 - c. Select **Shuffle sequence** if you want the call numbers to be generated in a random order.
6. Click **Finish**.

Result

This action generates a reliable messaging service call structure in the test.

Searching within tests

Search request data or response content by right-clicking in the data or content and selecting **Find**. To search for specific element types and to display the results in a table, click **Select**. For a still more powerful search and replace, use the **Test Search** function.

About this task

You can use a number of different methods to search within a test.

- Use the **Find** option to search within the **Test Details** area and, optionally, to replace text.
- Use the **Select** button to search within the **Test Contents** area and display a table of like test elements.
- Use the powerful **Test Search** function to search within the **Test Contents** and the **Test Details** areas. For example, you can search for a type of verification point and also declare whether the result should include enabled verification points, disabled verification points, or verification points in both states. The specific data that you can search for and the search options are protocol-dependent.

Locating specific types of test elements

A test script can include multiple requests and responses with many test elements and attributes. To locate elements of a specific type in the **Test Contents** area, click **Select**. The results are displayed in a table, and you can sort the table columns. This option is also useful for viewing attributes of test elements that are the same type.

1. In the Test Navigator, browse to the test and double-click it.

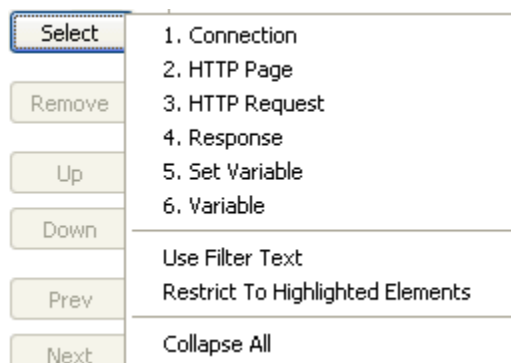
Result

The test opens.

2. To restrict the search to specific instances of elements, select them in the test. For example, you might want to search for text in specific responses, rather than in all responses.
3. Click the test editor tree to give it focus, and then click **Select**.

Result

A list of element types opens. This list is generated dynamically based on the contents of the test. For example, if a test does not contain verification points, they are not listed.



4. To include only the elements that you selected, select **Restrict To Highlighted Elements**.
5. To include only the elements that contain specific text from the **Test Contents** area, select **Use Filter Text**.
6. Select the type of test element to view from the list.

- If you selected **Use Filter Text**, enter the filter text in the prompt, and then click **OK**.

To use regular expressions in the filter text, click the **Search Options** icon to the right of the prompt, and then select **Regular expression**. By default, if **Regular expression** is not selected, the asterisk (*) and question mark (?) are interpreted as wildcard characters. To search for a URL that contains an asterisk or question mark, type a backslash (\) before the asterisk or question mark.

Results



The **Test Element Details** area displays the results under the heading **Multiple Items**.

Example

The following example shows results for an HTTP request. Other protocols might display less detailed information. Double-click a table row to locate the element within the test.

Test Element Details

Multiple Items

	URL	Delay	 Response	Size
GET	www.ibm.com/	0 ms.	302 - Found	206
GET	www.ibm.com/us/	0 ms.	200 - OK	31,458
GET	www.ibm.com/common/v16/hp/js/h...	0 ms.	200 - OK	5,742
GET	www.ibm.com/common/v16/css/all.css	0 ms.	200 - OK	138
GET	www.ibm.com/common/v16/css/scr...	0 ms.	200 - OK	18,188
GET	www.ibm.com/common/v16/css/scr...	156 ms.	200 - OK	99,371
GET	www.ibm.com/common/v16/css/us/...	0 ms.	200 - OK	15,847

Searching and replacing text in tests

With the **Test Search** function, you can search for text in a test or search within specific test elements and optionally replace the found text.

- In the **Test Navigator**, browse to the test and double-click it.

Result

The test opens.

- Right-click the test name, and then select **Test Search**.
- In **Search for text**, type the text to locate.

You can leave this field blank, depending on your search strategy. For example, if you know that a string occurs in elements or element instances that you are not interested in, by using the options described in steps 4, 6, and 8, you can locate the elements or element instances of interest before entering the search text into this field.

- If you have selected pages or requests within the test, click **More Options**, and then select **Restrict search to elements highlighted in Test Contents**.

This restricts the search to the selected pages and requests.

- To perform a case-sensitive search, select **Case sensitive**. To search with regular expressions, select **Regular expression**.

In regular expression mode, press Ctrl and type a space in **Search for text** for content assistance. Content assistance lists the regular expression patterns and the content that they match.

6. To highlight found elements in the **Test Contents** area, click **More Options**, and then select **Highlight found elements in Test Contents**.

You can use this option with the option that is described in step 4 to designate the element instances of interest before specifying the text of interest.

7. To have the search include children of the selected element, click **More Options**, and then select **Recursive**. This option is selected by default. If **Recursive** is cleared, then only the selected element is searched.
8. To have the search locate both encoded and decoded versions of the specified text, click **More Options**, and then select **Match encoded and decoded values**.

This option is selected by default. The type of encoding that the search supports varies depending on the protocol.

Example

For example, when searching in HTTP data, `abc%123` and `abc%25123` match.

9. In the **Elements to search** list, select all test elements to search.

Selecting the check box in step 4 restricts the elements that you can select in this step to the instances that are selected in the **Test Contents** area. For example, if you select **HTTP Pages** here and only one page is selected in the **Test Contents** area, only that page is found. If the check box in step 4 is cleared, every test page is found.

10. **Optional:** Click selected elements to define how to search them.

A new area opens, where you can define how to search a selected element.

To locate items, continue to the next step. To replace found strings, click **Replace**, and go to step 12.

11. Click **Search**.

Result

The results of your search are displayed in two views

- The Search view, which lists the objects that contain matches
- The Test Search Match Preview view, which displays the matches that were found

12. In the Search view, complete any of these search actions:

- To preview a found string in the Test Search Match Preview, click the object.
- To open your test at the location where an instance is found, double-click the object.
- To perform a different search action (such as proceed to the next match or previous match, replace), right-click the object, and select your choice.

13. If you clicked **Replace** in step 9, the **Replace** window opens. In the **With** field, type the replacement text.

14. Select the replacement action by clicking the appropriate push button.

Result

If you are making selective replacements, found instances are displayed in the same order as in the **Test Search Match Preview** view. Click **Replace** or **Skip** until all found instances have been displayed.

Exporting a test

To share the test scripts with manual testers or reviewers who do not have the workbench, export the test scripts to text files. You can export one file at a time.

1. In the Test Navigator view, double-click a test.
2. In the test editor, right-click the root node of the test and click **Export Contents to File**.
3. Select a project and specify the name of the file to export to.
4. To add a separator between two steps or lines, select the **Add line separators after each step** check box and click **Finish**.

Results

The text file opens on another tab in the workbench and is saved in the directory it is exported to.

Copying test assets with dependencies

You can export test assets, and then import them into another project or workspace without losing any dependencies. Test assets include projects, schedules, and tests. You can export and import test assets to collaborate with other testers.

Before you begin

If you plan to export assets with dependencies, make sure that you have migrated the test assets to the current version of the product before you start to export.

About this task

When you copy a test with dependencies, any datasets or custom code referred to by the test are also copied. When you copy a schedule with dependencies, any locations or tests referred to by the schedule are also copied. When you copy results, any schedules or tests referred to by the results are copied.

1. In the **Test Navigator** view, right-click the test assets to export, and then click **Export**.
You can export projects, schedules, tests, and test results with dependencies.
2. In the **Export** window, expand the **Test** folder, and then click **Test Assets with Dependencies**.
You can export test assets with dependencies if the test assets were created in the current version of the product. You cannot export test assets with dependencies if the test assets were created in a previous version of the product and the assets have not been migrated to the current version of the product.
3. Click **Next**.
4. Specify the path and name of the archive file into which you want to export the selected test assets.
5. Click **Finish**.
The assets are exported to the archive file. You are prompted if the total size of the test assets is larger than 4 GB, or if any individual test asset file is larger than 4 GB. To copy test asset files that are larger than 4 GB, copy the files manually.
6. If the target workspace is on a different computer, transfer the archive file to a location that is accessible to that computer.
7. In the **Test Navigator** view, select the test project into which you want to import the test assets.

The target project must have the same name as the source project. Optionally, you can import test assets with dependencies into a workspace where no projects exist. If you import test assets with dependencies into a workspace where no projects exist, the Import wizard creates projects based on information from the archive file. To import test assets into a project with a different name, you must first import the test assets into a project with the same name, and then manually move the assets into the project with the different name.

8. Click **File > Import**.

9. In the **Import** window, expand the **Test** folder, and then click **Test Assets with Dependencies**.

10. Click **Next**.

11. In the **Import with dependencies** window, click **Browse**, and then select the archive file.

The test assets are displayed in the **File contents** list.

12. Click **Finish** to import the test assets with dependencies from the archive file into the target project.

If a file that you are attempting to import already exists in the target workspace, you will be prompted to choose whether to overwrite the file. You can also choose to overwrite all files that already exist in the target workspace, or not to overwrite any files that already exist in the workspace. If you choose to overwrite all files that already exist in the target workspace, you will be prompted again if the import process encounters a `.classpath` or `.project` file in the source archive file.

Copying projects

You can export a test project from a workspace and import it into another workspace.

About this task

If you export test assets to an archive file and then import them to another project, ensure that both project names are the same. Otherwise, you might not be able to locate your imported test assets.



Note: You can also export the test project with all the dependent assets in to an archive file. See [Copying test assets with dependencies on page 279](#) for the instructions.

1. Start Rational® Performance Tester, and select the source workspace.

2. Export the test project to an archive file.

For instructions, see [Exporting resources to an Archive file](#). Datasets can be located either in the same project as the tests that use them or in different projects. Be sure to export all the datasets that the exported tests require.

3. If the target workspace is on a different computer, transfer the archive file to a location that is accessible to that computer.

4. Start Rational® Performance Tester, and select the target workspace.

5. Click **File > Import**. Expand the **General** folder, and click the **Existing Projects into Workspace** icon; then click **Next**.

6. Click **Select archive file**, and then click **Browse** to select the archive file. Click **Finish** to import the source project from the archive file into the target workspace.

7. **Optional:** If the imported project contains custom code or tests that have been run, you might need to change the Java™ build path.

The following examples are cases that might require a change to the Java™ build path:

- The Java™ build path was manually changed in the project from which it was exported. In this case, the same changes need to be made in the imported project. While importing, you are asked whether to overwrite the class path file, which stores the Java™ build path for project. Answering **Yes** reduces the likelihood that the build path will need to be changed.
- The project was imported onto a different computer with a different Java™ installation configuration. In this case, missing libraries must be removed from the build path.
- The project was imported into a workspace on a different disk drive. When you are asked whether to replace the class path file, answering **No** reduces the likelihood that the build path will need to be changed.

For instructions on changing the build path, see the [Java™ Build Path page](#).

What to do next

If you encounter errors after importing a test project or when using an existing workspace with a new version of the product, you might need to delete .java files from the `src` folder in the workspace:

1. Click **Window > Open Perspective > Resource** to open the Resource perspective.
2. In the **Navigator** window, expand the test project folder, and locate the `src` folder.
3. Delete all .java files in the `src` folder, except for those that contain custom code.
4. Return to your test perspective: Click **Window > Open Perspective**, and select **Performance Test** (or **Service Test**, if you are using Rational® Service Tester).

Disabling portions of a test

When you disable portions of a test, you can still see the disabled portion, but it is not executed during a run.

To disable an element:

1. In the Test Navigator, browse to the test, and double-click it.

Result

The test opens.

2. Right-click the element that you want to disable, and select **Disable**.

The element and the dependent child elements, which are disabled automatically, are shaded and preceded by two forward slashes (//) to remind you that they are disabled.



Note: To change the color or symbol that represents disabled elements, click **Windows > Preferences > Test > Test Editor**, and then click the **Colors and Fonts** tab.

Result

Although a disabled test element does not run, you can still work with it. For example, you can insert a test into a disabled user group for later use.

- To enable a disabled element, right-click it, and select **Enable**. Select **Enable All** to enable all disabled elements.

Exemple

Disabling an element affects other elements in the following ways:

Disabled element	Result
User group (percentage)	The percentages in the remaining user groups are recalculated. When you enable the user group again, remember to return all of the affected user groups to their original percentage.
Request or step that contains a data correlation reference	Substitution in the remaining actions that depend on this request does not work.
Request or step that contains a data correlation substituter	Substitution does not occur because the entire action is omitted. The substituter that uses the disabled data source is also disabled. To re-enable the substituter, select an enabled data source for substitution.
HTTP request that contains a server connection	No effect. The connection is automatically created in the next request.
Portion of custom code	Custom code with disabled arguments is flagged. If the disabling causes an unexpected number of arguments passed to custom code elements, you receive an error at runtime. To fix this, modify the custom code to check the number of arguments.
IF <i>data_source</i> construct	An IF construct is marked as invalid if it contains a disabled data source.
Test element and child are disabled	<p>If you disable a child element and then disable its parent (for example, a request and then a page), the disabled child element will have two prefixes: one created manually and one inherited. In the following example, the first request has inherited the disabled state. The second request has been manually disabled and has also inherited the disabled state:</p> <pre>//disabled page //request ///disabled request</pre>

Disabled element	Result
	Do one of the following to re-enable the second request: <ul style="list-style-type: none"> • Re-enable the request, and then re-enable the page. • Right-click the page and select Enable All.
A data source or a range of text that will be replaced	The Data table displays this text in gray.

Running test elements in random order

You can record multiple user scenarios in a test and then run each scenario in a random order. To do this, you put each scenario under a random selector and then select the proportion of time that the scenario should be run.

About this task

For example, you can record a test that includes logging on to a system, browsing through items in the system, buying various items, and then totaling the order. In this case, you could run the logging in and the totaling scenarios once, but put the browsing and buying scenarios under a random selector.

1. In the Test Navigator, browse to the test and double-click it.

Result

The test opens.

2. Click the test element that will be controlled by the random selector, and then click **Insert > Random Selector**. Use Shift+Click to select multiple elements.
3. You are asked whether you want to move the selected elements into a new random selector. Click **Yes**. Click **No** to insert an empty random selector into the test.



Note: To set whether or not elements are moved automatically, or whether you are prompted, click **Window > Preferences > Test > Test Editor**, and click the **General** tab.

4. Set the weight of the random selector. The weight determines the statistical probability that a specific element will be selected.
 - a. If you have added a number of test elements, the **Create weighted blocks** window is displayed. You can select adjacent elements and group them. Each element—whether in a group or by itself—must be weighted.
 - b. If you have added only one test element, the weighted block is displayed in the **Test Element Details** area with a default of 1.

Example

When a selector contains many different weights, you can mathematically determine the likelihood that a block will be executed. To do this, add the weights together and divide the weight for each block by that total.

For example, assume a selector contains six blocks set to the following weight:

- two blocks set to a weight of 1
- one block set to a weight of 2
- two blocks set to a weight of 5
- one block set to a weight of 9

The total of the weights is: $1 + 1 + 2 + 5 + 5 + 9 = 23$. Therefore, the statistical likelihood of selection is:

Weight of block	Likelihood of block being selected
1 (two blocks)	$1/23 = 0.0435$, or about 4.35% (for each block)
2	$2/23 = 0.0870$, or about 8.70%
5 (two blocks)	$5/23 = 0.2174$, or about 21.74% (for each block)
9	$9/23 = 0.3913$, or about 39.13%

Note that a higher weight increases the likelihood, but does not guarantee, that a block will be executed. Some variation might occur. For example, if you run a test 23 times, you cannot predict that the first and second blocks will execute exactly once, the third block exactly twice, the fourth and fifth blocks exactly five times, and the sixth block exactly nine times. However, the more times that the blocks are executed, the more accurate this prediction is.

Renaming test assets

As your test assets increase and become more complex, you might want to rename them. Use the Eclipse **Rename** function or save the assets under a different name.

Use either of the following steps to rename a test asset:

1. When you use the Eclipse **Rename** function, the new name is visible in the Test Navigator, but the underlying file system name is not changed. To use the Eclipse **Rename** function:
 - a. In the Test Navigator, right-click the test asset, and then select **Rename**
 - b. Type the new name, and then click **Enter**
Be sure to click **Enter**, or the file will not be renamed.
2. When you rename a test asset by saving it under another name, the underlying file system name is changed, but you must perform manual cleanup. To save a test asset under another name:
 - a. In the Test Navigator, browse to the test and double-click it. The test asset opens.
 - b. Click **File > Save As**, and save the asset under a different name.
 - c. Delete the original asset.

Example

The following table summarizes how renaming an asset affects the other assets in your workspace.

Renamed asset	Effect on other assets
Project	Do not rename a project. Renaming a project might result in lost or corrupted project assets.
Schedule	Renaming a schedule has no affect on other assets, but note that results cannot be renamed.
Test	<p>When you use Rename, schedules that contain the old test name will still run correctly. To avoid confusion, manually update the schedule to use the new test name.</p> <p>After you use Save As, manually update each schedule that uses the renamed test.</p>
Custom code	<p>If you rename the custom code class (.java file), then the reference to the class in the custom code action of the test will not work. Typically rename the custom code class in the Resource perspective or the Java™ perspective.</p> <p>If you change the name of the custom code class in the test editor that implements the custom code action, the modification does not change the corresponding .java file; instead the modification causes the custom code action to refer to a different (and possibly new) custom code class.</p>
Dataset	<p>When you use Rename and open a test that contains the dataset, you are prompted to save the changes (in this case, the renamed dataset that the test now uses).</p> <p>After you use Save As, manually update each test that uses the dataset.</p>
Location	<p>When you use Rename, locations (agent computers) are automatically updated in the schedules that use them.</p> <p>When you use Save As, manually update each schedule that uses the test.</p>
Results	You cannot rename results.
Weighted block	Renaming a weighted block has no affect on other assets. To rename a weighted block, click the block in the test, and type the new name in the Name field.

Deleting test assets

As your test assets grow and become more complicated, you might want to delete the assets that you no longer use.

In the Test Navigator, right-click the test asset, and then select **Delete**.

Result

The following table summarizes how deleting an asset affects the other assets in your workspace.



Note: If you are deleting a test asset, you can choose to delete it from other test assets that refer it and you can choose to delete other test assets that are referenced only by the test asset that you are deleting.

For example, if you delete a dataset, the **Remove references to test asset name in other test assets** option lets you delete the dataset from all the test assets that uses it.

If you delete a test, the **Delete files that are referenced only by test asset name** option lets you delete all the test assets such as recession and dataset that are referenced only by the test that you are deleting. If the dataset is used by another test too, it will not be deleted. The **Preview** button lets you see the assets that are referenced by the test.

Deleted asset	Effect on other assets
Project	You are prompted whether to delete the project contents. If you click Yes , the contents are physically deleted. If you click No , you will not see the contents in the Test Navigator, but the project remains in your workspace, which is, by default, <code>C:\Documents and Settings\user-name\IBM__SDP_FULL_SHORTNAME__n.n/workspace</code>).
Schedule	Deleting a schedule has no effect on other assets.
Test	If you delete the test in the Test Navigator, the test is physically deleted. If you open a schedule and delete a test, the test is deleted from the schedule, but the test remains available as a test asset.
Custom code	If you delete the custom code class (.java file), then the reference to the class in the custom code action of the test will not work. Typically you delete the custom code class from the Resource perspective or the Java™ perspective.

Deleted asset	Effect on other assets
	If you delete the name of the custom code class that implements the custom code action, the deletion does not change the corresponding .java file.
Dataset	<p>If you delete a dataset in the Test Navigator, the dataset is physically deleted. When you open a test that uses the dataset, you are prompted to take one of these actions:</p> <ul style="list-style-type: none"> • Locate the dataset • Remove the dataset reference from the test • Leave the invalid reference in <p>You must correct or delete the reference to run the test successfully.</p> <p>If you open a test and delete the dataset from the Common Options tab, only the reference to the dataset is deleted.</p>
Location	You are not asked to confirm the deletion, nor are you warned if a user group uses the location. The user group is marked with a red x when you open the schedule that contains it.
Results	You are asked to confirm the deletion, and the results are physically deleted.

Debugging custom code for tests and compound tests

If you have custom code added to a test or a compound test, you can debug the custom code for any errors by clicking the **Debug** button.

Before you begin

This procedure can only be done when custom code is part of a test or compound test. If there are multiple custom code classes added to a test or compound test, the debug action debugs all the custom code classes.

You can use the debug option only in full Eclipse mode of the product. You cannot debug in the streamline mode.

1. Open a test or a compound test from the Test Navigator view.
2. Click the **Debug** button.

Result

The Debug view opens. If there are any breakpoints in the custom code, the test run pauses at the breakpoint. Press F8 to resume the test run.

Providing tests with variable data (datasets)

You can produce more realistic tests by changing them to use datasets. During execution, a test that uses a dataset replaces a value in the recorded test with variable test data that is stored in the dataset. This substitution allows each virtual user to generate a different request to the server.

Dataset overview

Datasets provide tests with variable data during a run. When you record a test, you perform a sequence of steps that you expect a typical user to perform. After the recording, a test is generated that captures these interactions. When you run this test, it uses the same data that you used during recording. To vary the data in the test, you use a *dataset*, that contains variable data. At run time, this variable data is substituted for the data in the recorded test.

If you need to create a dataset with many records, you can initialize the dataset quickly by importing data from a comma-separated-value (CSV) file. Also, you can export test data from your dataset into a CSV file to enable you to maintain large volumes of test data as a spreadsheet for reuse. Earlier to 9.5, the dataset (formerly known as datapool) was in .datapool format and starting from the 9.5 release, the dataset is in the csv format.

You can copy the CSV file and paste into your project to import the data from a CSV file and create a dataset. Similarly, to export the dataset values as a CSV file, you must copy the dataset from your project and paste it into your local machine.



Note: Alternatively, you can use the **Import** option available in the **CSV editor** to import the data from a CSV file. For more information, see [Editing datasets on page 300](#).

Perform the following steps should you plan to create a test that searches the IBM® website for three items: IBM® Rational® Performance Tester, IBM® Rational® Functional Tester, and IBM® Rational® Manual Tester:

1. Record a test that searches for one item. For more information, see [Recording an HTTP test](#).
2. Create a dataset and associate it with the test. For more information, see [Creating a dataset associated with a test on page 288](#).
3. Associate a request in the test with a column in the dataset. For more information, see [Associating a test value with a dataset column](#).
4. Add a loop in the test to fetch the values from different rows of a dataset. A test without a loop fetches the value only from the first row of the dataset. For more information, see [Adding a loop to a test](#).

Creating a dataset associated with a test

You can create a dataset that contains variable data for tests to use when they run. This is the preferred way to create a dataset because the dataset is automatically associated with a test. You can create anything from an empty dataset that contains one column, which you can edit later, to a fully functioning dataset.

1. In the Test Navigator, browse to the test and double-click it.

Result

The test opens.

2. In the **Test Contents** area, click the name of the test.
3. In the **Common Options** tab, click **Add Dataset**.

The options listed in the following table, enable you to create anything from a simple dataset that you can edit later to a complete dataset.

To create	Do this in the Test Editor - Add Dataset window
A one-column dataset with a default access mode.	In Existing datasets in workspace , select <i>New Dataset<testname>.csv</i> , and click Finish . You can optionally name the dataset column in this session, and you can add other columns and data later.
A one-column dataset and choose the access mode.	In Existing datasets in workspace , select <i>New Dataset<testname>.csv</i> , and click Next . You can optionally name the dataset column in this session and you are prompted for the access mode. You can add other columns and data later.
An association between the test and an existing dataset.	Select the dataset. The dataset is associated with the test, and you can optionally set the access mode in this session.
A new, fully functioning dataset.	Select a project and click Use wizard to create new dataset .

4. Select **Open mode** for the dataset. This mode determines the view that virtual users have of the dataset. Different tests can open the same dataset differently, and you can change the open mode later by opening the test and double-clicking the dataset title.

Option	Description
Shared (per test execution) (default)	<p>When you choose the Shared (per test execution) option, the virtual users running in the test share the dataset values in sequential order.</p> <p>For example, if your dataset has 10 rows, the dataset values are taken from row 1, row 2, row 3, and so on when you select this option.</p>
Private	<p>Each virtual user draws dataset values from a private view of the dataset, with dataset rows apportioned to each user in sequential order.</p> <p>This option ensures that each virtual user gets the same data from the dataset in the same order. Be-</p>

Option	Description
	<p>cause each user starts with the first row of the dataset and accesses the rows in order, different virtual users will use the same row. The next row of the dataset is used only if you add the test that is using the dataset to a loop in the schedule with multiple iterations.</p>
<p>Shared (for all test executions)</p>	<p>When you choose the Shared (for all test executions) option, the virtual users running in multiple tests share the dataset values from the current row.</p> <p>For example, if your dataset has 10 rows and when you set the current row as row 5, the dataset values are taken from row 5 instead of row 1 when you select this option. If you had set the current row as row 1 and used the dataset values until row 5, the dataset values are retrieved from row 6 when you run the test next time.</p>

5. If you are setting how the test accesses the dataset during this session, select one of the following options:
 - **Sequential:** Rows in the dataset are accessed in the order in which they are physically stored in the dataset file, beginning with the first row and ending with the last.
 - **Random:** Rows in the dataset are accessed in any order, and any given row can be accessed multiple times or not at all. Each row has an equal chance of being selected each time.
 - **Shuffled:** Before each dataset access, the order of the rows is changed that results in a different sequence. The rows are accessed randomly but all rows must be selected once before a row is selected again.
6. Select one of the following options.

Option	Description
<p>Wrap when the last row is reached</p>	<p>By default, when a test reaches the end of a dataset or dataset segment, it reuses the data from the beginning. To force a test to stop at the end of a dataset or segment, clear the Wrap when the last row is reached check box. Forcing a stop might be useful if, for example, a dataset contains 15 records, you run a test with 20 virtual users, and you do not want the last five users to reuse information. Although the test is marked Fail because of the forced stop, the</p>

Option	Description
	<p>performance data in the test is still valid. However, if reusing dataset data does not matter to your application, the default of wrapping is more convenient. With wrapping, you need not ensure that your dataset is large enough when you change the workload by adding more users or increasing the iteration count in a loop.</p> <p> Note:</p> <ul style="list-style-type: none"> ◦ With Random access order, Wrap when the last row is reached option is unavailable because you never reach the end of the row. ◦ With Shuffled access order, if you select Wrap when the last row is reached option, you resume selecting from the beginning of the row with the same access order after each row has been selected once. No more selections are required if you clear the Wrap when the last row is reached option.
Fetch only once per user	<p>By default, one row is retrieved from the dataset for executing each test, and the data in the dataset row is available to the test only for the duration of the test. Select Fetch only once per user to specify that every access of the dataset from any test being run by a particular virtual user will always return the same row.</p>

Example

To illustrate how these options affect the rows that are returned, assume that a test contains a loop which accesses a dataset. The loop has 2 iterations. The following table shows the row that is accessed in each iteration:

Dataset option	Iteration 1	Iteration 2
Sequential and Private	row 1	row 2

Dataset option	Iteration 1	Iteration 2
Shared and Shuffled	row x	row y
Fetch only once per user	row x	row x

- If you are creating a fully functioning dataset, you can optionally import the data from a CSV file during this session by copying the CSV file and pasting into your project. For more information on importing dataset, see [Editing datasets on page 300](#).

Creating a dataset in a workspace

You can create datasets in a workspace containing variable data that tests use when they run. You can use this method to create a dataset if you have not yet created the test that will use it.

- Click **File > New > Dataset**.
- In the **New Dataset** window, click the project that contains the dataset. The project is displayed in the **Enter, create, or select the parent folder** field.
- In the **Name** field, type the name of the dataset, and then click **Next**.
- In the window for describing the dataset, add a description.
- In the **Dimensions** field, specify the number of rows and columns for the dataset that you want to create.
- Click **Finish**.

Results

The new dataset opens in a browser. For instructions on how to add data to or edit the dataset, see [teditdp_perf.dita on page 300](#).

What to do next

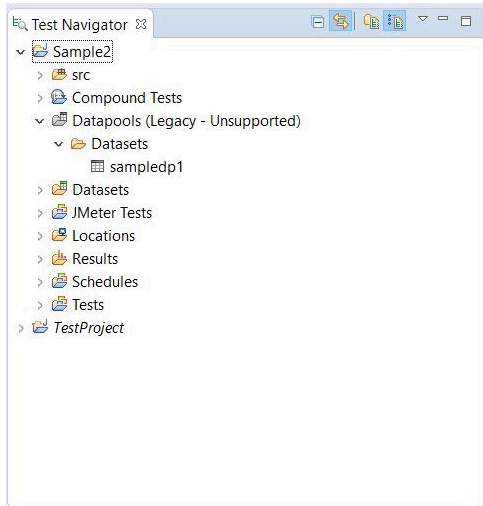
After you have created a dataset and added data to it, you must associate a value in the test with a column in the dataset.

Converting an existing datapool to a dataset

Starting from 9.5 the dataset formerly known as datapool is in the CSV format. You can convert any existing datapool to a dataset.

About this task

When you open the workspace earlier to 9.5 in IBM® Rational® Performance Tester 9.5, the existing datapools in the workspace are stored in the Datapools (Legacy-Unsupported) folder as shown in the following figure.



To convert the existing datapool to a dataset:

1. In the Test Navigator, browse and select the existing datapool.
2. Right-click and select **Convert to Dataset....** Verify that the name of the dataset is the name of the existing datapool and format is .csv.
3. Click **Finish**. The converted datapool opens in a CSV editor.

What to do next

After you have created a dataset and added data to it, you must associate a value in the test with a column in the dataset.

Creating datasets with multiple substitutions

Earlier to 9.2, you could substitute one dataset value at a time. Starting from 9.2, after the test is generated, you can view all the dataset candidates, add multiple candidates as dataset values, substitute values, and create a new dataset out of it. You can also substitute multiple dataset candidates for an existing dataset.

About this task

When you substitute multiple dataset candidates to create a new dataset, the same number of columns are created in the dataset. The names of the candidates become the names of columns and values in the dataset. When you substitute multiple dataset candidates in an existing dataset, the column names in the dataset are retained. If the number of substitutions chosen was greater than the number of columns in the dataset, the extra number of substitutions are added as columns in the dataset. For instance, if a dataset has three columns and you substitute five dataset candidates, two new columns are created by using the names of the dataset candidates.


To create a dataset from multiple dataset candidates:


1. In the Test Editor, select the name of the test and from the Test Details section, select **Common Options** and click **Show Dataset Candidates**.
Alternative: After the test generation when you open the test, you are prompted that *"Some test data may need to be correlated or substituted"*. If you click **Yes**, you can see the list of dataset candidates.
2. Select the dataset candidates that you want to add as values to the dataset and click **Substitute multiple candidates**.
The **Add Dataset** dialog shows the list of datasets that are in the project but not associated with the test.
3. To associate an existing dataset with the test and assign the selected dataset candidates as values and substitutions, select a dataset and click **Next**. To associate a new dataset with the test, click the **Use wizard to create new Dataset** and click **Next**.
4. Select **Open mode** for the dataset. This mode determines the view that virtual users have of the dataset. Different tests can open the same dataset differently, and you can change the open mode later by opening the test and double-clicking the dataset title.

Option	Description
Shared (per test execution) (default)	<p>When you choose the Shared (per test execution) option, the virtual users running in the test share the dataset values in sequential order.</p> <p>For example, if your dataset has 10 rows, the dataset values are taken from row 1, row 2, row 3, and so on when you select this option.</p>
Private	<p>Each virtual user draws dataset values from a private view of the dataset, with dataset rows apportioned to each user in sequential order.</p> <p>This option ensures that each virtual user gets the same data from the dataset in the same order. Because each user starts with the first row of the dataset and accesses the rows in order, different virtual users will use the same row. The next row of the dataset is used only if you add the test that is using the dataset to a loop in the schedule with multiple iterations.</p>
Shared (for all test executions)	<p>When you choose the Shared (for all test executions) option, the virtual users running in multiple tests share the dataset values from the current row.</p>

Option	Description
	For example, if your dataset has 10 rows and when you set the current row as row 5, the dataset values are taken from row 5 instead of row 1 when you select this option. If you had set the current row as row 1 and used the dataset values until row 5, the dataset values are retrieved from row 6 when you run the test next time.

5. If you are setting how the test accesses the dataset during this session, select one of the following options:
- **Sequential:** Rows in the dataset are accessed in the order in which they are physically stored in the dataset file, beginning with the first row and ending with the last.
 - **Random:** Rows in the dataset are accessed in any order, and any given row can be accessed multiple times or not at all. Each row has an equal chance of being selected each time.
 - **Shuffled:** Before each dataset access, the order of the rows is changed that results in a different sequence. The rows are accessed randomly but all rows must be selected once before a row is selected again.
6. Select one of the following options.

Option	Description
Wrap when the last row is reached	<p>By default, when a test reaches the end of a dataset or dataset segment, it reuses the data from the beginning. To force a test to stop at the end of a dataset or segment, clear the Wrap when the last row is reached check box. Forcing a stop might be useful if, for example, a dataset contains 15 records, you run a test with 20 virtual users, and you do not want the last five users to reuse information. Although the test is marked Fail because of the forced stop, the performance data in the test is still valid. However, if reusing dataset data does not matter to your application, the default of wrapping is more convenient. With wrapping, you need not ensure that your dataset is large enough when you change the workload by adding more users or increasing the iteration count in a loop.</p> <p> Note:</p>

Option	Description
	 <ul style="list-style-type: none"> With Random access order, Wrap when the last row is reached option is unavailable because you never reach the end of the row. With Shuffled access order, if you select Wrap when the last row is reached option, you resume selecting from the beginning of the row with the same access order after each row has been selected once. No more selections are required if you clear the Wrap when the last row is reached option.
Fetch only once per user	<p>By default, one row is retrieved from the dataset for executing each test, and the data in the dataset row is available to the test only for the duration of the test. Select Fetch only once per user to specify that every access of the dataset from any test being run by a particular virtual user will always return the same row.</p>

Example

To illustrate how these options affect the rows that are returned, assume that a test contains a loop which accesses a dataset. The loop has 2 iterations. The following table shows the row that is accessed in each iteration:

Dataset option	Iteration 1	Iteration 2
Sequential and Private	row 1	row 2
Shared and Shuffled	row x	row y
Fetch only once per user	row x	row x

7. Click **Finish**.

How dataset options affect values that a virtual user retrieves

The Open, Access, and Wrap modes that you select for a dataset affect the values that a virtual user retrieves.

The following table lists the most common types of datasets and the options that you select to create them.

Dataset purpose	Access		
	Open mode selection	Access mode selection	Wrap mode selection
The virtual user retrieves the value from the current row of the dataset in a random order for every attempted transaction. Note that before accessing each row of the dataset the order of the rows is rearranged.	Shared (for all test executions)	Shuffled	Fetch only once per user
The virtual user retrieves the value from the current row of the dataset in sequential order for every attempted transaction.	Shared (for all test executions)	Sequential	Fetch only once per user
The virtual user retrieves the value from the beginning of the row of a dataset in a random order for every attempted transaction.	Shared (per test execution)	Random	Wrap when the last row is reached
The virtual user retrieves the value from the current row of a dataset in sequential order for every attempted transaction. When a test reaches the end of a dataset, it reuses the data from the current row selection of the dataset.	Shared (for all test executions)	Sequential	Wrap when the last row is reached

Enabling a test to use a dataset

Before a test can use variable data from a dataset, you must update the test to include a reference to that dataset.

1. In the Test Navigator, browse to the test and double-click it. The test opens.
2. Right-click the test name, and click **Add > Dataset**.

Result

The **Select Dataset File** window is displayed listing the datasets available to the test. If a test is already using a dataset, it does not appear in the list.

3. In the **Existing Dataset in workspace** list, click the name of the dataset that your test will use and click **Next**.
4. Select the **Open mode** for the dataset. This mode determines the view that virtual users have of the dataset. This option is useful when you do a parallel test run.

Option	Description
Shared (per test execution) (default)	When you choose the Shared (per test execution) option, the virtual users running in the test share the dataset values in sequential order.

Option	Description
	For example, if your dataset has 10 rows, the dataset values are taken from row 1, row 2, row 3, and so on when you select this option.
Private	<p>Each virtual user draws dataset values from a private view of the dataset, with dataset rows apportioned to each user in sequential order.</p> <p>This option ensures that each virtual user gets the same data from the dataset in the same order. Because each user starts with the first row of the dataset and accesses the rows in order, different virtual users will use the same row. The next row of the dataset is used only if you add the test that is using the dataset to a loop in the schedule with multiple iterations.</p>
Shared (for all test executions)	<p>When you choose the Shared (for all test executions) option, the virtual users running in multiple tests share the dataset values from the current row.</p> <p>For example, if your dataset has 10 rows and when you set the current row as row 5, the dataset values are taken from row 5 instead of row 1 when you select this option. If you had set the current row as row 1 and used the dataset values until row 5, the dataset values are retrieved from row 6 when you run the test next time.</p>

5. Select the **Access mode** for the dataset:

- **Sequential:** Rows in the dataset are accessed in the order in which they are physically stored in the dataset file, beginning with the first row and ending with the last.
- **Random:** Rows in the dataset are accessed in any order, and any given row can be accessed multiple times or not at all. Each row has an equal chance of being selected each time.
- **Shuffled:** Before each dataset access, the order of the rows is changed that results in a different sequence. The rows are accessed randomly but all rows must be selected once before a row is selected again.

6. Select whether the test will reuse data when it reaches the end of the dataset.

By default, when a test reaches the end of a dataset or dataset segment, it reuses the data from the beginning. To force a test to stop at the end of a dataset or segment, clear the **Wrap when the last row is reached** checkbox. Forcing a stop might be useful if, for example, a dataset contains 15 records, you run a test with 20 virtual users, and you do not want the last five users to reuse information. Although the test is marked Fail because of the forced stop, the performance data in the test is still valid. However, if reusing dataset data does not matter to your application, the default of wrapping is more convenient. With wrapping, you need not ensure that your dataset is large enough when you change the workload by adding more users or increasing the iteration count in a loop.

7. Select whether the test will make the data in the dataset record permanent for each virtual user.

By default, one row is retrieved from the dataset for executing each test, and the data in the dataset row is available to the test only for the duration of the test. Select **Fetch only once per user** to specify that every access of the dataset from any test being run by a particular virtual user will always return the same row.

To illustrate how these options affect the rows that are returned, assume that a test contains a loop which accesses a dataset. The loop has 2 iterations. The following table shows the row that is accessed in each iteration:

Dataset option	Iteration 1	Iteration 2
Sequential and Private	row 1	row 2
Shared and Shuffled	row x	row y
Fetch only once per user	row x	row x

8. Click **Finish**.

Result

A reference to the dataset is added to the test, and the **Test Details** area is updated with the dataset information.

9. Save the test.

What to do next

Now that you have created a reference between the test and the dataset, the next step is to associate a value in the test with a column in the dataset.

Viewing dataset candidates when you open a test

Dataset candidates are displayed automatically when you open a test for the first time. From the dataset candidates window you can view the dataset candidates in the test, bookmark locations of interest, and add or remove dataset references.

1. Record a test.

Result

When the test opens for the first time in the Test Navigator, the **Show Dataset Candidates** window is displayed. The **Show Dataset Candidates** window is displayed only if there are dataset candidates and if **Always display this dialog when a test is first opened** is selected. To prevent the **Show Dataset Candidates** from being displayed when a test opens, clear the **Always display this dialog when a test is first opened** check box in the **Show Dataset Candidates** window.

- Do one of the following:

Option	Description
<p>To view details about the dataset candidates in a test</p>	<p>Navigate through the Dataset Candidates field to see them previewed in the Preview pane. Click the Next and Previous icons to move the selection down or up in the list of dataset candidates. Click the Show as Tree icon to toggle between tree format and list format. Click the Sort icon to sort the list of dataset candidates. Click the Bookmark icon to bookmark a location for later review.</p>
<p>To select a data source for a dataset candidate</p>	<p>Select the dataset candidate in the Dataset Candidates field, and then click Substitute. The Select Data Source window opens.</p>
<p>To find more values in the test that have the same value as the selected dataset candidate</p>	<p>Click Find More and Substitute. These values can be reviewed and substituted interactively as needed.</p>
<p>To remove a substitution</p>	<p>Select a substitution site, and then click Remove Substitution.</p>

- Click **Close** to close the **Show Dataset Candidates** window and proceed to the test in the test editor.
To display the **Show Dataset Candidates** window again while in the test editor, click the root node of the test. Then click the **Common Options** tab under **Test Element Details**, and then click **Show Dataset Candidates**.

Editing datasets

You can add, modify, remove, import, or export data from a dataset by using the CSV Editor. The working principle of the CSV Editor is similar to that of a spreadsheet.

Before you begin

You must have created a dataset. See [Creating a dataset in a workspace on page 292](#).

About this task

In 9.5.0 or later, you can use the CSV Editor to view and edit data in the dataset. You can also view the datasets in other editors by right-clicking the dataset and selecting the **Open With** option.

You can perform basic tasks in the CSV Editor by right-clicking any row, column, or cell of the dataset to organize your data in a better way. For example, updating the data in a cell, inserting or deleting rows and columns, or renaming column names.

When you edit the dataset in a CSV Editor, you can use the following keyboard shortcuts to control the cursor selection in the CSV Editor:

- **Tab** - To move the cursor control to the next available option.
- **Shift-Tab** – To move the cursor control to the previous option.
- **Shift+F10** – To open the context menu from the dataset cell.



Note: You cannot resize the width of rows in the CSV Editor. When you have a large amount of data in a cell, you can right-click the cell and select **Copy** (or Ctrl+C), and then paste it into a text-editing program to view the content. Alternatively, you can hover the mouse over the cell to view the content.

When you have a CSV file that has data separated from a character, then you can import that CSV file into the dataset. You can select any of the following separator characters from the **Configure Dataset** window, and the selection can be the separator character that you used in the CSV file:

- Comma
- Semicolon
- Space
- Tab
- Other

Consider that you have the data in the CSV file in the following format:

Name;CCNum
John;1234 5678 1234 5678
Bob;1122 3344 5566 7788
Amy;2233 4455 6677 8899

When you import the CSV file in the dataset, and then select the separator value as **Semicolon**, the data in the dataset is displayed as follows:

	Name	CCNum
1	John	1234 5678 1234 5678
2	Bob	1122 3344 5566 7788
3	Amy	2233 4455 6677 8899

If you want the data in its original format, that is, a semicolon (;) character to separate the data, then you can choose any other separator value from the **Configure Dataset** window.





Note: The default separator value is **Comma**.









1. Double-click the dataset that you want to edit in the **Test Navigator**.





Result



The dataset opens in the CSV Editor in a browser.


2. Perform the following actions to use the options available in the CSV Editor:

Options	Actions
Find and Replace 	<p>To find:</p> <ol style="list-style-type: none"> a. Click the Find and Replace icon . b. Enter the content that you want to search in the Find field. c. Select any or all the following options to find the search content more effectively: <ul style="list-style-type: none"> ▪ Select the Case sensitive check box to search the content that is the exact letter case of the content entered in the Find field. ▪ Select the Match entire cell contents check box to search for cells that contain only the characters that you have entered in the Find field. ▪ Select the Search using regular expression check box to search the pattern that matches strings. <p>For example, to search a cell that contains any number between 0 to 9, do the following:</p> <ol style="list-style-type: none"> i. Enter <code>\d</code> in the Find field. ii. Select the Search using regular expression check box. iii. Click Find. d. Click Find. If the text is found, the cell containing that text is selected. e. Click Find again to find further instances of the search text. <p>To find and replace:</p>

Options	Actions
	<p>a. Click the Find and Replace icon  .</p> <p>b. Enter the content that you want to search in the Find field.</p> <p>c. Enter the content that you want to replace in the Replace field.</p> <p>d. Select any or all the following options to find and replace the content more effectively:</p> <ul style="list-style-type: none"> ▪ Select the Case sensitive check box to find the content that is the exact letter case of the content entered in the Find field. ▪ Select the Match entire cell contents check box to find and replace for cells that contain only the characters that you have entered in the Find and Replace fields. ▪ Select the Search using regular expression check box to find and replace the pattern that matches strings. <p>e. Click Replace to replace the individual instances.</p> <p>f. Click Replace All to replace every instance of the content throughout the dataset.</p>
Undo 	<p>a. Click the Undo icon  .</p> <p>b. Select the recent changes from the list that you want to undo, and then click the list.</p> <p>The Undo option undoes anything you do in the dataset. The CSV Editor saves the unlimited undo-able action. You can perform the undo action even after you save your changes made to the dataset.</p>
Redo 	<p>a. Click the Redo icon  .</p> <p>b. Select the recent changes from the list that you want to redo, and then click the list.</p> <p>The CSV Editor saves the unlimited redo action.</p>
Import 	<p>a. Click the Import icon  .</p> <p>b. Click Choose File and select the CSV file that you want to import in the Import File dialog.</p> <p> Note: If the CSV file contains test data with Unicode characters in it, you must save the CSV file in UTF-8 format. You can then choose the CSV file and import the test data from the CSV file into the dataset.</p> <p>c. Optional. Click Overwrite to add the rows and columns from the selected CSV file from the beginning of the dataset.</p>

Options	Actions
	<p>d. Optional. Click Append to add rows and columns from the selected CSV file to the end of the dataset.</p> <p>e. Optional. Select the First row contains headers check box if your CSV file contains the header.</p>
Export 	Click the Export icon  to download the dataset as a CSV file.
Set as current row	<p>Right-click any cell in a row and select Set as current row.</p> <p>When rows are deleted:</p> <p>If you delete any row between row 1 to current row, the current row data is taken from the next row.</p> <p>For example, when you set the current row as 6, and then you delete any row between row 1 to row 6, the current row remains at row 6, but the content of row 7 is moved to row 6.</p> <p>When rows are inserted:</p> <p>If you insert any new row between row 1 to the current row, the current row data is taken from the previous row.</p> <p>For example, when you set the current row as 6, and then you insert any row between row 1 to row 6, the current row remains at row 6, but the content of row 5 is moved to row 6.</p>
Dataset configuration settings 	<p>In the Configure Dataset window, you can set the separator value, change the row and column settings, and configure the string values in the dataset.</p> <ol style="list-style-type: none"> Click the Menu icon  , and then select the Configure option. Select any of the separator values that you used in the CSV file. <p>The available options are Comma, Semicolon, Space, Tab, and Other. In the CSV file, if you have any other separator characters other than the available options, then you can select the Other option, and then can specify a value.</p> <p>For example, if the data in the CSV file is separated by a character #, then select the Other option and enter # in the field.</p> Configure the following options to change the row and column settings:

Options	Actions
	<ul style="list-style-type: none"> ▪ Column header - Use an up-down control button to increment or decrement the value of the column header. ▪ Data start point - Use an up-down control button to increment or decrement the value of the data starting pointer. ▪ Current row - Use an up-down control button to increment or decrement the value of the current row. <p>d. Configure the following options to change the string values in the dataset:</p> <ul style="list-style-type: none"> ▪ Treat as null - Enter a string value that is to be treated as null when running the test. ▪ Treat as empty - Enter a string value that is to be treated as empty when running the test. <p>For example, when you run the test and the data 123 in the dataset to be treated as empty, then you can specify 123 in the Treat as empty field.</p> <ul style="list-style-type: none"> ▪ Treat empty text as null - Select this field when you want the dataset that contains any blank cells, and the value of those blank cells to be interpreted as null. <p>e. Click Update to apply the changes.</p>
Discard 	Click the Menu icon  , and then select Discard to discard the changes made to the dataset.

3. Click the **Save** icon  to save the changes made to the dataset.

Results

You have edited the dataset.

Encrypted datasets overview

You can encrypt one or more columns in a dataset. If you want to encrypt confidential information such as a set of passwords or account numbers that are used during a test, you can use an encrypted dataset.

Dataset columns are encrypted using the RC4 private-key algorithm. You can use only one password to encrypt columns in any given dataset. Encrypted datasets are not checkmark_perf.jpg on agent computers that are running the z/OS® or AIX® operating systems.



Important: If you forget the password to a dataset, there is no way to recover the password.

When you run a test that uses a dataset that contains encrypted variables, you are prompted for the dataset password. If the test uses multiple encrypted datasets, you must enter the password for every encrypted dataset that the test uses.

When you run a test that uses a dataset with an encrypted column, the value of the column is decrypted at a run time. The column value is sent as a cleartext string in the requests to the server. The actual values of the encrypted dataset variables are not displayed in the test log. The test log displays asterisks for the encrypted dataset variables.

To see the actual values of variables that are sent to the server at run time, you must use custom code. You can send the dataset column value to custom code that writes the value to a file other than the test log. If the custom code writes to the test log using `tes.getTestLogManager().reportMessage()`, then asterisks are displayed instead of the decrypted variables.

Encrypting a dataset column

To secure test data, you must encrypt datasets. You can encrypt data in the columns of a dataset by using an encryption key. When you run a test that utilizes a dataset with encrypted variables, you must enter the encryption key for the encrypted column that the test uses.

Before you begin

You must have created a dataset. See [Creating a dataset in a workspace on page 292](#).

1. Double-click the dataset in the Test Navigator.

Result

The dataset is displayed in a browser.

2. Right-click any cell in a column that you want to encrypt and select **Encrypt column data**.

Result

The **Encrypt Column** window is displayed.

3. Enter an encryption key in the **Encryption Key** field to encrypt the data in the column.



Remember: When you have already encrypted other columns in the dataset, you must enter the same encryption key that you used previously. You can use only one encryption key to encrypt columns in a dataset.



Important: The encryption keys you use to encrypt data in a dataset are not stored on the server nor can be retrieved from the server. Therefore, you must remember to store the encryption keys in a secure location. You must use the same encryption keys to view the encrypted values, to decrypt data, or enable the use of the encrypted dataset during test runs.

4. Click **Encrypt Column**.

Result

Asterisks are displayed instead of actual data for the encrypted column.

Results

The dataset column is encrypted.

Decrypting a dataset column

To view the content of an encrypted dataset, you can decrypt the dataset. Removing encryption from a dataset revokes the protection offered to the test data.

Before you begin

You must have created a dataset with at least one encrypted column. See [Creating a dataset in a workspace on page 292](#) and [Encrypting a dataset column on page 306](#).

1. Double-click the dataset in the Test Navigator.

Result

The dataset is displayed in a browser.

2. Right-click encrypted cells that display the contents with asterisks, and then select **Decrypt column data**.

Result

The **Decrypt Column** window is displayed.

3. Enter the encryption key that you used to encrypt the data in the column in the **Encryption Key** field.
4. Click **Decrypt Column**.

Result

Asterisks are replaced with the actual data in the decrypted column.


Results

The encryption is now removed from the selected column in the dataset. When you run a test that uses a dataset that contains decrypted data, the variable data is substituted for the data in the recorded test without prompting for the encryption key.

Navigating between a dataset and a test

After you have created a dataset or imported a comma-separated values (CSV) file into a dataset, you can navigate between the dataset and associated tests in the test editor. You can enlarge the test and the dataset, list the datasets that a test uses, navigate from a row in a dataset to the corresponding element in the test, see the data for a page or request, and add or remove dataset references.

1. In the Test Navigator, browse to the test and double-click it. The test opens.
2. Do one of the following actions:

Option	Description
Maximize the test window	Double-click the test tab (for example, ). Do not click the x , or you will close the test. To return to the default perspective, click Window > Reset Perspective .
View the datasets that a test uses	In the Test Contents area, click the first line of the test, which is the test name.
Navigate from a row in a dataset to its corresponding element	<ol style="list-style-type: none"> In the Test Contents area, click the test name, which displays the dataset. Expand the dataset to display the rows. Double-click the row.
View the data for a page or request	In the Test Contents area, click the page or request.
To add a reference to a dataset	In the Test Element Details area, drag your cursor over the candidate, right-click, and select Substitute > Select Data Source . The Select Data Source window opens. If you have not already added the dataset to the test, click Dataset , and then add the new dataset.
Remove a reference to a dataset	In the Test Element Details area, drag your cursor over the reference, right-click, and select Remove Substitution .

- Save the test, if you have made any changes.

Test variables

A test variable is a user-defined, name-value pair that stores and refers to information throughout a test and between tests.

A variable is declared in the test variables section of the test. You can use it throughout the test as a reference for any field that can be substituted. Substituting data from a test variable is achieved using the **Test Variables** page of the Test Data Source view. You can do the following actions to a test variable:

- Provide a default value to the variable during declaration.
- Change the value of the variable using Set Variable statement. You can use the Add and Insert menus of the Test Editor to create Set Variable statements.
- Set hard-coded value or value retrieved from a data source, such as dataset or reference that appears before the Set statement to the variable.

If a variable is initialized at various places such as test, compound test, schedule, or user group, the product uses the following order to initialize the value of the variable when running the test. The variable set in the variable table of the compound test editor takes the highest precedence followed by others:

1. Compound test setting in the variable table UI
2. Compound test specified in a var file
3. User group setting in the variable table UI
4. User group specified in a var file
5. Schedule specified setting in the variable table UI
6. Schedule specified in a var file
7. Command line



Note: You must select **All tests for this user** from the **Visible in** drop-down list to take the precedence of variable initialization.

Sharing variables among tests

In order to share variables between tests, all the tests must contain the variables with the same name. The variables must also have **Visible in** set to **All tests for this user**. When these conditions are met and multiple tests have been placed in a schedule, then variable in the dataset of one test can be used in the other test.

A common reason to share data between tests is to perform data correlation. With data correlation, a variable is set to a response that comes from a request in one test and is used in requests performed in a different test. Assume that you are testing an employee database. The Create Employee test creates an employee record and the Modify Employee test modifies an employee record. When a new record is created, it is assigned a record ID. Variables can be used to pass the record ID from a response in the Create Employee test to the Modify Employee test. A user-defined variable is not shared among different virtual users. The variable is shared only among the different tests of the same virtual user. If you set **Visible in** to **This test only**, then dataset from a test is not available to another even if both tests contain variables with the same name.

If you want to share variables between the different types of test scripts in your product, consider the following points:

- Declare the test variables with the same name across all the test scripts for the variables to communicate with each other. Set **Visible in** to **All tests for this user**.
- Include the required test scripts into a compound test.

Using test steps, you can share the default values of the variables to another test script. You can also assign new values to the variables and use the latest values in another test script.

If you want to share variables between the test scripts of different testing products such as Rational® Functional Tester, Rational® Performance Tester, or Rational® Integration Tester, you must consider the following points:

- If you are using Installation Manager, you must shell-share or install the products in the same package group.
- Declare the test variables with the same name across the Rational® Functional Tester and Rational® Performance Tester test scripts. Set **Visible in to All tests for this user**.
- Include the required test scripts into a compound test.
- If you are using Rational® Integration Tester test scripts, you must map your tags with the test variables of Rational® Functional Tester or Rational® Performance Tester.

Using variables to access datasets

You can define variables so that they share data from a dataset throughout tests. This is achieved by having the value field of a Set Variable statement substituted from a dataset. By doing so, the first test which appears in the schedule can set the variable from a dataset and share it with the other test in a schedule.

Assume that you have two tests that log in to an application using a user ID from a dataset. The first test can set the value of a variable from the dataset, and both tests can use the variable, instead of directly using the dataset. In this case both use the same record from the dataset. This is similar to the fetch-only-once-per-user behavior of a dataset. However, fetching once means that during playback, a virtual user gets only one record from the dataset. The one-record limit holds even if the tests are in a loop, and are run several times by the virtual user. By using the user-defined variables, the virtual user retrieves a new record each time through the loop, and both tests can use the same record.



Note: Assignment (set) operators can not only have a variable value substituted from a dataset, but also in the declaration of a variable. You can substitute the assignment operator and variable value from any data source, and thus that value can be shared between tests as well.

Array variables

You create an array variable to add multiple values to a variable. If you create a secondary HTTP request, add complete paths of the requests in the array variable that can be used a custom code during playback.

Declaring and assigning test variables

When you declare a variable, you can create a container for it, initialize it to a string or a dataset value, and set its scope. Then, within the test, you can reassign another value to the variable.

About this task

If the data that you want to assign to a variable is only available after a specific test step, instead of initializing the variable, you need to add a variable assignment further down in the test, so that when the assignment occurs, the data that you need to use is available. Otherwise, when you try to initialize the variable (or do the assignment), the value that you want to use will not be available and will not show up as an option to select.

If a variable is initialized at various places such as test, compound test, schedule, or user group, the product uses the following order to initialize the value of the variable when running the test. The variable set in the variable table of the compound test editor takes the highest precedence followed by others:

1. Compound test setting in the variable table UI
2. Compound test specified in a var file
3. User group setting in the variable table UI
4. User group specified in a var file
5. Schedule specified setting in the variable table UI
6. Schedule specified in a var file
7. Command line



Note: You must select **All tests for this user** from the **Visible in** drop-down list to take the precedence of variable initialization.

To create, initialize, and assign a value to a test variable:

1. In the Test Navigator, browse to the test and double-click it.

Result

The test opens.

2. To create a container for the test variables that you create in a test:

- a. Open the test, and in the **Test Contents** area, click **Test Variables**.

- b. Select **Add > Test Variable Container**.

Result

A container named **Test Variables** is created for the user-defined variables.

- c. Select the container to rename it.

Result

The **Test Element Details** area opens for you to type a new name in the **Name** field.

3. To declare or define a test variable:

- a. Open the test, and in the **Test Contents** section, click the user-defined container to contain the variable.

- b. To create a variable, select **Add > Variable Declaration**. To create an array variable, select **Add > Array Variable Declaration**.

- c. Type the name of the variable, and click **OK**.

Result

The variable is added as the last element in the container and the **Test Element Details** area opens.

- d. In the **Test Element Details** area, set the scope and initial value for the variable.

Visible in: Select **This test only** to restrict data to the current test only. Even if another test has a variable with the same name, that variable will not change. Select **All tests for this user** to share the value of this variable when the test runs in a schedule. For the variable to be shared, both tests must have a variable with the same name and must have this option enabled.

Check Value: Select **When first used** to check whether or not a variable is initialized only after the test execution reaches the first request that uses a variable. Select **At test start** to check whether or not a variable is initialized when starting the execution of the test. If the variable is not initialized, then an error message is displayed, depending on the behavior set.

If not initialized, set to: Select **Text** to initialize the variable to a specific value whenever the test runs in the schedule. Select **Dataset value** and, in the **Select Data Source** window, select the dataset that will initialize the variable.

Run-time error if variable not initialized: Select the action for the run when it encounters an uninitialized test variable. If you select **Issue test log warning** or **Issue test log error**, verify that the **Test log** page in the schedule sets errors, failures, and warnings to **All**, which is the default setting. If you select **Exit the test**, the schedule continues to run although the virtual users that have the uninitialized variable stop. If you select **Do nothing**, the test continues to run.

4. To assign or initial a value to a test variable:

a. Open the test, and in the **Test Contents** area, select a test element.

b. Select **Insert > Variable Assignment**, which inserts the assignment before the selected element.

Result

The **Test Editor** window opens and lists the variables available to the test.

c. Select the variable that you are assigning a value to and, in the **Set to** box in the **Test Element Details** area, set the value for the variable.

You can set the value to a text string, to any data source that exists in the test before the assignment statement, or to **Not initialized**.

Result

A `set` statement is added to the test, with the value you chose.

Initializing variables from the command line

To initialize test variables from an XML file, you can run the test from the command-line interface using the `varfile` option.

Before you begin

- You must read and familiar with command line parameters. See [Command-line parameters on page 375](#).
- You must have completed the following tasks:

- Declared the variables using Rational® Performance Tester.
- Created an XML file that contains the variables with values. The XML file would have a structure similar to the following image:

```
<?xml version="1.0" encoding="UTF-8"?>
<inits>
  <variable_init value="9.999.99.999" name="hostname"/>
</inits>
```

About this task

If a variable is initialized at various places such as test, compound test, schedule, or user group, the product uses the following order to initialize the value of the variable when running the test. The variable set in the variable table of the compound test editor takes the highest precedence followed by others:

1. Compound test setting in the variable table UI
2. Compound test specified in a var file
3. User group setting in the variable table UI
4. User group specified in a var file
5. Schedule specified setting in the variable table UI
6. Schedule specified in a var file
7. Command line



Note: You must select **All tests for this user** from the **Visible in** drop-down list to take the precedence of variable initialization.

1. Open the command prompt.
2. Run the following command to change the directory to the `cmdline` directory:

```
cd <cmdline_directory_path>
```

For example,

```
cd C:\Program Files\IBM\SDP\cmdline
```

Based on the operating system that you used to install Rational® Performance Tester, you can locate the `cmdline` directory as follows:

Operating system	The default path to the directory
Windows®	C:\Program Files\IBM\SDP\cmdline
Linux™	/opt/IBM/SDP/cmdline
Mac	/Application/IBM/SDP/cmdline

3. Run the following command to run the test:

```
cmdline -workspace workspace_full_path -project proj_rel_path -suite suite_name or suite_rel_path
```

For example,

```
cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "/myproj/proj123" -suite "test1.testsuite"
```

OR

```
cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "/myproj/proj123" -suite "Tests/test1.testsuite"
```



Note: You can add an additional parameter to the preceding command according to your testing requirements and then run the test.

Result

The test run starts and the status is displayed on CLI.

Related information

[Command-line parameters on page 375](#)

Initializing variables from Engineering Test Management

If you want to run an IBM® Rational® Performance Tester test from IBM® Engineering Test Management, you can pass the execution variables defined in Engineering Test Management to the Rational® Performance Tester test.

Before you begin

- Configure the Engineering Test Management adapter in Rational® Performance Tester. For more information, see the [Configuration on page 114](#) topic.
- Variable names must be the same in Engineering Test Management and Rational® Performance Tester.
- The **Visible in** value for the variable in the Rational® Performance Tester test must be set to **All tests for this user**.

About this task

When you pass an execution variable to a Rational® Performance Tester test, the value initialized in the test is replaced by the value in the execution variable. If you modify the value that is initialized in the test, after the test is executed, the modified value is passed back to the execution variable in Engineering Test Management.

To initialize an execution variable value to a test, run the test from Engineering Test Management. For information about execution variables, see [Using execution variables in manual test](#).

Correlating response and request data

For a test to run correctly, a request that is sent to a server might need to use a value that was returned by a previous request. By ensuring that this data is correlated accurately, you can produce better performance tests.

Data correlation overview

A request can include data that was returned in the response to a previous request. Associating data in this manner is called *data correlation*.

Interactions with an application are typically related to each other. For example, consider the following interactions with a web-based application, in which each request depends on information returned from a previous response:

1. A payroll clerk types the web address for an application, which sends a login prompt. When the clerk logs in, the web server returns a page that indicates that login has succeeded and a unique session ID to the web browser that the clerk is using.
2. The clerk clicks a link on the returned page, which requests that the web server open the page for searching the employee database. The web browser includes the session ID when sending the request. Based on the session ID, the web server knows that the request comes from someone who is already logged on, and so opens the search form for the employee database. The clerk then searches for a specific employee. The web server returns a photograph of that employee and the employee's unique ID.
3. The clerk clicks a link that requests the web server to return the payroll record for the employee. With this request, the web browser sends two IDs:
 - The session ID, so that the web server knows that the request comes from some who is logged on
 - The employee ID, so that the web server can locate and return the correct information

In this example, request 2 depends on request 1, and request 3 depends on requests 1 and 2.

If you record these interactions in a test, before running the test with multiple users, you would vary the test data. For example, you would replace the user name and password values, the employee name search values, or both, with values that datasets contain. When you run the test, each virtual user returns a different employee payroll record, based on the contents of the datasets.

In a generated test, where data in a request depends on data that is contained in the response to a previous request, the request data is substituted from the response data on which it depends. The term for this internal linking of response and request data is *data correlation*. When you run a test with multiple users and varied data, data correlation is required to ensure that the test runs correctly.

A *reference* is a value in a test (typically in a response) that can be used by a subsequent value in the test (typically in a request). When the test generator detects that a request value must be substituted from a previous value, it designates the earlier value as a reference and correlates the subsequent request value with the reference. This process is called *automated data correlation*. You can also manually correlate any two values in a test or unlink existing correlations.



Note: You can change or disable automated data correlation. To do so, click **Window > Preferences**, expand **Test**, and then click **Test Generation**.

By default, the empty strings are not correlated because it might increase the time taken to generate a test. However, sometimes empty strings such as spouse name or middle initial become important to correlate. To correlate the empty strings, click **Window > Preferences > Test > Test Generation > HTTP Test Generation > Data Correlation** and select the **Create substitutions for empty strings** check box.

Generally, the HTML response content after the recording appears as `<input type="username" name="User" id="aaa" value="John"/>`. Some applications dynamically update the `name` attribute. Therefore, when you play back the test, the HTML response content appears as `<input type="username" name="idt020" id="aaa" value="John"/>`. Because the `name` attribute changes dynamically, data correlation does not occur and the playback fails. For data correlation to correlate the response content based on the `ID` attribute, ensure that you have selected **ON** in the **Prioritize correlation based on ID** option at **Window > Preferences > Test > Test Generation > HTTP Test Generation > Data correlation**.

To help you work with correlated data, the test editor uses color coding and provides navigational aids:

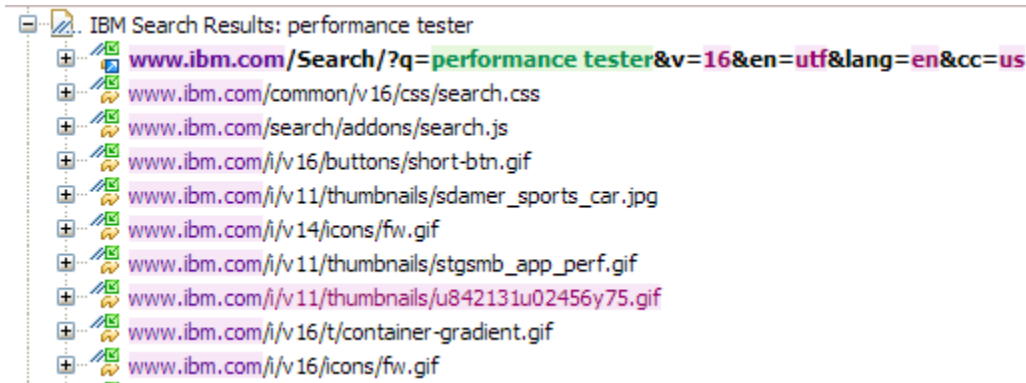
- When you click a page, you see a Test Data table for that page. By default, related dataset candidates are shown in green text on a light green background, values that are already associated with a dataset are shown in white text on a green background, and references are shown in blue text.

Test Data Options ▾

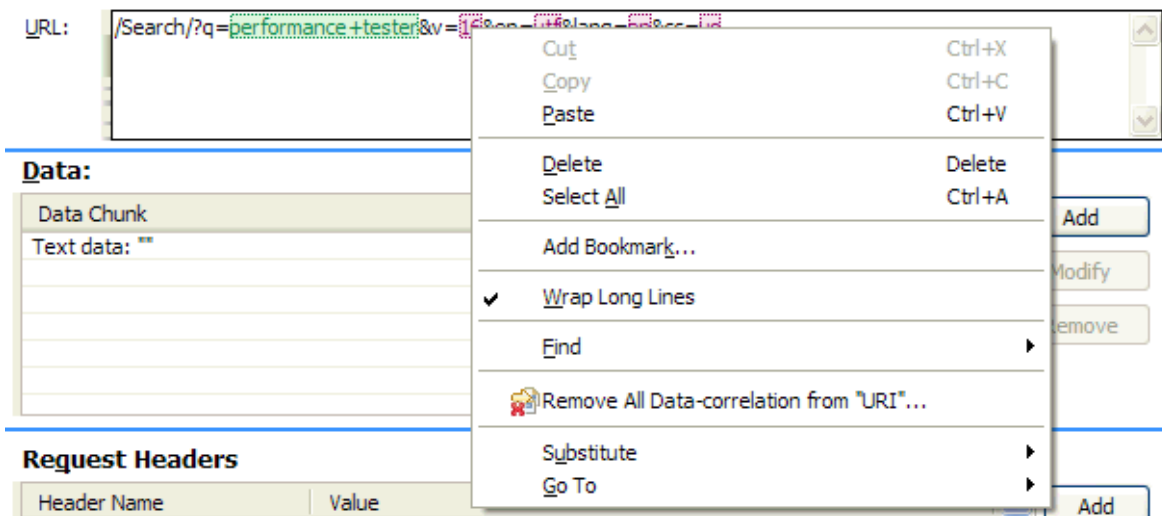
Name	Value	Substituted with
action	shopping	
category	1	
action	getimage	"action2" ("getimage" - Content)
inventoryID	V0001	"inventoryID7" ("V0001" - Content)
action	getimage	"action2" ("getimage" - Content)
inventoryID	V0005	"inventoryID6" ("V0005" - Content)
action	getimage	"action2" ("getimage" - Content)
inventoryID	V0003	"inventoryID5" ("V0003" - Content)

URL Encode

- If correlated data is not displayed, right-click the table and verify that **Show References** is selected. To navigate directly to a page request containing correlated data, double-click a table row. To associate correlated data from this table with a dataset, click the row, click **Substitute**, and then click **Select Data Source** to open the **Select Data Source** window. You can also use the **Test Data Sources** view to make substitutions. In the test editor, right-click the **Test Data** table, and then select **Link with Test Data Sources View**. When you click a row in the **Test Data** table, the **Test Data Sources** view displays information about the selected substitution site.
- When you expand a page, green text indicates page requests that contain dataset data or candidates. Blue text indicates page requests that contain references.



- When you click a highlighted request, dataset candidates are highlighted in light green, data that is associated with a dataset is highlighted in dark green, and correlated data is highlighted in red. If you right-click a value for correlated data, as shown in the example, you can then click **Go To** to see its reference:



- References are highlighted in dark blue.

Viewing data correlation

You can switch between viewing all test elements in the test editor and viewing only elements related to data correlation in the test editor. Viewing only data correlation elements makes it easier to add and remove substitutions.

1. In the Test Navigator, browse to the test, and double-click it. The test opens.
2. In the **Test Contents** area, click **Options**.
3. Click **Show > Data Correlation**.

Result

The test editor window displays only elements that are related to data correlation. Alternately, click **View** under **Test Contents** to switch between **Display all Test Contents** and **Show Substitutions**.

4. Select a single test element in the **Test Contents** area to see the current data source and to remove or change the substitution in the **Test Element Details** area. Select multiple elements in the **Test Contents** area to see

the data in tabular form in the **Test Element Details** area. Different controls are available depending on the type and number of elements that you select in the **Test Contents** area.

5. **Optional:** In the Test Elements Details area, click **Substitute > Select Data Source** to open the **Select Data Source** window, where you can specify the data source for the selected substitution site.

What to do next

To view all test elements, click **Options > Show > Data Correlation** again.



Note: If you select a test element while viewing all test contents, and then switch to viewing only data correlation elements, then the corresponding substituters and dataset candidates are selected. For example, if you select an HTTP page in the test editor, and then switch to viewing only data correlation elements, then all substituters and dataset candidates for all requests from the HTTP page are selected.

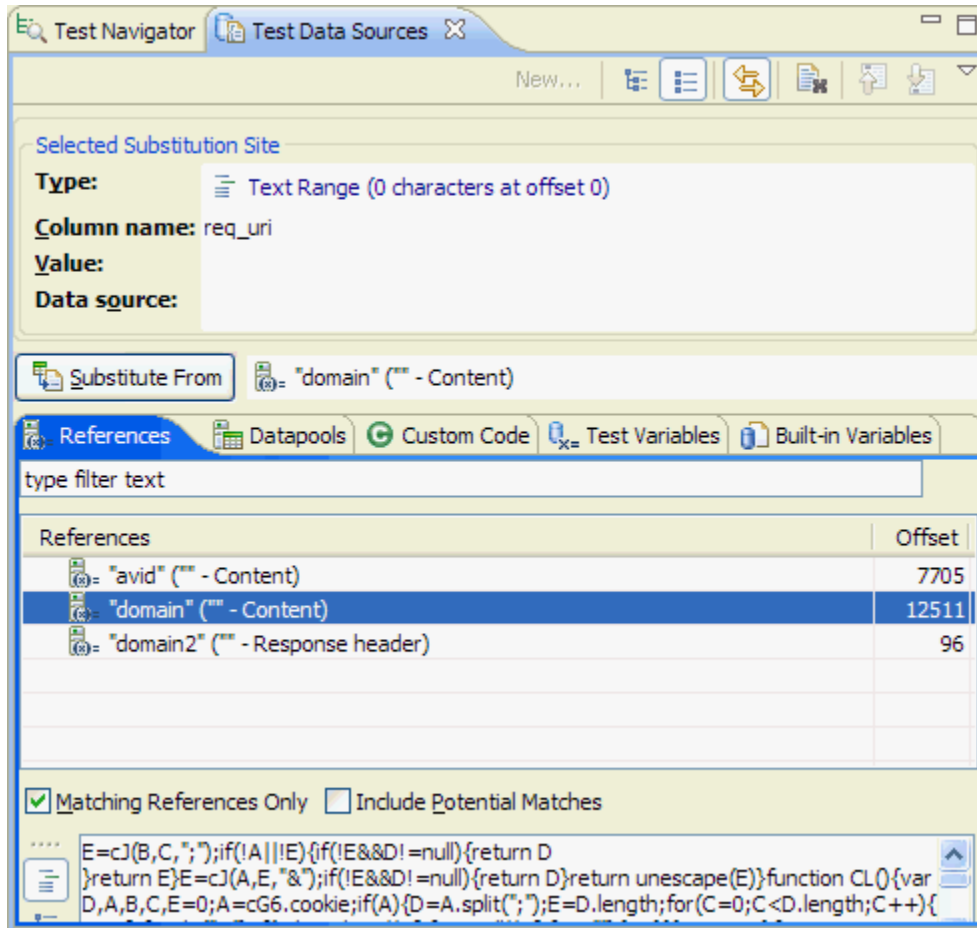
Test data sources overview

Use the **Test Data Sources** window to add or change data correlation for any checkmark_perf.jpg test element.

The test generator attempts to perform automated data correlation. You can also manually correlate request values with other test data. The **Test Data Sources** window displays the following types of test data that you can substitute from:

- References
- Datasets
- Custom code
- Test variables
- Built-in datasources

You can right-click references, custom code, and built-in data sources to display a menu that contains commonly used commands. In addition, you can use the toolbar and menu at the top of the **Test Data Sources** window to complete common operations, such as creating a dataset or switching from tree view to list view. The **Substitute From** push button is enabled when you select a substitution site in the test editor and a data source from one of the five categories in the **Test Data Sources** window. Click **Substitute From** to correlate the data source and the substitution site.



References

The data sources that have been created in the test and the potential matches that are available for a selected substitution site. For example, text that is found in a response and used in a subsequent request is automatically created as a reference. Clear the **Matching References Only** check box to see all the references that occur before the substitution site in the test. Select **Include Potential Matches** to see a list of locations that might match the substitution site. Replace the `type filter text` string with keywords to filter the list of references. Select a reference in the **References** table to display the reference in the pane at the bottom of the window. Use the controls to the left of the preview pane to switch between inline view and tree view.

Datasets

The datasets that have been added to the test. To add a new dataset, click the **Add new Dataset** icon



Custom Code

The custom code that is available as data sources for this substitution site, if you have written Test Execution Services (TES) custom code.

Test Variables

The test variables and their types that are available as data sources for this substitution site. Replace the `type filter text` string with keywords to filter the list of test variables.

Built-in Datasources

The dynamically calculated data sources, such as **Current Date**, **Random Number**, **Sequential Number** and **Timestamp**, that are available as data sources for this substitution site. To create a built-in data source, right-click inside the **Built-in Datasources** page, and select **New**, or click the **New** push button at the top of the window.

To assign a unique value for every substitution, when creating the built-in data sources, select the **Get New Value Each Time Used** check box.

Detaching the Test Data Sources window

The **Test Data Sources** window is detachable. To detach the **Test Data Sources** window, right-click the **Test Data Sources** tab, and then select **Detached**. When detached, the **Test Data Sources** window is always displayed in front of the other windows that make up the workbench.

Correlating a request value with test data

If a test runs without error but does not generate the results that you expect, you might need to correlate a value in a request with other test data.

About this task

You can correlate a request value with the following types of test data:

- References
- Datasets
- Custom code
- Test variables
- Built-in datasources

For example, if you recorded a test and searched on a date, you might want to substitute the built-in data source *Current Date* so that the test will search on the playback date, not on the recorded date. For information on the different types of test data, see [Test data sources overview on page 318](#).

1. In the **Test Navigator**, browse to the test, and double-click it. The test opens in the test editor.
2. Locate the value that the other test data will replace.
3. Highlight the value: Press and hold the left mouse button and drag your mouse over the value.
4. In the **Test Data Sources** window, click the appropriate tab: **References**, **Datasets**, **Custom Code**, **Test Variables**, or **Built-in Datasources**. To see all references, clear the **Matching Only** check box.
5. In the **Test Data Sources** window, select the test data to use. For references and custom code, you can double-click the entry in the **Test Data Sources** window to find the data source in the test editor.
6. Click **Substitute From**.

Result

The value is shown in purple text to indicate that it has been correlated and the correlation is added to the Test Data table, which contains the substitution sites for the page.

Substituting request content with file contents

You can substitute the content portion of a protocol request with the contents of a file. This feature is only checkmark_perf.jpg in certain sections of a test, depending on the protocol. For example, HTTP tests support file content substitution only in the POST data section of a request. SOA tests support file content substitution for MIME and DIME attachments, XML node values and fragments, and text content. File content substitution works in the same manner as other substitutions. All standard test data sources such as test variables, datasets, and references can be used. The data from the source is treated as a full path to a file. The file is opened, its contents are read, and then those contents are used in the substitution.

Before you begin

You must copy the files that contain the substitution content to the agent computers. You must record a test with locations for substitution from a file. For example, record an HTTP test that contains multipart MIME data in a POST request.

1. Create a data source that contains the full path to the file from which you want to substitute content. Specify an absolute path to the file. Use path separator characters appropriate to the operating system of the agent computer running the test. Optionally, specify a character set to use in reading the file. The existence of the file is not validated. If the file cannot be opened when the test runs, a message is written to the test log. If you use path separator characters that are not appropriate for the operating system of the agent computer, the substitution cannot be completed. For example, if you use a path of `D:\DataFiles\file1` on an agent computer running Linux™, the substitution cannot be completed, because Linux™ uses forward slashes as path separator characters.
2. In the test editor, navigate to the request where you want to substitute content, and then select the request data that you want to substitute.
3. Right-click, and then select **Substitute > Select Data Source**.
4. Select the data source that contains the path to the file from which you want to substitute.
5. Right-click the substitution site, and then select **File Contents Substituter**.

Results

When the test runs, the content in the protocol request is substituted with the specified file contents.

HTTP POST data is displayed in the test editor in chunks. You can create a file contents substitution in the POST data of an HTTP POST request by selecting the data chunk that you want to correlate, and then clicking **Substitute**. The test data source that you select is automatically treated as a file contents substituter. The entire data chunk is replaced with the contents of the file when the test runs, even if only a portion of the text in a text data chunk is selected by the substituter.

Built-in Datasources

You can use built-in data sources instead of creating custom data sources to substitute the recorded values. The Built-in Datasources in the **Test Details** section displays the data sources that have been used and unused. You can also modify their properties from a location.

The built-in data sources are Current Date, Random Number, Sequential Number, and Timestamp. The values of these data sources are dynamically calculated and submitted to the test.

Creating a Random Number datasource

You can create a random number datasource in Rational® Performance Tester to assign unique random numbers to all the virtual users in a test. This datasource can generate unique integers or floating-point numbers for the virtual users in the `schedule`. You can choose to distribute the numbers of the virtual users or clients in a uniform, normal, or negative exponential way.

Before you begin

You must have created a test.

1. Open the test from the **Test Navigator** pane, and then select the name of the test in **Test Contents**.
2. Select **Built-in Datasources** from the **Category** drop-down list in **Test Details**.
3. Select **Random Number**, and then click **Add Built-in datasource**.

Result

The **Built-in-Datasource Selection Wizard** is displayed.

4. Enter a name for the datasource in the **Name** field.
5. Select one of the following options to apply the distribution method for the random numbers from the **Distribution** section:

Choose from:

- **Uniform:** Click this option to generate the random numbers with a uniform distribution. You can specify the minimum and maximum values for the generated numbers.

The default minimum and maximum values are 0 and 100.

- **Normal:** Click this option to generate the random numbers with a normal or Gaussian distribution. You can specify the average and the standard deviation for the generated numbers.

The default average and standard deviation values are 1.0.

- **Negative Exponential:** Click this option to generate the random numbers with an exponential distribution. You can specify the average for the generated numbers.


The default average value is 1.0.



6. Select one of the following options to apply the format for the random numbers from the **Formatting Options** section:

Choose from:

- **Common:** Select this option to format the random number in decimal, hexadecimal, octal, or scientific notations.

The **Common** option has the following fields to apply the format for the random numbers:

Fields	Description
Floating point	You can select this checkbox to generate the random number with a decimal point.
Precision	<p>You can enter a number in this field to show the smallest change in the random number.</p> <p>For example, if you entered 3 as the value in the Precision field, then the generated random number is in the following format:</p> <p>42.000</p> <p> Note: The Precision field is enabled only when you select the Floating point checkbox.</p>
Format as	<p>You can choose an option from the Format as drop-down list to apply the selected format for the random number.</p> <p>When you select the Floating point checkbox, then the Format as field lists the following options:</p> <ul style="list-style-type: none"> ▪ Decimal ▪ Scientific Notation ▪ Scientific Notation Uppercase <p>When you clear the Floating point field, then the Format as field lists the following options:</p> <ul style="list-style-type: none"> ▪ Decimal ▪ Hexadecimal ▪ Hexadecimal Uppercase ▪ Octal
Enable width control	You can select this checkbox to control the width of the generated random number.
Fixed width	You can enter a number in this field to have a specified number of digits in the random number.

Fields	Description
	 Note: The Fixed width field is enabled only when you select the Enable width control checkbox.
Zero-padded	<p>You can select this checkbox to fill the unused digits in the random number with zero.</p> <p>For example, if you selected the value as <i>Hexadecimal</i> for the Format as field, entered <i>4</i> as a value in the Fixed width field, and selected the Zero-padded checkbox, then the generated random number is displayed in the following format:</p> <p>004c</p>
Left-justified	<p>You can select this checkbox to align the random number from the left side.</p>  Note: The Zero-padded field is disabled when you select the Left-justified field. <p>For example, if you selected the value as <i>Hexadecimal</i> for the Format as field, entered <i>4</i> as a value in the Fixed width field, and selected the Left-justified checkbox, then the generated random number is displayed in the following format:</p> <p>4c</p>

- **Custom:** Select this option to specify the custom format for the random number.

The **Custom** option has the following field:

Format mask: You can enter a standard Java formatting syntax in this field to apply the format for the random number.



Note: You can preview the changes in the **Formatted output** field based on your selection of the fields.

7. Select the **Get new value each time used** checkbox to substitute the built-in datasource every time with a new random number for the requests.
8. Click **Finish**.

Results

You have created the random number datasource for the test.

What to do next

You can assign random numbers to virtual users in the `schedule` by using an environmental variable. See [Assigning random numbers to virtual users on page 325](#).

Assigning random numbers to virtual users

When a recorded value in the test contains variable data, and you want to substitute that value with a random number during the `schedule` run, you can assign random numbers by using a **Random Number** built-in datasource.

Before you begin

You must have performed the following tasks:

- Created a test and added a random number datasource to it. See [Creating a Random Number datasource on page 322](#).
- Created a schedule and added a User Group to it. See [Adding a user group to VU Schedule](#).
- Added a location for the User Group.

About this task

The default configuration uses the same seed for the generation of random number to provide predictable results from one `schedule` run to another. You can introduce variability from one `schedule` run to another by setting a new seed for each `schedule` run by using the `rptSeed` System Property.

You can use the System Property called `rptSeed` to initialize the random number seed for each virtual user. The default value of `rptSeed` is `1`. You can override the default value by adding the System Property `rptSeed` to the General Property called `RPT_VMARGS` in Rational® Performance Tester.

You can provide unique values for `rptSeed` for each `schedule` run only when you need variability in the random number generation. Otherwise, you can use the default value.

1. Expand the **Location** directory from the **Test Navigator** pane.
2. Double-click the location from the list.
3. Click the **General Properties** tab.
4. Click **Add**.
5. Enter `RPT_VMARGS` in the **Property Name** field.
6. Assign a value to `rptSeed` that you want to use in your `schedule` run in the **Property Value** field.

For example, `RPT_VMARGS=-DrptSeed=123`

The value `123` results in different random numbers for a given virtual user in the `schedule` as compared to results when using the default value of `1`. If you want different random numbers to be generated for your subsequent `schedule` runs, you must change the value of `rptSeed` before you start the next `schedule` run.

7. Click **OK**.

Results

You have assigned the desired value to the random number generator seeds.

What to do next

You can run the `schedule` and verify in the test log that you are getting the random number variability as desired.

Creating a Sequential Number datasource

You can create a sequential number datasource in Rational® Performance Tester to assign unique sequential numbers to all the virtual users in a test. This datasource can generate unique integers or floating point numbers for the users.

Before you begin

You can use the sequential built-in datasource option wherever data correlation substitutions are permitted, such as Transactions or Delays.

About this task

You define an initial value that should be assigned to the first virtual user and a step value that is a number by which the current value increments after each retrieval by a virtual user. If initial value is 1 and step value is 5, the workbench generates numbers in the sequence of 1, 5, 10, 15, and so on and each number is mapped to a virtual user. If a test is run on multiple agent machines, the workbench assigns a sequential value to all the users in all the agent machines.

You can also assign a full sequence of numbers of one virtual user. The sequence number increments in the request for each time the request in the multi-request generator is executed.

1. In the Test Contents area of the test, click an element in the test where data correlation substitution is permitted, such as a transaction name or delays.
2. In the Test Element Details area, right-click the name of the element and click **Substitutue > Built-in Datasources**.
3. In **Built-in Datasource Selection Wizard**, click **Sequential Number** and click **Next**.
4. Assign a name for the datasource.
5. In **Initial Value**, type a number to be assigned to the first virtual user.
6. In **Step Value**, type a number.
7. In **Formatting Options**, you can format the number in the manner you want to use.
8. **Optional:** To assign a full sequence of numbers to one virtual user, select the **Execute for individual user** check box.
9. **Optional:** To substitute the built-in datasource every time with a new value for the requests, select the **Get new value each time used** check box.
10. Click **Finish**.
11. Save and run the test.

Results

After you add the test to a schedule and run the schedule, the test log displays each element name where the datasource is applied with the unique sequential number assigned to the virtual user.

Correlating multiple fields in a test

Some tests are structured in such a way that you must correlate data for multiple fields. For example, assume that you plan to dataset an item that a virtual user is buying. For the test flow to be correct, you must also dataset all occurrences of that item in the test. You can find and correlate all instances of that item in one procedure. Typically, you use **Find More and Substitute** in the **Show Dataset Candidates** window to correlate data for multiple fields. See [Viewing dataset candidates when you open a test on page 299](#). Alternatively, you can use the **Test Search** page to correlate data for multiple fields.

To find all instances of a field in a test and correlate some or all of the instances with a data source, such as a dataset:

1. In the Test Navigator, browse to the test, and double-click the test. The test opens.
2. Locate the item or the substitution site to change or create a reference for. If the item is plain text, select the item. If the item is an existing reference, click the highlighted area.
3. Right-click, and then click **Find > More Substitution Sites**.
4. Click **OK**.
5. On the **Test Search** page, select **Case sensitive** to perform a case-sensitive search or **Regular expression** to perform a search using regular expressions. In regular expression mode, press Ctrl+spacebar key in **Search for text** for content assistance. Content assistance lists the regular expression patterns and the content that they match.
6. Click **More Options**, and then select the appropriate options:

Restrict to elements highlighted in Test Contents

Search only in elements that are selected in the **Test Contents** area.

Highlight found elements in Test Contents

Highlight found elements in the **Test Contents** area.

Recursive

Searches the child test elements in addition to the element. For example, if you search an HTTP page, select this option to search the requests and responses within the page.

Match encoded and decoded values (protocol-specific)

When selected, searches for matches of the unencoded and URL-encoded versions of the specified text. For example, when searching in HTTP data, `abc%123` and `abc%25123` match.

Include matches with overlapping data correlation

Include sites that are contained in, or overlap with, an existing substitution site. If you decide to substitute, the conflicting substitutions are automatically removed.

Include matching substituters

Click to return elements that originally matched the search string but have since been substituted. Clear to skip existing substitution sites when results are returned.


7. Click **Close**.
8. Click **Search**. The search results are displayed in the **Search** view.

9. In the **Search** view, select the matches to substitute, and then right-click the selection.
10. Optional: To select all matches, right-click the test name.
11. Click **Substitute in DataSource View**.

Result

This action sends the selected matches to the **Test Data Sources** window.

12. In the **Test Data Sources** window, click the tab that corresponds to the type of data source to use:

Option	Description
References	The data sources that have already been created in the test and the possible matches that are available as data sources for the selected substitution site. For example, text that is found in a response and used in a subsequent request is automatically created as a reference. Clear the Matching References Only check box to see all the references that occur before the substitution site in the test.
Datasets	The datasets that have been added to the test. To add a new dataset, click the Add new Dataset icon ().
Custom Code	If you have written test execution services (TES) custom code, the custom code that is available as data sources for this substitution site.
Test Variables	The test variables and their types that are available as data sources for this substitution site.
Built-in Datasources	The dynamically calculated data sources (Current Date , Random Number , Sequential Number , and Timestamp) that are available for this substitution site. To create a new built-in data source, right-click inside the Built-in Datasources page, and select New .

13. Select the data source, and click **Substitute From**.

Result

The **Substitute Multiple Items** window is displayed, showing information about the data source and substitutions sites that you selected.

14. For each site with a selected check box, click **Substitute Checked** to substitute the data source or clear the check box to skip the site.
Click **Always Prompt** to examine every substitution site one at a time. Click **Prompt on overlapping data correlations** to examine a site only if the site you are substituting into is contained in, or overlaps with, another substitution site. If you decide to substitute, the conflicting substitutions are automatically removed.

Results

The selected instances of the field are correlated with the data from the data source.

Guidelines for adjusting data correlation

When you run a test, you might notice that the server is not under the expected load or that your database is not being updated as expected. Incomplete or incorrect data correlation can cause these problems.

To identify data correlation problems:

1. Use the **Potential Correlation Errors** view to find missing or incorrect data correlations. See [Finding data correlation errors on page 331](#) for more information.
2. Run a test individually or in a schedule with the **Log Level** for errors, failures, and warnings set to **All**.
3. After the run, open the test log as explained in [Viewing the test logs on page 467](#).

The data correlation algorithms that are used during test generation are based on well known best practices. However, because these practices continually evolve, various types of errors can occur during automated data correlation:

- **Insufficient correlation:** Test values that must be correlated are not. Some possible causes follow:
 - Two parameters that must be correlated have different names.
 - A value must be correlated with a previous value that does not occur in the expected location.
 - A parameter or value must be correlated with a previous parameter or value that does not occur in the test because it is a computed value.
- **Superfluous correlation:** Unrelated test values are correlated.
- **Incorrect correlation:** Test values that must be correlated are correlated incorrectly.

Insufficient correlation: Parameters have different names or occur in unexpected locations

When two parameters that must be correlated have different names, automated data correlation does not recognize that the two parameters are related. For example, consider this request: `http://www.example.com?id=12345`. Suppose that this request must be correlated with the server response that contains `customer_ID=12345`, not `ID=12345`. In this case, the `ID` parameter must be correlated with `customer_ID`.

Data correlation typically links a response value that was returned from the server with a subsequent request value. The automated correlation algorithms search in the URL and the POST data for potential matches; however, other schemes for returning parameters are possible. For example, consider this request: `http://www.example.com?id=12345`. Suppose that this request must be correlated with the server response that contains the name and entity pair `href name="customer_ID" entity="12345"`, not `ID=12345`. In this case, the `ID` parameter must be correlated with `name="customer_ID"` and value `12345` must be correlated with `entity="12345"`.

Here are some additional causes of insufficient correlation:

- Siebel uses the star array format. Standard correlation algorithms can neither retrieve from this format nor substitute into this format.
- SOAP designates correlation parameters in external XML files. The correlation algorithms cannot correlate parameters in the external file with parameters in the test.

To manually correlate data in these cases:

1. In the test editor, use search or browse to locate the two parameters for correlation.
2. Navigate to the parameter that occurs later in the test, and select the parameter. This is the substitution site.
3. In the **Test Data Sources** window, click the **References** tab.
4. Select the data source to use as a reference, and then click **Substitute From**.

Insufficient correlation: One parameter is unnamed

Sometimes a parameter or value must be correlated with a previous parameter or value that is not named in the test, because it is computed, for example, by a JavaScript™ program. In this case, in order to correctly correlate the data, you must understand how and where the parameter or value is computed, and then use a custom code block. See [Extending test execution with custom code on page 402](#) for more information about custom code.

For example, consider the web address `http://www.example.com?login_stamp=12345_Apr_11_07`, where the value for `login_timestamp` is the concatenation of the login ID and the current date. In this case, you must generate a custom code that concatenates the login ID and the date.

For another example, suppose that the server returned the login ID and date as separate entities: `href "customer_id=12345" Date="Apr_11_07"`. In this case, you can put these parameters in separate references and, in subsequent requests that use customer ID and date, substitute them separately.

Superfluous correlation

Automated data correlation is based on pattern matching: A parameter or parameter value is correlated with a subsequent parameter or parameter value with an exact or similar name. But sometimes parameters with exact or similar names are in fact unrelated. In the best case, unneeded correlation is either harmless or adds a slight load that is inappropriate. In the worst case, the application does not expect a correlation and fails during playback.

To remove a superfluous data correlation:

1. In the test editor, search or browse to locate the substitution site that must not be correlated. By default, purple letters indicate correlated data.
2. Right-click the substitution site.
3. Click **Remove Substitution**.

Incorrect correlation

A parameter that requires data correlation might occur many times throughout a test. For example, a session ID parameter that is used initially when a user logs in might also be used in every subsequent request. If multiple instances of a parameter in a test are not same, the correlation algorithms might use the wrong instance.

With the HTTP Test Generation preferences, you can optimize automatic data correlation for accuracy or for efficiency.

- **Accuracy:** Each occurrence of a parameter is correlated with the nearest previous occurrence. This is the default setting.
- **Efficiency:** Each occurrence of a parameter is correlated with a single previous occurrence.



Note: If you do not manually apply a correlation in the Referer field in an HTTP request header, then the Referer field is automatically correlated as needed. If you manually apply a correlation in the Referer field in an HTTP request header, then no automatic correlation is performed.

Incorrect correlations are more likely to happen when **Optimize automatic data correlation for execution** is set to **Efficiency**. To fix an incorrect correlation:

1. In the test editor, search or browse to locate the value that is incorrectly correlated.
2. Right-click the substitution site.
3. Click **Remove Substitution**.
4. Right-click the substitution site again.
5. Click **Substitute**, and select the correct parameter.

Generally, the HTML response content after the recording would look like `<input type="username" name="User" id="aaa" value="John"/>`. Some applications dynamically update the name attribute. So, when you play back the test the HTML response content would look like `<input type="username" name="idt020" id="aaa" value="John"/>`. Because the name attribute is changing dynamically, data correlation would not occur and the playback would fail.

Such correlations are the result of the tool using the *name* attribute as the basis for correlating other attributes in the response code instead of the *ID*. To correlate the responses based on ID, select **ON** in **Window > Preferences > Test > Test Generation > HTTP Test Generation > Data correlation types > Prioritize correlation based on ID**.

Finding data correlation errors

You can use the **Potential Correlation Errors** view to find missing or incorrect data correlations.

Before you begin

Run a test or a single-user schedule. The **Potential Correlation Errors** view does not support multiple-user schedules. If verification points fail while you are running a test, you are prompted to open the **Potential Correlation Errors** view when the test run is complete.

To find data correlation errors:

1. In the **Test Navigator**, select the result of the test run where you want to find correlation errors.
2. Right-click the result, and then select **Find Data Correlation Errors**. You can choose **Missing Correlation**, **Incorrect Correlation**, or **All**.
3. The **Potential Correlation Errors** view opens.
After the test log is processed, the view is populated. Depending on the size of the test log, it can take significant time to populate the view. The potential missing or incorrect data correlations are displayed, in descending order of the likelihood that the correlation is incorrect. Selecting an item in the **Potential Correlation Errors** view automatically selects the corresponding element in the test editor, so that you can fix the potential error.
4. Use the **Compare with Test Log** toolbar button in the upper-right corner of the view to compare the request or response in the test with the same object in the test log.
5. For missing correlations, use the **Suggest Fix** toolbar button in the upper-right corner of the view to search for other instances of the value in all responses in the test. If a matching value occurs in an earlier response in the test, create a reference in that response.

Disabling data correlation

You can disable a data correlation source or a substitution site. When you disable a data source, none of the substitution sites that use the source will be correlated when you run tests. When you disable a substitution site, only that specific substitution site is disabled. Other substitution sites that use the same reference will be correlated when you run tests. You can also disable data correlation entirely for subsequent tests that you record.

To disable a data correlation source or substitution site:

1. In the **Test Navigator**, browse to the test and double-click it. The test displays in the test editor.
2. In the **Test Contents** area, click a request.
3. In the **Test Element Details** area, locate the data correlation source or substitution site.
4. Right-click the data value and select **Disable** from the menu.
To re-enable a disabled data source or substitution site, right-click the data value and select **Enable** from the menu.

Results

The data correlation source or substitution site is disabled.



Note: To disable data correlation for the entire workspace, click **Window > Preferences > Test Generation**, and clear **Enable automatic data correlation**. Subsequent tests that you record or regenerate will not include data correlation.

Recorrelating test data

If you disabled automatic data correlation before recording a test, you can regenerate the test with automatic data correlation enabled.

1. Click **Window > Preferences > Test > Test Generation**.

Result

The **Test Generation** preferences window opens.

2. Click the **Data Correlation** tab.
3. Select the types of data correlation to enable, and then click **OK**.
4. In the **Test Navigator**, browse to the test and double-click it. The test displays in the test editor.
5. Click **Edit > Re-correlate test data**.

Results

The test is regenerated with the types of automatic data correlation that you selected.

Simulating services with stubs

Service stubs enable you to simulate the behavior of an actual service for a wide variety testing or integration purposes.

Service stub overview

Service stubs are simulations of an actual service, which can be used to functionally replace the service in a test environment. A stub server replaces the actual application server.

From the point of view of the client application, the service stub looks identical to the actual service that it simulates. To use a service stub in replacement of the actual service, you must be able to replace the URL of the original service in the client application with the URL of the stub server.



Important: For version 8.7 and later, you cannot use the schedule option of IBM® Rational® Performance Tester to deploy stub servers remotely. If you have already deployed stub servers remotely, you must install IBM® Rational® Service Tester for SOA Quality or Rational® Performance Tester on those computers and then deploy the stub servers locally.

Use case examples

There are several cases where it can be useful to deploy a stub services instead of using the actual services for your tests:

- If you are testing a local service that uses data from another remote service, you might need to inject specific content to the service under test from the remote service. You can simulate the remote service with a service stub to ensure that the local service responds properly to some specific input.
- Some commercial services charge users for each call. If you are testing such a service, you can develop and debug your test against a stub service, which is based on the WSDL of the actual service, without being charged by the commercial service.
- During integration of a large application involving multiple clients and services, some services might not yet be operational, although their WSDL specifications are available. You can simulate the missing services with service stubs, which will allow you to proceed with the integration work.

Service stub architecture

You create a service stub by providing an existing WSDL specification. The service stub is generated with the exact same ports and bindings as the original service so that it can be addressed with exactly the same interface. Each operation in the service returns a default response of the type defined by the WSDL.

You can edit the service stub in the stub editor to change the default response or to create conditional responses that simulate the actual responses of the original service.

When you have finished editing the service stub, you can deploy it on a local stub server, which runs in the workbench. The stub server simulates an actual application server and can host multiple service stubs. You control the stub server from the stub monitor view.

Finally, to use the service stub instead of the original service, you change the URL used by the client application to point to the local stub server instead of the original application server. This URL, as well as the WSDL of the service stub, is provided in the stub monitor view.

Creating a service stub


You can use a WSDL (Web Service Description Language) specification file to generate a service stub that can simulate the behavior of the original service and uses the exact same interface.

Before you begin

Service stubs are stored in test projects. If your workspace does not contain a test project, the test creation wizard creates one, enabling you to change its name. To store a service stub in a specific project, verify that the project exists before you create the stub.

If you are using Secure Sockets Layer (SSL) authentication, ensure that you have any required key files in your workspace.

The wizard can import WSDL files from the workspace, the file system, a remote repository, or from a URL. Ensure that the WSDL files use the correct syntax for the test environment. Service stub generation might not work with some Web Services Description Language (WSDL) files.

1. In the workbench, click **File > New > Other > Test > Test Assets > Service Test** or click the **New Service Stub**  toolbar button.
2. Select the WSDL of the service that you want to simulate. If necessary, you can import the WSDL from the file system, a URL, or a WSRR or UDDI repository.
3. Click **Next**.
4. Select a project location and a name for the new service stub. Click **Finish**.

Results

The wizard generates a working service stub that reproduces the interface of the original service as defined in the WSDL specification. Each operation is reproduced with a default response. You can edit the service stub with the stub editor to change the default response or to create conditional responses.

Editing a service stub

Service stubs are generated with a single default response for each operation in the WSDL specification. You can edit the service stub to change the default responses or to add conditional responses that can simulate the actual service.

To edit the behavior of a service stub:

1. In the test navigator, double-click the stub to open the stub editor.

Each operation simulated by the stub is represented by an operation element, which contains **Case** elements that describe a condition. Each case contains a response element. Case elements are similar to test verification points and use the same presentation.
2. To change the default response of an operation:
 - a. Expand the operation and the **Case : Default** element, and then select the response element.

The Case : Default element describes the response of the service stub when no other case condition is met.
 - b. Edit the **Message** content to specify the XML content returned by the service stub.
3. To add a conditional response case:
 - a. Right-click the operation and select **Add > Equals Case, Contains Case, or Query Case**.

These conditional case types are similar to the *Equals*, *Contain* and *Query* verification points in service tests.

 - Use **Equal Case** to specify a response that is returned by the stub when the entire incoming message content fully matches the specified message content.
 - Use **Contains Case** to specify a response that is returned by the service stub when a portion of the incoming message content matches the specified message content.
 - Use **Query Case** to specify a response that is returned by the service stub when an XPath query meets the specified criteria.

You can add as many case elements as necessary to simulate the behavior of the original service. Use the **Up** and **Down** buttons to change the order in which the case conditions are evaluated. Only the first matching condition is executed.

The default case cannot be removed and is always the last case element in the operation.
 - b. Select the response element and edit the **Message** content to specify the XML content returned by the service stub. Use the **Form**, **Tree**, and **Source** views to change the XML content display mode.
4. Select **File > Save** or click the **Save** toolbar button.

What to do next

When you have finished editing the service stub, you can deploy the stub to a stub server.

Deploying service stubs

You deploy and run service stubs on a stub server, which is a small application server dedicated to running service stubs. The client application, or test, addresses the stub server instead of the actual application of the original service.

Before you begin

The local stub server runs in the workbench on the local computer. Service stubs can be accessed locally. The local stub server is automatically stopped when you close the workbench.

To use a service stub instead of the original service, you must be able to change the endpoint of the client application or service test to replace the URL of the original application with the URL of the stub server.

1. In the stub editor, click the **Deploy** button.

Alternatively, you can right-click the stub in the test navigator and select **Deploy On > Local stub server**

Result

This opens the **Stub Monitor** view.

2. In the **Stub Monitor** view, click  **Run**.

If you make any changes to the service stub, the stub is redeployed to the stub server after saving.

3. To add more service stubs to the stub server, click **Add** and select a service stub from the workspace.
4. Copy the URL of the service stub from the **Stub Monitor** view and paste it into the configuration of the client application.

You can also directly access the WSDL specification of the service stub, which is a copy of the original WSDL with replaced URL endpoints.

What to do next

You can validate that the service stub is responding correctly by using the generic service client to invoke a call.

Recording service stub activity in a log file

With service stub logging, you can monitor the interactions between an application and the stub server. When the option is enabled, one log file is created for each deployed stub. The log files are presented as a formatted HTML report.

Before you begin

You must have created one or several service stubs.

To log service stub activity:

1. Add the following virtual machine (VM) argument to the `eclipse.ini` file: `-DSTUB_LOG_LEVEL=log_level`.

Use one of the following values for the `log_level` variable:

- 0: Disable the log.
- 1: Log stub activity without details.
- 2: Log stub activity including content of sent and received messages.

- 3: Same as level 2 with HTTP headers of received messages.
- 4: Same as level 3 with attached files.

You can also add the following optional arguments:


- `-DSTUB_LOG_KEEP_PREVIOUS=true`: This option creates a separate log file each time the service stub is redeployed. If the value is not `true` or if the option is not present, the log file is erased if the service stub is redeployed or when the stub server is stopped.
- `-DSTUB_LOG_SERIALIZE_XML=true`: This option displays the XML content (with log levels 2, 3, and 4) without formatting or indentation. If the value is not `true` or if the option is not present, the XML content is formatted and indented in the log.

The `eclipse.ini` file is located in the same directory as the `eclipse.exe` launcher binary file that is used to run the product.

Example

For example, to enable logging with basic content, add the following line to the end of the `eclipse.ini` file:

```
-DSTUB_LOG_LEVEL=2.
```

2. Restart the workbench, and in the **Stub Monitor** window, click the **Run** icon  to restart the stub servers.
3. If the server was launched by a schedule in the performance testing application, then corresponding logs are automatically created in the workspace. If not, complete the following steps to retrieve the log files from the stub server:



Important: The stub server must be running.

- a. After running your tests, to view the service stub log files, open the **Stub Monitor**, and click the tab for the stub server.
- b. Click the **Synchronize** toolbar button for the selected server.

Result

An HTML log file is created and displayed for each deployed service stub.

Result

The stub log reports are located in a folder named `stubLogs`, which is in the same folder as the corresponding service stub.

Setting log level for service stubs


While recording a service test, you can set the level of the log details that you want to collect for debugging purposes.

Before you begin

You must stop the stub server.

About this task

The log level that you set in this way takes precedence over the log level setting that you specify in the `eclipse.ini` on [page 336](#) file.

1. In the **Stub Monitor** view, in the Service Stubs section, click the **Edit log options** icon .
2. Select one of the log level options and click **OK**.

What to do next

Start the server again for the changes to take effect.

Sending service requests with the generic service client

The generic service client enables you to send requests to services for which you do not have a convenient client and to view the responses returned by the service.

Creating transport protocol configurations

Read these topics to configure various transport protocols.


Creating an HTTP transport configuration

You can create an HTTP transport configuration that describes the transport settings for a service request. Transport and security settings can be associated with any service request.

Before you begin


If you are using Secure Sockets Layer (SSL) authentication, ensure that you have valid key files in your workspace.

If you are using SOAP security, ensure that you have configured the environment with the correct libraries and configuration files.

1. Click the **Generic service client**  toolbar button to open the generic service client and click the **Transport** tab.

Result

This opens the **Transport Configurations** page.

2. On the **Transport Configurations** page, click **Create an HTTP configuration**  to create a new HTTP transport configuration.
3. Type a **Name** for the new transport configuration.
4. Specify the following options for the HTTP transport:

HTTP/2



Note: Testing HTTP/2 service is in the Beta mode. For more information, see [Preparing to record a HTTP/2 service on page 215](#).

To test a service that uses the HTTP/2 protocol, select the **Activate** check box. This check box is automatically selected when you record a service by using a browser. If you use the Generic Service Client component to create a HTTP/2 test, you have to manually select the check box.

HTTP/2 client connection timeout

Specifies the time limit for the HTTP/2 client to connect to the HTTP/2 server.

Time out for the HTTP/2 session creations

Specifies the time limit to create the HTTP/2 session. This time starts after the connection is established.

Enable HTTP/2 Push

The Push functionality of HTTP/2 automatically identifies and passes the related objects or requests to the client when a request is sent to the server. Clear the check box to not use the functionality.

Initial session window

Specifies the buffer size on the sessions.

Initial stream window

Specifies the window size for buffer on each stream after the connection is established.

HTTP/2 Client Input Buffer Size

Specifies the buffer size that is used to read the network traffic.

Maximum Quantity of Messages that can be queued

Specifies the maximum number of messages that can be queued for the HTTP/2 client on a thread.

Maximum Quantity of HTTP/2 thread pool

Specifies the maximum number of thread pools that will be used by the HTTP/2 client to distribute the workload.

Minimum Quantity of HTTP/2 thread pool

Specifies the minimum number of thread pools that will be used by the HTTP/2 client to distribute the workload.

HTTP/2 client bytearray pool size

Specifies the buffer size to receive the unciphred values.

Server Name Indication

Note: Not applicable for HTTP/2.

Clear this check box if you do not want to connect to the host computer by using the Server Name Indication protocol. If the host computer is already configured with Server Name Indication protocol, you should keep this check box selected.

Use HTTP Keep Alive

Select this option to keep the HTTP connection open after the request. This option is not available if you are using IBM® Rational® AppScan®.

Use SSL

Select this option to use an SSL configuration. Click **Configure SSL** to create an SSL configuration or select an existing configuration.

Platform Authentication

In this section, specify the type of authentication that is required to access the service. Select **None** if no authentication is required.

Basic HTTP authentication

Select this option to specify the **User Name** and **Password** that are used for basic authentication.

NTLM authentication



Note: Not applicable for HTTP/2.

Select this option to use the Microsoft™ NT LAN Manager (NTLM) authentication protocol. NTLM uses challenge-response authentication. This view lists what is negotiated (checkmark_perf.jpg by the client and requested of the server) and what is authenticated (the client reply to the challenge from the server).

Kerberos authentication



Note: Not applicable for HTTP/2.

Select this option to use the Kerberos authentication protocol between the client and server.

Connect through proxy server



Note: Not applicable for HTTP/2.

If the HTTP connection needs to go through a proxy server or a corporate firewall, specify the **Address** and **Port** of the proxy server. If the proxy requires authentication, select either **Basic proxy authentication** or **NTLM proxy authentication**.

Proxy authentication

In this section, specify the type of authentication that is required to access the proxy. Select **None** if no authentication is required.

Basic proxy authentication

Select this option to specify the **User Name** and **Password** that are used for basic authentication.

NTLM proxy authentication

Select this option to use the Microsoft™ NT LAN Manager (NTLM) authentication protocol. NTLM uses challenge-response authentication. This view lists what is negotiated (checkmark_perf.jpg by the client and requested of the server) and what is authenticated (the client reply to the challenge from the server).

Custom class

Note: Not applicable for HTTP/2.

Select this option if the communication protocol requires complex, low-level processing with a custom Java™ code to transform incoming or outgoing messages. Click **Browse** to select a Java™ class that uses the corresponding API. This option is not available in IBM® Security AppScan®.

See [Creating SSL configurations on page 348](#) for more information about SSL authentication.

5. Click **OK** to create the new configuration.

What to do next

Once created, you can use your new configuration with any service request that uses the HTTP transport protocol. You can use the **Configurations** list in the generic service client to edit existing configurations or to create duplicate configurations.

Configuring the workbench for NTLMv2 authentication

NTLMv2 authentication requires access to a third-party library. To record and execute a test that contains NTLMv2 authentication, you must download the library and place it at the right location.

Before you begin

Before you can test SOAP-based services that use security algorithms, you must obtain and install a third-party library file.

About this task

By default, the HTTP test generation does not enable NTLMv2 authentication, even if it was part of the recording. To automatically enable the correct NTLM version from the recording, set the **Generated NTLM Version** setting to **Guess from recorded data** in the HTTP Test Generation preferences.

To configure the workbench to enable NTLMv2 authentication

1. Download the `jcifs-1.3.19.zip` file from <https://www.jcifs.org/src/>.
2. Unarchive the zip file and copy the JAR file to the installation directory: `InstallationDirectory\plugins\com.ibm.rational.test.lt.provider_<version>`
3. To automatically enable the correct NTLM version from the recording, In the workbench, click **Window > Preferences > Test > HTTP Test Generation** and set the **Generated NTLM Version** setting to **Guess from recorded data**.

Results


When a test was recorded with NTLMv2, the **Generated NTLM Version** setting is selected in the test editor, under **NTLM Authentication**.

Creating a JMS transport configuration

You can create an JMS transport configuration that describes the transport settings for a service request that uses the Java™ Message Service (JMS) protocol, including JBoss and IBM® WebSphere® JMS. Transport and security settings can be associated with any service request.

Before you begin

If you are using SOAP security, ensure that you have configured the environment with the correct libraries and configuration files.




1. Click the **Generic service client**  toolbar button to open the generic service client and click the **Transport** tab.

Result

This opens the **Transport Configurations** page.

2. On the **Transport Configurations** page, click one of the following buttons:

Choose from:

- **Create a basic JMS configuration**  to create a new generic JMS transport configuration.
 - **Create a JBoss JMS configuration**  to create a JMS configuration preconfigured for JBoss.
 - **Create a WebSphere JMS configuration**  to create a JMS configuration preconfigured for WebSphere® JMS.
3. Type a **Name** for the new transport configuration and select whether the service is a **queue** or a **topic** destination.
 4. Type the address of the JMS end point.
 5. Select **Use temporary object** to provide the address of the reception point to the service as a temporary object. If you disable this setting, you must manually specify the reception point address.
 6. If the service requires authentication, select **Basic Authentication** and type the user name and password to access the service.
 7. If the service requires a custom Java™ Naming and Directory Interface (JNDI) adapter, you can provide your own Java™ class that extends the Apache Axis class. In this case, select Custom Adapter and specify the name of the custom Java™ class. See [Extending test execution with custom code on page 402](#) for more information about custom code.

8. Specify whether the message type is **Text** or **Binary**.
9. If necessary, click **Add** or **Edit** to specify the **Context factory properties** or **Connector properties** required to access the service.
10. Click **OK** to create the new configuration.

What to do next

Once created, you can use your new configuration with any service request that uses the JMS transport protocol. You can use the **Configurations** list in the generic service client to edit existing configurations or to create duplicate configurations.

Creating a WebSphere® MQ protocol configuration

When you want to send requests to a service that uses WebSphere MQ transport protocol, you can create a protocol configuration to describe the transport settings for a service request.


Before you begin

If you are using SOAP security, ensure that you have configured the environment with the correct libraries and configuration files.

About this task

By default, messages are sent in bytes. Starting from 10.1.0, you can select message type as Text Message. After you create the protocol configuration, you can change the message format by selecting the **Text Message** check box in the **Message Structure**.

Transport and security settings can be associated with any service request. You can edit the existing configuration or duplicate the default configuration. You must have configured the environment with the correct libraries and configuration files when you use SOAP security.

1. Click the **Generic service client**  toolbar button, and then click the **Transport** tab.
2. From the **Protocol** list, right-click **MQ**, and then click **New MQ protocol configuration**.
3. Enter a name for the new transport configuration in the **Name** field.
4. Enter a name for the queue manager that receives the call in the **Queue Manager Name** field.
5. Enter a name for the queue managed by the queue manager in the **Send Queue Name** field.
6. Select the **Authentication** check box and specify the user name and password to authenticate with the MQ server.
Alternatively, add or update the login credentials in the **Protocol Configuration** tab of a service test.
7. Select the **Use Local Queue Manager** check box when the WebSphere MQ server is running on the local computer.
8. Perform the following steps if the MQ server is installed on a remote computer:
 - a. Clear the **Use Local Queue Manager** check box.
 - b. Enter the remote WebSphere MQ server details in the following fields:

- The IP address or host name in the **Address** field.
 - Listener port number in the **Port** field.
 - Server connection mode channel name in the **Client Channel** field.
9. Select the **Use Temporary Queue for Response** check box if you want the server to create a temporary queue for receiving messages.
 10. Perform the following steps to specify the queue that receives the response messages from the queue manager:
 - a. Clear the **Use Temporary Queue for Response** check box.
 - b. Enter a name for a queue in the **Receive Queue Name** field.
 11. **Optional:** Specify the name of the target service in the **Target service** field when you are using the Microsoft .NET framework with SOAP over MQ.
 12. **Optional:** Select **Use RFH2 header** when you are using SOAP over MQ. Otherwise, specify the **Message Descriptor** and **Encoding** options for the message header.
 13. **Optional:** Click **Configure SSL** to select an existing SSL configuration or to create a new one when the service requires SSL authentication.
 14. Click **OK** to create the protocol configuration.

Results

You have created a configuration for the WebSphere MQ transport protocol.

What to do next

- You can use the protocol configuration for the WebSphere MQ with any service request.
- You can change the message format by selecting the **Text Message** check box in the Message Structure.

Related information


[Creating SSL configurations on page 348](#)

Creating a WebSphere® Java MQ transport configuration

You can create a transport configuration that describes the transport settings for a service request that uses the IBM® WebSphere® Java MQ protocol. Transport and security settings can be associated with any service request.

About this task


This topic has instructions to specify the MQ server settings. If you have a single MQ server, you can choose to use the **Default Java MQ protocol configuration** option. If, for a new request, you must point to another MQ server, you can use the instructions in this topic to create a new transport configuration.

1. Click **Generic service client**  and click the **Transport** tab.
2. To create a new Java MQ transport configuration, in **Configurations**, select **Java MQ**.
3. In **Create Java MQ protocol configuration**, specify a name for the transport configuration.

4. Complete the following steps in the **Settings** tab:
 - a. **Host:** Specify the host name or IP address of the MQ server.
 - b. **Port:** Specify the port number that is used on the MQ server.
 - c. **Channel:** Name of the MQ communication channel that is used for sending and receiving messages and specified on the server. This field is case-sensitive.
 - d. **Queue Manager:** Name of the MQ queue manager as specified on the server.
 - e. **Optional: Use credential:** To access the secure server, specify the login credentials that is needed by the connection.
5. **Optional:** If necessary, complete the following steps in the **SSL** tab:



Learn more about the UI elements in the SSL tab:

- Select the **Use MQ SSL** check box when the connection to the Queue manager uses SSL.
 - **Peer Name:** Distinguished Name (DN) of the queue manager to be used by SSL. The Distinguished Name is available in the SSL certificate. In MQ, a DN pattern is specified by using the **sslPeerName** variable of **MQEnvironment**. Connections succeed only if Peer Name matches the pattern that is specified.
 - **Cipher Suites:** Select one of the available cipher suites to use for encrypting the transport communications.
 - **Fips Required:** This option specifies whether the requested cipher suite must use FIPS-certified cryptography in WebSphere MQ.
 - **KeyResetCount:** The total number of non-encrypted bytes that can be sent and received within an SSL conversation before the secret key is renegotiated. If left blank or set to zero (default), the secret key is never renegotiated. This value is ignored if no cipher suite is specified. Valid values are integers 0 - 999,999,999.
 - **SSL Configuration:** Select a SSL setting for the connection or click **Configure SSL** to create a new SSL configuration. See [Creating SSL configurations on page 348](#).
6. **Optional:** Use the **Options** tab to configure actions such as read, write, and browse on the selected MQ Queues. Click  to select the configuration options.
 7. **Optional:** Use the **Advanced** tab to specify the number of queue manager connections for reading messages, temporary destination settings, and to associate a reply with a request.
 8. To test the connection, click **Test Transport** and then click **OK**.

Result

You have created a new transport configuration to point to a MQ server.

What to do next

You can now send the Java MQ requests to the configured server. See [Sending WebSphere Java MQ endpoint requests on page 357](#).

Creating Microsoft™ .NET transport configurations


You can manually create a Microsoft™ .NET transport configuration to describe the transport settings for service requests that use the Windows™ Communication Foundation (WCF) protocol.

Before you begin

If you are using SOAP security, ensure that the environment is configured with the correct libraries and configuration files.

Certificates and libraries required by the Microsoft™ client proxy must be installed on the computer, including Microsoft™ .NET libraries.


You must link a modified version of the Microsoft™ client proxy configuration file of the WCF service (by default `client.exe.config`) to the Microsoft™ .NET transport configuration. You must rename the file to `soaclient.exe.config` and edit it as described in the following procedure.

 **Tip:** You can create a Microsoft™ .NET transport configuration automatically by importing the Microsoft™ .NET WSDL file. In this case, you must still manually edit the Microsoft™ .NET transport configuration to point to the modified `soaclient.exe.config` file as described in the following procedure. For more information, see [Sending service requests with WSDL files on page 350](#)


About this task

The product supports testing WCF services that use the following bindings:

- BasicHttpBinding
- WsHttpBinding
- NetMsMqBinding for 1-way calls only
- WSFederationHttpBinding
- WS2007FederationHttpBinding
- NetTcpBinding
- Custom bindings that do not integrate custom extensions in the channel, serialization of the message, transport, and security

 **Note:** The following WCF services are not checkmark_perf.jpg:

- Transaction and scopes
- Duplex mode requests, such as callbacks or 2-way services based on the Microsoft™ Message Queuing (MS-MQ) transport

 **Only for IBM AppScan users:** To use Generic Service Client with IBM Appscan to test a WCF application, add the following code to the WCF configuration file:



```
<system.diagnostics> <trace autoflush="true" />
  <sources> <source name="System.Net"
    maxdatasize="1048576"><listeners><add
name="System.Net"/></listeners></source> <source
  name="System.Net.Cache"><listeners><add
name="System.Net"/></listeners></source> <source
name="System.Net.Http"><listeners><add name="System.Net
"/></listeners></source> <source
name="System.Net.Sockets"><listeners><add
name="System.Net"/></listeners></source> <source
name="System.Net.WebSockets"><listeners><add
name="System.Net"/></listeners></source> </sources>
  <sharedListeners> <add
name="System.Net"
  type="IBM.ServiceModel.Soa.Extension.tools.TrafficTraceListener,
  Soa-Behavior-Library"
  initializeData="" />
</sharedListeners> <switches> <add name="System.Net"
  value="All"/> <add name="System.Net.Cache"
value="All"/> <add name="System.Net.Http"
value="All"/> <add name="System.Net.Sockets"
value="All"/> <add name="System.Net.WebSockets"
value="All"/> </switches></system.diagnostics>
```

IBM Appscan expects only HTTP requests in WCF. The following HTTP bindings are checkmark_perf.jpg:

- BasicHttpBinding
- Custombinding above standard httpTransport
- WsHttpBinding
- WsFederationHttpBinding
- WS2007FederationHttpBinding

Also, the following patterns are checkmark_perf.jpg:

- Action value (mandatory)
- Reply Action value (mandatory)
- Protection level

1. Create a modified `soaclient.exe.config` file by completing the following steps:

- a. Create a copy of `client.exe.config` (or `proxy_client_name.config`) file from the Microsoft™ .NET project and rename the copy to `soaclient.exe.config`.
- b. Edit the `soaclient.exe.config` file to use the version of Microsoft™ .NET that the product supports, as specified on the following line:

```
<supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.0"/>
```


- c. Edit the `soaclient.exe.config` file so that the endpoints in the configuration file point to the client contract of the product, as specified on the following line:

```
contract="IBM.ServiceModel.Soa.Extension.Stub.IStubTest"
```

d. Import the modified `soaclient.exe.config` file into the workspace.

Result

After you create the `soaclient.exe.config` file, you can skip the following steps and import the WSDL file to automatically create a Microsoft™ .NET transport configuration based on the information provided by the WSDL. For more information, see [Sending service requests with WSDL files on page 350](#).

2. Click the **Generic service client** toolbar button () to open the generic service client and click the **Transport** tab.
3. On the **Transport Configurations** page, click **Create a Microsoft .NET configuration**.
4. Type a name for the new transport configuration and specify the following options:

Location of `soaclient.exe.config`

Specify the location of the `soaclient.exe.config` file. You must create this file manually by copying and editing the `client.exe.config` file from the Microsoft™ .NET service.

User authentication

If the service requires authentication, select **User Authentication** and type the user name and password to access the service.

Endpoint protection

By default, the transport configuration uses the endpoint protection level that is described in the `soaclient.exe.config` file. Use this setting to specify a different **Protection level**:

- **Signature**: Select this option to digitally sign requests.
- **Encryption and Signature**: Select this option to digitally sign and encrypt requests.

Advanced properties

Use this table to list the request and response actions by order of the methods in the WSDL file.

Click **Add** to specify the name and value of request and response actions that are required by the service. This table is generated automatically when you import a Microsoft™ .NET WSDL file.

5. Click **OK** to create the transport configuration.

What to do next

After you create the configuration, you can use it with any service call that uses the Microsoft™ .NET transport protocol. You can use the **Configurations** list in the generic service client to edit existing configurations or to create duplicate configurations.

Creating SSL configurations

You can create a Secure Sockets Layer (SSL) configuration that describes the settings for a service request that uses SSL certification mechanisms. SSL configurations can be associated with any service request that uses the HTTP or IBM® WebSphere® MQ transport protocols.


Before you begin

If you are using SSL, ensure that you have valid certificate keystore files in your workspace.

If you are using SOAP security, ensure that you have configured the environment with the correct libraries and configuration files. See [Configuring the environment for SOAP security on page 208](#) for more information.


About this task

If you have to use different mutual SSL authentications for virtual testers in a test, you can create a dataset that stores all of the trust aliases names. In the test editor, in the **SSL Configuration** tab, you add a SSL configuration and associate it with the dataset. When a schedule is run, the SSL configuration is applied to each virtual tester.

1. Click the **Generic service client**  toolbar push button to open the generic service client, and click the **Transport** tab.
2. Either open an existing HTTP or WebSphere® MQ transport configuration, or create a new one, and then click **Configure SSL**.

3. Click **Rename**



to rename the default SSL configuration or **New**  to create one.

4. Specify the following settings for the SSL configuration.

Server Authentication

This section describes how the client trusts the server.

Always trust server

Select this option if no authentication is required or to ignore server certificates so that all servers are trusted. If you are using single authentication and you want to accept trusted servers only, then disable this option and specify a truststore that contains the trusted server certificates.

Client truststore

When you are using single authentication, the client truststore contains the certificates of all trusted servers. Click **Browse** to specify a KS, JKS, or JCEKS file containing valid certificates of the trusted servers.

Password

If the client truststore file is encrypted, type the password required to access the file.

Mutual Authentication

This section describes how the server trusts the client in addition to server authentication.

Use client-side certificate

If you are using double authentication, select this option to specify a keystore containing the client certificate. This certificate allows the server to authenticate the client.

Client certificate keystore

Click **Browse** to specify a KS, JKS, or JCEKS file containing a valid certificate that authenticates the client.



Password

If the client truststore file is encrypted, type the password required to access the file.

Select trust alias for Mutual Authentication

Select an alias to be used for the SSL configuration. There could be multiple aliases in a keystore for different security certificates. Choose an appropriate alias for a user. You can also use dataset to store aliases that you can apply to virtual users at run time.



Note: You can copy the contents from an SSL configuration into another SSL configuration by using **Copy**  and **Paste**  in the SSL editor.

5. Click **OK** to create the configuration, and close the SSL editor.

What to do next

When the SSL configuration is created, you can use the SSL configuration with any service request that uses SSL certification. You can use the SSL editor to edit existing configurations.

Sending service requests with WSDL files

You can send requests to services based on SOAP, Java Messaging Service (JMS), WebSphere® MQ, and Microsoft™ .NET that use a Web Service Description Language (WSDL) file to specify the contents of the service request.

Before you begin

Ensure that you have a valid WSDL file, which is accessible either on the file system, in the workspace, at a specific URL, or in an IBM® WebSphere® Service Registry and Repository or a Universal Description Discovery and Integration (UDDI) repository.

Ensure that the WSDL files use the correct syntax for the test environment. The generic service client might not work with some WSDL files.

If the service uses Secure Sockets Layer (SSL) authentication, create an SSL configuration before sending the request. For more information, see [Creating SSL configurations on page 348](#).

If the service uses SOAP security for encryption, signature, or other security algorithms, you must first configure the environment with the correct libraries and configuration files, and then create a WSDL security profile. For more information, see [Configuring the environment for SOAP security on page 208](#) and [Creating security profiles for WSDL files on page 257](#).

To import a WSDL file from a secured site that requires mutual authentication, you must have the Keystore file in the workspace.



About this task

When you create a call from a WSDL file, the call is configured automatically with any SOAP, JMS, WebSphere® MQ, or Microsoft™ .NET endpoints that are available in the WSDL file. Select the corresponding transport configuration on the **Transport** page of the request.






Note: For the specific requirements related to Microsoft™ .NET support, see [Creating Microsoft .NET transport configurations on page 346](#).

To send a service request based on a WSDL file:

1. Click the **Open the Generic Service Client** toolbar button  and select the **Requests** page.
2. Click **Add**  and select the method to add a WSDL file or click the corresponding shortcut button on the main page.

Choose from:

- Click **Add WSDL from Workspace** to add a WSDL file from the local workspace.
- Click **Add WSDL from File System** to add a WSDL file from the file system.
- Click **Add WSDL from URL** to download and import an online WSDL from the web.
- Click **Add WSDL from WSRR** to add a WSDL from WebSphere® Service Registry and Repository. Enter the URL of the WebSphere® Service Registry and Repository and click **Connect**. You can click **Search**  to browse the contents of the repository.
- Click **Add WSDL from UDDI** to add a WSDL from a Universal Description Discovery and Integration (UDDI) repository. Enter the URL of the UDDI and click **Connect**. You can click **Filter**  and **Search**  to browse the contents of the repository.



Note: If you are importing the WSDL file from a secured site that requires certificate authentication, click **Import Properties** and, for **Keystore**, select the keystore file that contains the certificate to be provided to the server, and for the **Keystore password**, type the password.

3. Click **OK**.

Result

The WSDL file is added to the **Request Library**.

4. In the **Request Library**, expand the WSDL file, binding, and operation, and then select the call element.

Result

The generic service client shows three steps: **Edit Data**, **Invoke** and **View Response**. The details for the call are displayed under the **Edit Data** step.

5. On the **Message** page, use the Form, Tree, or Source views to edit the contents of the request.

Each view shows a different format of the same data. To add or remove XML elements in the Form or Tree view, click **Schema > Validate and Assist** to comply with an XML Schema Definition (XSD) specified in the schema catalog.

6. On the **Transport** page, specify the transport configuration for the request.

The transport information from the WSDL file is imported automatically into the transport configuration.

For Microsoft™ .NET, select the corresponding transport configuration and specify the location of the `soaclient.exe.config` file. You must create this file manually. For details, see [Creating Microsoft .NET transport configurations on page 346](#).



Note: If you are using IBM® Security AppScan®, only the HTTP and .Net transport protocols are available.

7. On the **Request Stack** page, specify whether to override the security or processing algorithms that are applied to the outgoing request for the WSDL file.

Click **Show Response Stack** to add a **Response Stack** page to edit the security or processing algorithms for incoming responses.




Note: These settings apply only to the current request. If you want to edit the request or response stack for all requests that use the current WSDL file, click **Edit WSDL Security** to open the **WSDL Security Editor**.

8. When you are ready to send the service request, click **Invoke**.

Result

The generic service client sends the request and displays the message return under the **View Response** step.

What to do next

Successful requests are recorded and added to the **Request History** list. If you are using IBM® Rational® Performance Tester or IBM® Rational® Service Tester for SOA Quality, you can create a service test by clicking the **Generate Test Suite** button (.

Sending HTTP endpoint requests



You can send requests to services that use an HTTP endpoint.

Before you begin

If the service uses Secure Sockets Layer (SSL) authentication, create an SSL configuration before sending the request. For more information, see [Creating SSL configurations on page 348](#).

If the service uses SOAP security for encryption, signature, or other security algorithms, you must first configure the environment with the correct libraries and configuration files, and then create a security profile for the WSDL file. For more information, see [Configuring the environment for SOAP security on page 208](#) and [Creating security profiles for WSDL files on page 257](#)

To send a request to an HTTP service:

1. Click the **Open the Generic Service Client** toolbar button  and select the **Requests** page.
2. Click the **Add** icon  and click a type of request that you want to send or in Request Library, right-click **EndPoints** and select a type of request that you want to send.
3. In the **Configure Protocol** window, select **HTTP** and specify the HTTP transport configuration.
If necessary, click **New** to create an HTTP transport configuration for the call.

To send the HTTP/2 requests, in the **Create HTTP Protocol configuration** window, click the **Activate** check box. Before capturing the HTTP/2 traffic, configure the computer. See [Preparing to record a test for the HTTP/2 service on page 215](#) for instructions.

4. Type the URL of the call, the HTTP method and version, and specify any header or cookie properties.
Click the **Rest mode** check box to split the URL into resource and parameters.
5. Click **Next**.
6. On the **Select Root Element** page, if the service uses a specific XML Schema Definition (XSD), select one from the list or click **Browse** to import the XSD file, and then, select the root element for the request.
If no XSD is available for the service, select **No Schema**.
7. Click **Finish**.

Result

The request is added to the **Endpoints** section of the **Request Library**.

8. In the **Request Library**, select the request element.

Result

The generic service client shows three steps: **Edit Request**, **Invoke**, and **View Response**. The details for the request are displayed under the **Edit Request** step.

9. Based on the request selected in Step 2, on the **Message** page, use the **Form**, **Tree**, or **Source** views to edit the contents of the request.

Each view shows a different format of the same data. To add or remove XML elements in the **Form** or **Tree** view, click **Schema > Validate and Assist** to comply with an XSD specified in the schema catalog.

10. On the **Attachments** page, specify any file attachments to send with the request.
To add an attachment, click **Add** and follow the wizard to attach a file with the request.
11. On the **Transport** page, if necessary, change the transport configuration to be used by the request.
To create and edit transport and security configurations, use the **Transport** tab.
12. If you selected SOAP XML request in step 2, on the **Request Stack** page, specify whether you want to override the security or processing algorithms that are applied to the outgoing request for the WSDL file.
To add a **Response Stack** page to edit the security or processing algorithms for incoming responses, click **Show Response Stack**.




Note: These settings apply only to the current request. To edit the request or response stack for all requests that use the current WSDL file, click **Edit WSDL Security** to open the **WSDL Security Editor**.

13. When you are ready, click **Invoke** to send the service request.

Result

The generic service client sends the request and displays the message return under the **View Response** step.

What to do next

Successful requests are recorded and added to the **Request History** list. If you are using IBM® Rational® Performance Tester or IBM® Rational® Service Tester for SOA Quality, you can create a service test by clicking the **Generate Test Suite** button ()



Sending a JMS endpoint request

You can send requests to services that use a Java™ Messaging Service (JMS) endpoint.

Before you begin

If the service uses Secure Sockets Layer (SSL) authentication, create an SSL configuration before sending the request. For more information, see [Creating SSL configurations on page 348](#).

To send a request to a JMS service:

1. Click the **Open the Generic Service Client** toolbar button () and select the **Requests** page.
2. Click **Add** () and click a type of request that you want to send or in Request Library, right-click **EndPoints** and select a type of request that you want to send.
3. In the **Configure Protocol** window, select **JMS** and specify the JMS transport configuration.
If necessary, click **New** to create a JMS transport configuration for the call.
4. Click **Add** to specify any properties that are to be sent with the call.
5. Click **Next**.
6. On the **Select Root Element** page, if the service uses a specific XML Schema Definition (XSD), select one from the list or click **Browse** to import the XSD file, and then, select the root element for the call.
If no XSD is available for the service, select **No Schema**.
7. Click **Finish**.

Result

The request is added to the **Endpoints** section of the **Request Library**.

8. In the **Request Library**, select the request element.

Result

The generic service client shows three steps: **Edit Request**, **Invoke**, and **View Response**. The details for the request are displayed under the **Edit Request** step.

9. Based on the request selected in Step 2, on the **Message** page, use the **Form**, **Tree**, or **Source** views to edit the contents of the request.
Each view shows a different format of the same data. To add or remove XML elements in the **Form** or **Tree** view, click **Schema > Validate and Assist** to comply with an XSD specified in the schema catalog.
10. On the **Transport** page, if necessary, change the transport configuration to be used by the request.
To create and edit transport and security configurations, use the **Transport** tab.
11. If you selected SOAP XML request in step 2, on the **Request Stack** page, specify whether you want to override the security or processing algorithms that are applied to the outgoing request for the WSDL file.
To add a **Response Stack** page to edit the security or processing algorithms for incoming responses, click **Show Response Stack**.




Note: These settings apply only to the current request. To edit the request or response stack for all requests that use the current WSDL file, click **Edit WSDL Security** to open the **WSDL Security Editor**.

12. When you are ready, click **Invoke** to send the service request.

Result

The generic service client sends the request and displays the message return under the **View Response** step.

What to do next

Successful requests are recorded and added to the **Request History** list. If you are using IBM® Rational® Performance Tester or IBM® Rational® Service Tester for SOA Quality, you can create a service test by clicking the **Generate Test Suite** button .

Sending a WebSphere® MQ endpoint request



You can invoke calls to services that use a WebSphere® MQ endpoint.

Before you begin

If the service uses Secure Sockets Layer (SSL) authentication, create an SSL configuration before sending the request. For more information, see [Creating SSL configurations on page 348](#).

If the service uses SOAP security for encryption, signature, or other security algorithms, you must first configure the environment with the correct libraries and configuration files, and then create a security profile for the WSDL file. For more information, see [Configuring the environment for SOAP security on page 208](#) and [Creating security profiles for WSDL files on page 257](#).

To send a request to an WebSphere® MQ service:

1. Click the **Open the Generic Service Client** toolbar button  and select the **Requests** page.
2. Click **Add**  and click a type of request that you want to send or in Request Library, right-click **EndPoints** and select a type of request that you want to send
3. In the **Configure Protocol** window, select **WebSphere MQ** and specify the WebSphere® MQ transport configuration.
If necessary, click **New** to create an WebSphere® MQ transport configuration for the call. For more information about creating a new WebSphere MQ transport configuration, see [Creating a WebSphere MQ transport configuration on page 343](#).
4. Specify the SOAP action.
If the service requires that you override the header specified in the WebSphere® MQ transport configuration, select **Override MQ protocol configuration values** and specify the correct details.
5. Click **Next**.
6. On the **Select Root Element** page, if the service uses a specific XML Schema Definition (XSD), select one from the list or click **Browse** to import the XSD file, and then, select the root element for the request.
If no XSD is available for the service, select **No Schema**.

- Click **Finish**.

Result

The request is added to the **Endpoints** section of the **Request Library**.

- In the **Request Library**, select the request element.

Result

The generic service client shows three steps: **Edit Request**, **Invoke**, and **View Response**. The details for the request are displayed under the **Edit Request** step.

- Based on the request selected in Step 2, on the **Message** page, use the **Form**, **Tree**, or **Source** views to edit the contents of the request.

Each view shows a different format of the same data. To add or remove XML elements in the **Form** or **Tree** view, click **Schema > Validate and Assist** to comply with an XSD specified in the schema catalog.

- On the **Transport** page, if necessary, change the transport configuration to be used by the request.

To create and edit transport and security configurations, use the **Transport** tab.

- If you selected SOAP XML request in step 2, on the **Request Stack** page, specify whether you want to override the security or processing algorithms that are applied to the outgoing request for the WSDL file.

To add a **Response Stack** page to edit the security or processing algorithms for incoming responses, click **Show Response Stack**.




Note: These settings apply only to the current request. To edit the request or response stack for all requests that use the current WSDL file, click **Edit WSDL Security** to open the **WSDL Security Editor**.

- When you are ready, click **Invoke** to send the service request.

Result

The generic service client sends the request and displays the message return under the **View Response** step.

What to do next

Successful requests are recorded and added to the **Request History** list. If you are using IBM® Rational® Performance Tester or IBM® Rational® Service Tester for SOA Quality, you can create a service test by clicking the **Generate Test Suite** button .

Sending OData endpoint batch requests

To test services that use OData protocol, you can send requests in a batch. The request contains HTTP operations such as GET, POST, and PUT to manage data in the service.



Before you begin

You must have sent individual requests through Generic Service Client (GSC).

About this task

When you send requests in a batch, you can group a set of operations into one HTTP request. You can start a batch request from GSC or from a service test. To initiate a batch request from a service test in the Test editor, select multiple requests to include in a batch, right-click and select **\$batch odata requests**.

To initiate a batch request from GSC, complete the following steps:

1. Click the **Open the Generic Service Client** toolbar button  and select the **Requests** page.
2. Click the **Add** icon  and click a type of request that you want to send or in Request Library, right-click **EndPoints** and select **Send a Batch Request**.
3. In the **ODATA batch information** page, select the OData version that your application supports.
4. To set HTTP headers, ensure that the **Set ODATA batch request http headers** radio button is selected.

If needed, you can change the headers on the next page of the wizard.

5. To group appropriate requests into change sets, select the **ODATA batch with changesets** radio button.
6. In **Selection of calls to batch**, select the requests to include in the batch.
If you initiated the batch request from the service test, the requests are already selected.
7. Click **Next**.
8. In the **Configure Protocol** window, select **HTTP** and specify the HTTP transport configuration.
If necessary, click **New** to create an HTTP transport configuration for the call.

To send the HTTP/2 requests, in the **Create HTTP Protocol configuration** window, click the **Activate** check box. Before capturing the HTTP/2 traffic, configure the computer. See [Preparing to record a test for the HTTP/2 service on page 215](#) for instructions.

9. Click **Finish**.

Result

The request is added to the **Endpoints** section of the **Request Library**.

10. In the **Request Library**, select the request element.

Result

The generic service client shows three steps: **Edit Request**, **Invoke**, and **View Response**. The details for the request are displayed under the **Edit Request** step.

11. On the **Transport** page, if necessary, change the transport configuration to be used by the request.
To create and edit transport and security configurations, use the **Transport** tab.
12. When you are ready, click **Invoke** to send the service request.

Result

The generic service client sends the request and displays the message return under the **View Response** step.



Sending WebSphere Java MQ endpoint requests

You can send requests to services that use a WebSphere Java MQ endpoint.

Before you begin

If the service uses Secure Sockets Layer (SSL) authentication, create an SSL configuration before sending the request. For more information, see [Creating SSL configurations on page 348](#).

To send a request to a Java MQ service:

1. Click **Open the Generic Service Client**  and select the **Requests** page.
2. Click **Add**  or in Request Library, right-click **EndPoints** and select a type of request to send.
3. In the Configure Protocol window, select **WebSphere Java MQ** and specify the transport configuration. If necessary, create the transport configuration for the call by clicking **New** (see [Creating a WebSphere Java MQ transport configuration on page 344](#)).
4. Complete the following information in the **General** tab:



Learn more about the UI elements in the General tab:

Queue

Name of the queue as defined on the WebSphere MQ server.

Message type

The types of messages are these:

- *Datagram* means that the message does not require a reply.
- *Request* means that the message requires a reply.
- *Reply* means that the message is a reply to an earlier request message.
- *Report* means that the message is reporting on some expected or unexpected occurrence, usually related to some other message. An example is a request message that contained data that was not valid.

Message Persistence

This value indicates whether the message is persistent or not. If the message is persistent, it survives the system failures and restarts of the queue manager. If the message is not persistent, it survives a restart if it is present on a queue having the NPMCLASS(HIGH) attribute. However, even with the NPMCLASS(HIGH) attribute a message does not survive a QMGR class. Nonpersistent messages on queues having the NPMCLASS(NORMAL) attribute are discarded at queue manager restart, even if the message is found on the auxiliary storage during the restart procedure.

Dynamic Reply

Select this check box for the WebSphere MQ server to dynamically create a temporary queue as a reply. If this check box is not selected, the message in Reply Queue is used.

Reply Queue

This is the name of the message queue to which the application that issued the get request for the message should send the reply and report messages.

Reply Manager

This is the name of the queue manager on which the reply-to queue is defined.

**Additional properties**

Specify the additional properties for the queues.

5. **Optional:** If necessary, complete the following information on the **Config** tab:

**Learn more about the UI elements in the Config tab:****Message Priority**

This is the priority of the message. The lowest priority is 0.

Encoding

This is the numeric encoding of numeric data in the message. This value does not apply to numeric data in the MQMD structure itself.

Expiry Interval

This is the period of time, in tenths of a second, after which the message becomes eligible to be discarded if it has not already been removed from the target queue. The expiry interval is set by the application that put the message.

Character set

This is the character set identifier of the character data in the application message data.

6. **Optional:** In the **Report** tab, select the report messages to receive.
7. **Optional:** If necessary, complete the following information in the **Context** tab:

**Learn more about the UI elements in the Context tab:****Application Identity Data**

This information is defined by the application suite. Use it to provide information about the message or its originator.

Application Origin Data

This information is defined by the application suite. Use it to provide additional information about the origin of the message.

Accounting Token

This information is needed by the application to appropriately charge for the work that is done as a result of the message.

**User ID**

This is the user identifier of the application that originated the message.

8. **Optional:** In the **Identifiers** tab, for the messages that require binary input, specify the ID in the string format in the second column. The first column is filled automatically in the hexadecimal format.
9. **Optional:** In the **Segmentation** tab, select the segment of the message and click **Next**.
10. This step is not applicable for a Text request. On the Select Root Element page, if the service uses a specific XML Schema Definition (XSD), select one from the list. If the XSD element is not listed, click **Browse** to import the XSD file, and select the root element for the request. If no XSD is available for the service, select **No Schema**.
11. Click **Finish**. The request is added to the **Endpoints** section of the Request Library.
12. In the **Request Library**, select the request element.

Result

The generic service client shows three steps: **Edit Request**, **Invoke**, and **View Response**. The details for the request are displayed under the **Edit Request** step.

13. Based on the request selected in Step 2, on the **Message** page, use the **Form**, **Tree**, or **Source** views to edit the contents of the request.
Each view shows a different format of the same data. To add or remove XML elements in the **Form** or **Tree** view, click **Schema > Validate and Assist** to comply with an XSD specified in the schema catalog.
14. On the **Transport** page, if necessary, change the transport configuration to be used by the request.
To create and edit transport and security configurations, use the **Transport** tab.
15. If you selected SOAP XML request in step 2, on the **Request Stack** page, specify whether you want to override the security or processing algorithms that are applied to the outgoing request for the WSDL file.
To add a **Response Stack** page to edit the security or processing algorithms for incoming responses, click **Show Response Stack**.




Note: These settings apply only to the current request. To edit the request or response stack for all requests that use the current WSDL file, click **Edit WSDL Security** to open the **WSDL Security Editor**.

16. When you are ready, click **Invoke** to send the service request.

Result

The generic service client sends the request and displays the message return under the **View Response** step.

What to do next

Successful requests are recorded and added to the **Request History** list. If you are using IBM® Rational® Performance Tester or IBM® Rational® Service Tester for SOA Quality, you can create a service test by clicking the **Generate Test Suite** button ()

Testing all operations in a WSDL file

You can use the generic service client to rapidly send requests to a service using all the operations in a Web Services Description Language (WSDL) file. The calls are generated with default values based on the type of data.





Before you begin

Ensure that you have a valid WSDL file. Ensure that the WSDL files use the correct syntax for the test environment. The generic service client might not work with some Web Services Description Language (WSDL) files.

If the service uses Secure Sockets Layer (SSL) authentication, create an SSL configuration before invoking the call. See [Creating SSL configurations on page 348](#) for details.

If the service uses SOAP security for encryption, signature, or other security algorithms, you must first configure the environment with the correct libraries and configuration files, and then create a security profile for the WSDL. See [Configuring the environment for SOAP security on page 208](#) and [Creating security profiles for WSDL files on page 257](#) for details.

Calls will be generated for each operation in the WSDL file using the default values for each type. For example, strings will use the default value `str`. You can change the default values in the **XML Default Values** preferences.


1. Open the generic service client and click the **Requests** tab, and then, click  **Add a WSDL file**.
2. In the Add WSDL Files window, select an existing WSDL or import a WSDL with one of the following methods:
Choose from:
 - Click **Import from File** to import a WSDL file from the file system.
 - Click **Import from URL** to download and import an online WSDL from the web.
 - Click **Import from WSRR** to import a WSDL from an IBM® WebSphere® Service Registry and Repository (WSRR). Enter the URL of the WSRR and click **Connect**. You can click  **Search** to browse the contents of the repository.
 - Click **Import from UDDI** to import a WSDL from a Universal Description Discovery and Integration (UDDI) repository. Enter the URL of the UDDI and click **Connect**. You can click  **Filter** and  **Search** to browse the contents of the repository.
3. Click **OK**.

Result

The WSDL is added to the **Call Library**.

4. In the Call Library, right-click the WSDL and select **Test WSDL Methods**.
 The call is automatically configured with any SOAP or JMS endpoints that are available in the WSDL.

What to do next

Successful calls are recorded and added to the **Request History** list. If you are using IBM® Rational® Performance Tester or IBM® Rational® Service Tester for SOA Quality, you can click the **Generate Test Suite**  button to create a service test.

Viewing message content

The **Raw Transaction Data** view displays the raw XML, text, or binary content of any service request or response that is selected in the generic service client.

About this task

The **Raw Transaction Data** view displays plain text, XML, or binary data, depending on the type of the message content.

To view text, XML, or binary message content:

1. In the generic service client, click the **View** menu, and select **Raw Transaction Data**.
If you are using IBM® Rational® Performance Tester or IBM® Rational® Service Tester for SOA Quality, click **Window > Show View > Other > Generic Service Client > Raw Transaction Data**
2. Select a service request or response.
If you are using Rational® Performance Tester or Rational® Service Tester for SOA Quality, this view is also linked to the selected request or response in service tests, service stubs, or in the test log.
3. Depending on the nature of the message content, the following actions are available:

Text mode

When a plain text element is displayed, you can select and copy text. Click **Colorize Text** to enable or disable text colorization for HTML.

XML mode

When an XML element is displayed, you can select and copy text. Click **Colorize Text** to enable or disable text colorization for XML. Click **Enable XML Pretty Serialization** to improve readability by adding line breaks and indentation to the XML content.

If the XML content is modified by a request or response stack or by the WSDL security editor, the **Stack Contents** pane displays the list of steps in the stack. You can select each step to view the changes to the XML content. You can also select one or two steps and click **Compare Steps** to open a comparison window.

Binary mode

When a binary element is displayed, you can switch between **Binary** and **Raw-ASCII** views. Right-click the binary view to perform the following actions:

- **Select**: Opens the **Select** window, where you can select binary data by string or by specifying the number of characters to select. When a portion of binary data is selected, you can copy it to the clipboard.
- **Go to Offset**: Opens the **Go to Offset** window, where you can move to bytes at a particular offset.
- **Find**: Opens the **Find** window, where you can search for and replace binary data in a number of formats.
- **Encodings**: Select the encoding to use for displaying binary data in the text column.

Synchronizing a remote WSDL file

For web services that make their Web Services Description Language (WSDL) file available from a URL, you might have to ensure that the WSDL that you work with is always up to date. By synchronizing the WSDL, you ensure that the local copy of the WSDL in your workspace is regularly synchronized with the remote WSDL.


Before you begin

Ensure that you have a valid WSDL file. Ensure that the WSDLs use the correct syntax for the test environment. The product might not work with some WSDL files.

WSDL synchronization only works with remote WSDLs that are imported from a URL.

The WSDL synchronization runs either when the workbench is started or after a specified period. If the remote WSDL changes, the local copy of the WSDL is updated. Depending on the changes, a merge is performed and any service requests that use the WSDL are updated. If the changes to the WSDL cannot be automatically applied to the service requests, for example if an operation is removed or renamed or if the XML structure of the service request is changed, the test is marked with a error.

To import a synchronized remote WSDL:

1. Open the generic service client, click the **Requests** tab, and then, click **Add a WSDL file** .
2. In the **Add WSDL Files** window, click **Import from URL** to download and import a remote WSDL from the web.
3. On the **Import WSDL from URL** page, type the URL of the remote WSDL.
 - If you are connecting through a proxy or a corporate firewall, click **Proxy properties** to specify your network settings.
4. In the **Synchronization policy** area, specify whether and when to synchronize WSDLs:
 - Choose from:**
 - Select **Never** if you do not want the remote WSDL to be updated.
 - Select **On session launch** to synchronize the WSDL each time you start the workbench.
 - Select **Every** to specify a synchronization period in days.
5. Click **OK**.

Result

The WSDL is added to the **Call Library**.

What to do next

After the WSDL is imported, you can change the synchronization settings by right-clicking the WSDL in the generic service client **Call Library** or in the test navigator. Then select **WSDL Synchronization**. The **WSDL Synchronization** window also displays the date of the latest synchronization.

Related information

[Sending service requests with WSDL files on page 350](#)

[Testing all operations in a WSDL file on page 361](#)

Synchronizing a local WSDL file with GSC

If you edit a local WSDL source file, the Generic Service Client (GSC) should display the changes in the UI. You must keep the GSC up-to-date with the WSDL changes to ensure that you test the latest service request.

About this task

When you set GSC to automatically pick the WSDL changes, the GSC calls are fully re-created. This means that when you make some changes to WSDL, there might be some content that you did not change, however, was dependent on the changed content. Therefore, when you use this preference, the whole structure of the GSC calls is re-created.

The Request History view in GSC shows the changes occurred to the WSDL file.

To apply the local WSDL changes in GSC:

1. Click **Window > Preferences > Generic Service Client**.
2. Select the **Apply WSDL changes to GSC** check box.

Adding static XML headers to a service request

You can add static XML headers to service requests to ensure compliance with WS-Addressing, WS-ReliableMessaging, and WS-Coordination specifications as well as other predefined standards.

About this task

Static XML headers are compliant with the web service specifications for service-oriented architecture (SOA). Checks are performed to ensure that the XML headers are valid.

To add a static XML header to a request:

1. Open a service request in the generic service client. The location of the XML header depends on the product that you are using:
Choose from:
 - For IBM® Security AppScan®, click the **Request Stack** tab and in the algorithm stack for the request, click **Add > Static XML Headers**.
 - For IBM® Rational® Performance Tester, IBM® Rational® Service Tester for SOA Quality, or other products, click the **Message** tab and click **Form**.
2. On the **Header** bar, click **Add (+)** to open the menu.
3. Select the web service specification for the request to be comply with, or click **More** to open a detailed list of specifications.

Result

The XML structure of the header is created.

4. Edit the header as required.
Some elements require completion or content to be specified. XML elements that are invalid or require attention are marked with a warning or an error symbol.

Related information

[Editing WSDL security profiles on page 256](#)

[Adding WS-Addressing to a security configuration on page 270](#)

Opening file attachments

When a service sends a file attachment with the response, you must import it as a resource to open the attachment.

Before you begin

Ensure that you have specified an editor to view the attachment type in. Click **Window > Preferences > General > Editors > File associations**, and specify the editor.

1. Open the message return, and click the **Attachment** tab.
File attachments are listed with a default name, a MIME type, and a contents ID.
2. Select the line for the attachment that you want to open, and click **Open**.
3. In the **Create Resource** window, type a name for the resource, and select a location where it will be imported, and click **OK**.
Ensure that the name of the resource includes a file extension that is compatible with the MIME type of the attachment.

What to do next

After the attachment has been imported, you can click on **Open** again to open the file in the corresponding editor.

Chapter 7. Test Execution Specialist Guide

This guide describes tasks that you can perform on schedules, test execution with custom code, and Extending Rational® Service Tester for SOA Quality to support other protocols. This guide is intended for testers or test execution specialists.

Running schedules

After you have added the user groups, tests, and other items to a schedule, and you are satisfied that it represents a realistic workload, you run the schedule.

Running a local schedule or test

You can run a test locally or a schedule, in this context, is used to refer to VU Schedule and Rate Schedule on remote locations with a default launch configuration.

Before you begin

To play back tests against the applications that require client authentication such as Digital Certificates or Kerberos, you must provide the appropriate authentication before playing back the test.

- To play back a test with a digital certificate, see [Playing back a test with a digital certificate on page 237](#).
- To play back a test that require Kerberos authentication, see [Generating tests that use Kerberos](#).

About this task

When you run a schedule or test in this way, IBM® Rational® Performance Tester automatically sets up a simple launch configuration. A test runs on the local computer, with one user. A schedule runs with the user groups or Rate Runner groups and the locations that you have set. However, the execution results have a default name (the same as the schedule or test, with a suffix) and are stored in a default location.

The Rate Schedule can be run only on agent locations. The Rate Schedule can be run on agents that were installed only with PVU-based licenses.

When you run a schedule with multiple agents, an agent might be lost, especially during the long load test run. Losing an agent is not common and occurs during some extreme cases such as when computer's memory is exhausted. When an agent is lost, by default, the schedule is stopped. When the schedule is stopped in this manner, you must fix the reason of agent loss or add more agents before running the schedule. To continue to run the schedule without the lost agent, in the Schedule editor, click the **Advanced** tab and clear the **Loss of an agent halts execution** check box.

Typically, the agents divide the load among themselves. So, running a schedule without the lost agent might give unpredictable results. If you use a segmented dataset and if you run a schedule without the lost agent, the data is not redistributed among the surviving agents. Also, if the schedule has multiple stages, by default, the load is distributed among the surviving agents at the next stage. But, if the **Replace lost users in current stage** check box is selected, then the load is distributed evenly among the surviving agents in the current stage. If the check box is cleared and a

percentage of users or clients are allowed to exit during stage execution, the load is distributed among the surviving agents in the next stage. Loss of an agent in a schedule run is logged in the Performance Report.

To stop a test gracefully without causing incomplete page hits, select the **Active actions are allowed to complete if stop requested** check box at **Window > Preferences > Test > Test Execution**.

To receive email notification for the status of the run, specify the email properties in **Window > Preferences > Test > Test Execution**.

1. In the Test Navigator, expand the project until you locate the schedule or test.
2. Right-click the schedule or test, and then click **Run As > Performance Schedule** or **Run As > Test**.



Note: If you run an HTTP schedule on a remote Macintosh computer, the test fails. The cipher suite that is used for recording must be available in Oracle JDK on the Macintosh computer. For example, you can use `TLS_RSA_WITH_AES_128_CBC_SHA` on Macintosh.

Results

After you run a test or a schedule, the Performance Test Runs view opens. In this view, you can add comments about the selected result and view the settings that were used to run the schedule. To add comments, in the lower-left panel of the Performance Test Runs view, click **User Comments**. The comments that you enter are displayed on the Summary page of performance reports. To view the settings that were used for a schedule run, click **Schedule Settings**. The Performance Test Runs View Schedule Settings page displays and shows the statistics and test log settings that were used for the run.



Note: When you record a test that includes a file download, the file is not physically saved to disk. However, you can confirm that the file was retrieved from the server by looking in the response of the request that asked for the file. One method to locate the request for large downloaded files is to look for a request with a large response size.

What to do next

You can configure a schedule or test. A typical reason for setting up a configuration is to control where the execution results are stored. For more information, see [Setting a launch configuration on page 367](#).

Setting a launch configuration

Instead of using the default launch configuration, you can specify the file name for the execution results, the name of the folder for the execution results, and, for a test, the number of users.

About this task

You generally run a schedule, in this context, is used to refer to VU Schedule and Rate Schedule by right-clicking it and selecting **Run > Run VU Schedule** or **Run > Run Rate Schedule**. However, you should set a launch configuration when:

- You want to specify a name for the execution results, or you want them in a separate folder.
- You plan to run a test outside of a schedule, in this context, is used to refer to VU Schedule and Rate Schedule, and you want to run the test with more than one user.
- You want the launch configuration to appear in your toolbar menu.
- You want the launch configuration to be available to other users.

To set a launch configuration:

1. In the Test Navigator, expand the project until you locate the schedule or test.
2. Right-click the schedule or test, and then click **Run As > Run configuration**.
If the Perspectives page is displayed, keep the defaults.
3. In the **Configurations** area on the left, click **VU Schedule** or Rate Schedule, and then click **New**.

Result

A test configuration, initially named `New_configuration`, is created. Typically, you supply a configuration name that is similar to the schedule name.

At this point, you can run the schedule if you click **Run**. However, you will not have created a meaningful configuration.

4. Click the **Test Logs** tab and check the default settings. To change the default settings, clear the **Use defaults** check box and type a file name for the execution results. The product appends a time stamp to this name. To overwrite the file each time that you run the configuration, select the **Override existing test log** check box.
5. Click the **Common** tab to inspect or modify your run preferences.
6. In **Save as**, select one of the following options:

Option	Description
Local	This launch configuration is stored in your work-space, and it is not visible to other users.
Shared	Other users have access to the launch configuration; you are asked where to store it.

7. For **Display in favorites menu**, select one or more of the following options:

Option	Description
Run	The configuration is displayed in your Run toolbar menu. If you select a toolbar menu at all, this is the logical choice for a schedule or test.
Debug	The launch configuration is displayed in your Debug toolbar menu.
Profile	The configuration is displayed in your Profile toolbar menu.

8. Verify that **Launch in background** is selected. If you do not run the configuration in the background, you cannot do anything in Eclipse until it finishes running the configuration.
9. Click **Apply**, and then click **Run** to run the configured schedule or test, or click **Close** to save the configuration and run it later.

Running a configured schedule

If you do not use the default launch configuration, you can configure the schedule and then run it.

Before you begin

You must configure the schedule before you run it. For more information, see [Setting a launch configuration on page 367](#).

1. In the Test Navigator, expand the project until you locate the schedule.
2. Right-click the schedule, and then click **Run > Run**.
3. In the Configurations area on the left, click **Test Schedule**, and then click the name of the schedule to run.
4. Click **Run**.

Results

While the schedule is running, the reports are updated in real time, and you can see the changes.

Configuring multiple host names for a location

You can run several locations on the same computer by configuring multiple host names for a location. This configuration affects all tests running at that location; all tests will run with the configured port.

To configure multiple host names for a location:

1. Open the hosts file, which maps IP addresses to hosts, with an ASCII editor.
On Windows™, the hosts file is in `C:\Windows\system32\drivers\etc\hosts`. On Linux™, the hosts file is in `\etc\hosts`.
2. At the end of the hosts file, add your IP mappings. Use one IP address, but map it to two (or more) logical host names.

Example

For example, you could add map the IP address 123.4.5.6. to two logical hosts, as show in the bottom two lines:

Result

```
# Copyright (c) 1993-1999 Microsoft Corp.
#
# This is a sample HOSTS file used by Microsoft TCP/IP for Windows.
#
# This file contains the mappings of IP addresses to host names. Each
# entry should be kept on an individual line. The IP address should
# be placed in the first column followed by the corresponding host name.
# The IP address and the host name should be separated by at least one
# space.
#
# Additionally, comments (such as these) may be inserted on individual
# lines or following the machine name denoted by a '#' symbol.
#
# For example:
#
#       102.54.94.97       rhino.acme.com           # source server
#       38.25.63.10      x.acme.com              # x client host
#
127.0.0.1       localhost
123.4.5.6       remlocationa
123.4.5.6       remlocationb
```

3. Create two deployment locations that have names identical to the names you added in the hosts file:

- a. Open the schedule that contains the user group that you want to run on multiple hosts.
- b. Open the user group, and click the **Location** tab.
- c. Click **Add > Add New**, and enter the location data. Make sure the locations have different directories (in this example, they are tempa and tempb).

Example

Result

Name	Hostname	Directory
remlocationa	123.4.5.6	D:\\tempa
remlocationb	123.4.5.6	D:\\tempb

- d. Click **Finish**.

Automating tests

You can run a schedule from the command line. You can also set preferences to export results after the run completes from the command line or from the workbench. Together, these features let you run tests and analyze results without opening the workbench. You can even write scripts to process the exported results.

Creating a command-line config file

You can create a command-line config file from Rational® Performance Tester which you can use while running tests or schedules from the Command-line interface (CLI) and Maven.

Before you begin

You must have completed the following tasks:

- Created tests in Rational® Performance Tester. See Recording HTTP tests.
- Installed Maven if you are running tests from the Maven build. For more information, refer to [Apache Maven Documentation](#).

About this task

You can create a command-line config file from Rational® Performance Tester by right-clicking the test asset. The required parameters are automatically assigned, and you can specify any optional parameters, while creating the config file. You can use the config file to run the tests or schedules from the command-line interface and Maven plug-in that is provided with Rational® Performance Tester.

1. Open Rational® Performance Tester.
2. Browse and select the test or schedule from the **Test Navigator** pane.
3. Right-click the selected test or schedule, and then click **Create command line config file**.
4. Enter a name for the new configuration file in the **File name** field, and then click **Next**.
5. Perform the following steps in the **Command Line Arguments** window:
 - a. Select the format of the config file from the following options:
 - **Regular** – Select this format to run tests or schedules from CLI.
 - **Maven** – Select this format to run tests or schedules from the Maven build.
 - b. Specify the values in the fields from the available configuration options, if you want to add more parameters to the config file.
6. Click **Finish**.

Result

The **Config file created** dialog displays with the location of the config file.

Results

You have created the command line config file.

What to do next

You can run the tests either from the Maven build or CLI. See [Running a test or schedule from a command line on page 395](#) or [Testing with Maven on page 185](#).

Activating secure storage of dataset passwords

You can store the encrypted dataset passwords in the Eclipse secure storage location, so that when you run the tests from the command line, Rational® Performance Tester automatically uses the password and completes the test run.

Before you begin

You must have created an encrypted dataset. See [Encrypting a dataset column on page 306](#).

1. Open Rational® Performance Tester.
2. Click **Window > Preferences > Test > Test Execution > Automation Security**.
3. Select the **Activate Secure Storage Support for Encrypted Datasets** check box.
The password is storing in the Eclipse secure storage. You must not share the computer's login credentials with others.
4. Click **Add**, and then select the encrypted dataset from the project.
5. Click **OK**.

Result

The **Enter Password** dialog is displayed.

6. Enter the password that you used while encrypting the dataset, and then click **OK**.

Results

You have added the password of the encrypted dataset in Rational® Performance Tester.

What to do next

You can run the tests from CLI. See [Running a test or schedule from a command line on page 395](#).


When you run the tests from the command line, the test runs completes without the need to specify the password. If another user runs the same tests with encrypted datasets, the dataset password must be entered for the tests to run successfully.


Configuring to export report counters automatically

When you run a test or schedule from the Command-line interface (CLI), you can configure Rational® Performance Tester to export reports automatically.

1. Open Rational® Performance Tester.
2. Click **Windows > Preferences > Test > Performance Test Reports > Export Reports**.
3. Perform the following actions in the **Export Reports** window:

Options	Actions
Command Line	<p>Select this checkbox to export the report after the test run is complete when you run the test or schedule from the command-line interface. The report is stored in the Exported Results Data directory. You can access this report from the Test Navigator pane.</p> <p>The report is also saved in the Results directory. You can access the Results directory by navigating to the location of your workspace.</p> <p>For example, <code>C:\Users\username\IBM\rational_sdp\workspace_name\project_name\Results</code></p> <p>If you select the Command Line option, you can also select the Print simple CSV reports to command line option. When you select these two options, then you can dis-</p>

Options	Actions
	<p>play the exported data on the command-line interface, if you used the -stdout parameter. Otherwise, the report information is printed in the <code>CommandLineLog.txt</code> file. You can access this file from the following path:</p> <pre>C:\Users\username\AppData\Local\Temp\CommandLineLog.txt</pre> <p> Note: The command line preferences take the precedence over preferences set in Rational® Performance Tester.</p> <p>Consider that you selected the Command Line and Print simple CSV reports to command line options. You then run the test or schedule from the command-line interface by using the -exportstats or -exportstatsformat parameter. In this scenario, the report is exported to the path mentioned for the -exportstats parameter.</p>
Workbench	<p>Select this checkbox to export report after the test run is complete when you run the test or schedule from Rational® Performance Tester. The report is stored in the Exported Results Data directory. You can access this report from the Test Navigator pane.</p> <p>The report is also saved in the Results directory. You can access the Results directory by navigating to the location of your workspace.</p> <p>For example, <code>C:\Users\username\IBM\rational_sdp\workspace_name\project_name\Results</code></p>
Simple (one value per counter and per time range)	<p>Select this checkbox and choose the format of the report from the drop-down list to export a simple report.</p> <p>You can choose either <code>CSV</code> or <code>JSON</code> from the drop-down list.</p>
List All Time Ranges	<p>Select this checkbox to include data from all the time ranges. The default time range that is included is <code>Entire Run</code>.</p>
Full (one value per counter and per sample interval)	<p>Select this checkbox and choose the format of the report from the drop-down list to export a comprehensive report that includes the result name, node name, and time ranges. Generally, you do not include these details unless you export the customized reports that include counters from specific test runs.</p> <p>You can choose either <code>CSV</code> or <code>JSON</code> from the drop-down list.</p>

Options	Actions
Split output if counters number exceeds	Select this checkbox and specify a value to create multiple CSV or JSON files if the number of counters in the exported file exceed the specified value.
Include per instance counters	Select this checkbox to include detailed data of counters for each page.
Export each agent separately	Select this checkbox to group the data in the exported CSV or JSON file by agent that is used during test run.
One file per agent	<p>Select this checkbox to group the data by agent as a separate CSV or JSON files.</p> <p> Note: The One file per agent option is available only when you select the Export each agent separately option.</p>
File name format	<p>Select a format for the file name of the exported file from the drop-down list. You can select one of the following options from the drop-down list:</p> <ul style="list-style-type: none"> ◦ Pre-9.0 release: Select this option to save the exported report file name as [REPORT_NAME][AGENT_NAME][SPLIT_INDEX]. ◦ 9.0 release: Select this option to save the exported report file name as [TEST_NAME][AGENT_NAME][SPLIT_INDEX]_[TIME_STAMP]_[REPORT_ID]_[EXPORT_TYPE]. ◦ Custom pattern: Select this option and specify a pattern that you would like to save the exported report file.
HTML report	Select this checkbox to export full report data in a HTML format.
Executive Summary report	<p>Select this checkbox to export the executive summary report in a HTML format. The report that you selected from the drop-down list summarizes the state of the test or schedule run and displays the report on only one HTML page. You can print this report.</p> <p>The Executive Summary report is saved in the Results directory. You can access the Results directory by navigating to the location of your workspace.</p> <p>For example, <code>C:\Users\username\IBM\rational_sdp\workspace_name\project_name\Results</code></p>

Options	Actions
Select reports to export	<p>Select the required reports checkbox from the list to export reports into CSV, JSON, or HTML formats.</p> <p>If you select one or more reports, then each report is exported to a separate CSV or JSON file in the Exported Results Data directory. You can access these reports from the Test Navigator pane.</p> <p>Select the Show Report Ids checkbox to view the ID of each report. The IDs are used when you export the specific reports from the command-line interface.</p>

4. Click **Apply and Close**.

Results

You have configured Rational® Performance Tester to export test reports automatically into the CSV or JSON file.

Command-line parameters

You can find the information about parameters that you can use while running tests from the command-line interface (CLI).

Rational® Performance Tester supports the usage of CLI to run tests. Based on the operating system where you installed Rational® Performance Tester, you can go to the following directory to use CLI:

Operating system	The default path to the directory
Windows®	C:\Program Files\IBM\SDP\cmdline
Linux™	/opt/IBM/SDP/cmdline
Mac	/Application/IBM/SDP/cmdline

After navigating to the directory, you can run the test by using CLI. The CLI supports various parameters. Based on your requirements, you can use those parameters while running tests. You can enter `cmdline -help` in CLI to view all the `checkmark_perf.jpg` parameters.

The workspace is locked after you run the tests from CLI. To check the progress of the `test` or `schedule` during the run, you can invoke another workspace and open the project through that workspace.

The CLI does not provide a way to specify the secure storage password for encrypted datasets. You must provide the password in Rational® Performance Tester and ensure that it is stored and persisted in the test before you run the test that is associated with an encrypted dataset from CLI. See [Activating secure storage of dataset passwords on page 371](#).

You must provide the values for certain parameters to run tests from CLI. The minimum parameters that you need to provide are **-workspace**, **-project**, and **-suite** or **-schedule**, or **-configfile**. The remaining parameters are optional. For

example, if you provide a complete path of a configuration file for the **-configfile** parameter that contains the values of the parameters for a `test` or `schedule` run, then you need not provide values for the **-workspace**, **-project**, and **-suite** or **-schedule** parameters.



Note: If Rational® Performance Tester is not installed in default location, then you must provide the value for **-eclipsehome** and **-plugins** parameters along with other mandatory parameters.

You can refer to the following tables to know the required and optional parameters and its description:

Table 2. Required parameters for CLI


Pa- ra- me- ters	Description
- work- space	Enter the complete path of the Eclipse workspace. For example, <pre>cmdline -workspace "C:/Users/IBM/rationalsdp/workspace"</pre>
- project	Enter the path including the file name of the project relative to the workspace. For example, <pre>cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "proj123"</pre> OR <pre>cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "/myproj/proj123"</pre>
- suite	Enter the path including the file name of the performance test relative to the project. For example, <pre>cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "/myproj/proj123" -suite "test1.testsuite"</pre> OR <pre>cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "/myproj/proj123" -suite "Tests/test1.testsuite"</pre> <p> Note: You must not use both the -suite and -schedule parameters simultaneously in a single command.</p> <p>You can specify multiple tests separated by a colon character (:) to run multiple tests simultaneously.</p> <p>For example,</p> <pre>cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "/myproj/proj123" -suite "test1.testsuite:test2.testsuite:test3.testsuite"</pre>

Table 2. Required parameters for CLI (continued)


Pa- ra- me- ters	Description
	<p>You can run any or all tests and compound tests that are in a folder by using the wildcard character * enclosed in quotes in different combinations in the command line.</p> <p>cmdline -workspace <workspace_full_path> -project <project_relative_path> -suite "*".</p> <p>For example, you can run the following commands:</p> <ul style="list-style-type: none"> • cmdline -workspace "workspace" -project myProject -suite "*" to run all the tests in myProject. • cmdline -workspace "workspace" -project myProject -suite "test*" to run all the tests in myProject that have the file names starting with 'test'. • cmdline -workspace "workspace" -project myProject -suite "Compound Tests/*" to run all the tests in the Compound Tests folder under myProject. • cmdline -workspace "workspace" -project myProject -suite "Web*.testsuite:!Web2" to run all the tests in myProject that have the file names starting with 'Web', but not 'Web2' • cmdline -workspace "workspace" -project myProject -suite "Web*.testsuite:/myProject2/Test1*.test-suite" to run all the tests in myProject that have the file names starting with 'Web' and all the tests in myProject2 that have the file names starting with 'Test1'
- sched- ule	<p>Enter the path including the file name of the schedule relative to the project.</p> <p>For example,</p> <pre>cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "/myproj/proj123" -schedule "sch1"</pre> <p>OR</p> <pre>cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "/myproj/proj123" -schedule "Schedules/sch1"</pre> <p> Note: You must not use both the -schedule and -suite parameters simultaneously in a single command.</p> <p>You can specify multiple schedules separated by a colon character (:) to run schedules simultaneously.</p> <p>For example,</p> <pre>cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "/myproj/proj123" -schedule "sch1:sch2:sch3"</pre> <p>You can run the following command to run the schedule: cmdline -workspace <workspace_full_path> -project <project_relative_path> -schedules "sch1*"</p>

Table 2. Required parameters for CLI (continued)



Pa- ra- me- ters	Description
	For example, <code>cmdline -workspace "workspace" -project "PT1" -schedules "sch1*"</code>
- con- fig- file	<p>Enter the complete path to a configuration file that contains the parameters for a <code>test</code> or <code>schedule</code> run.</p> <p>For more information about creating a command line config file from Rational® Performance Tester, see the Related information section.</p> <p> Note: If you provide a complete path of a configuration file for the <code>-configfile</code> parameter, then you need not provide values for the -workspace, -project, and -suite parameters.</p> <p>The contents of a default configuration file are as follows:</p> <pre data-bbox="282 852 846 1052">eclipsehome=C:/Program Files/IBM/SDP/ plugins=C:\Program Files\IBM\IBMIMShared\plugins workspace=C:/Users/IBM/rationalsdp/workspace project=proj1 suite=Tests/test1.testsuite exportlogmillis=false overwrite=false</pre> <p> Note: If you are creating a config file manually, the file must be in the UTF-8 format. You must not use double quotation marks (") in the file even for values that contain spaces.</p>

Table 3. Optional parameters for CLI

Parameters	Description
-compare	<p>Use this parameter along with -exportstatshtml to export the result in compare mode.</p> <p>The value can be paths to the runs and are relative to the workspace. You must separate the paths by a comma.</p>
-duration	<p>Use this parameter to change the duration of the stages in the rate schedule.</p> <p>For example,</p> <pre data-bbox="630 1734 1377 1843">cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "proj123" -suite "test1.testsuite" -rate "Rate Runner Group 1=1/s, 3/m" -duration "Stage1=10s; Stage2=5m"</pre>

Table 3. Optional parameters for CLI (continued)


Parameters	Description								
	<p>The stage number specified must exist in the rate schedule.</p> <p>You can specify multiple stages separated by semicolon (;). The time units that you can specify for the duration are as follows:</p> <ul style="list-style-type: none"> • ms (millisecond) • s (second) • m (minute) • h (hour) • d (day) <p> Note: The -duration parameter creates a new copy of the rate schedule that contains the specified number of duration.</p>								
-eclipsehome	<p>Use this parameter to provide the complete path to the directory that contains <code>eclipse.exe</code>, if Rational® Performance Tester is not installed in the default location.</p> <p>The default location of <code>eclipse.exe</code> is as follows:</p> <table border="1" data-bbox="610 1087 1419 1310"> <thead> <tr> <th data-bbox="610 1087 881 1138">Operating system</th> <th data-bbox="881 1087 1419 1138">The default path to the directory</th> </tr> </thead> <tbody> <tr> <td data-bbox="610 1138 881 1199">Windows®</td> <td data-bbox="881 1138 1419 1199">C:\Program Files\IBM\SDP</td> </tr> <tr> <td data-bbox="610 1199 881 1255">Linux™</td> <td data-bbox="881 1199 1419 1255">/opt/IBM/SDP</td> </tr> <tr> <td data-bbox="610 1255 881 1310">Mac</td> <td data-bbox="881 1255 1419 1310">/Application/IBM/SDP</td> </tr> </tbody> </table> <p>For example,</p> <pre data-bbox="626 1392 1403 1476">cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "/myproj/proj123" -suite "test1.testsuite" -eclipsehome D:\Testtool\IBM\SDP</pre>	Operating system	The default path to the directory	Windows®	C:\Program Files\IBM\SDP	Linux™	/opt/IBM/SDP	Mac	/Application/IBM/SDP
Operating system	The default path to the directory								
Windows®	C:\Program Files\IBM\SDP								
Linux™	/opt/IBM/SDP								
Mac	/Application/IBM/SDP								
-execsummary	<p>Use this parameter to export all of the reports for the test run in a printable format, also known as an executive summary, to the local computer.</p> <p>You must specify the path to store the executive summary.</p> <p>For example,</p> <pre data-bbox="626 1728 1403 1808">cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "proj123" -suite "test1.testsuite" -execsummary "C:/Users/Tets/summary"</pre>								

Table 3. Optional parameters for CLI (continued)

Parameters	Description
-execsummaryreport	<p>Use this parameter to export a specific report as an executive summary for the test run to the local computer.</p> <p>You must specify the ID of the report to export. You must use this parameter along with the -execsummary parameter.</p> <p>For example, you can specify <code>http</code> to export the executive summary of an HTTP performance report.</p> <pre>cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "proj123" -suite "test1.testsuite" -execsummary "C:/Users/Tets/summary" -execsummaryreport "http"</pre>
-exportlog	<p>Use this parameter to specify the file directory path to store the exported HTTP test log.</p> <p>For example,</p> <pre>cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "proj123" -suite "test1.testsuite" -exportlog "C:/Users/logs/logexport.txt"</pre> <p>In the -exportlog parameter, you can provide multiple values when running multiple tests. You must use colon (:) to separate the entries of the parameter value.</p> <p>For example,</p> <pre>cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "proj123" -suite "test1.testsuite:test2.testsuite" -exportlog "C:/Users/logs/logexport.txt:C:/Users/logs/logexport2.txt"</pre> <p>If there are multiple values for the -suite parameter and a single value for the -exportlog parameter, then the -exportlog parameter generates the appropriate number of test logs by appending 0, 1, 2, and so on.</p> <p>For example,</p> <pre>cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "proj123" -suite "test1.testsuite:test2.testsuite:test3.testsuite" -exportlog "C:/Users/logs/logexport.txt"</pre> <p>The preceding command generates the following test logs:</p> <ul style="list-style-type: none"> • logexport_0.txt • logexport_1.txt • logexport.txt

Table 3. Optional parameters for CLI (continued)


Parameters	Description
	<p>The last test log generated has the same name as that of value of -exportlog parameter.</p> <p> Note: If there are multiple values for the -suite and -exportlog parameters, then the number of -suite values must match with the number of -exportlog values. Otherwise, the following error message is displayed:</p> <pre>Error, number of -suite and -exportlog entries do not match.</pre>
-exportlogmillis	<p>Use this parameter if you want to view the timestamp of the test logs in milliseconds.</p> <p>The default value for the -exportlogmillis is set to <code>false</code>. Therefore, the timestamp in the test logs does not contain milliseconds.</p>
-exportstatreportlist	<p>Use this parameter to specify the IDs of reports that you want to export in place of the default report.</p> <p>You can provide multiple report IDs separated by a comma. You can navigate to the Preferences of Rational® Performance Tester (Window > Preferences > Test > Performance Test Reports > Export Reports), and then you can select Show Report IDs checkbox to view the report IDs.</p> <p>You must use the -exportstatreportlist parameter along with the -exportstats or -exportstatshtml parameter.</p> <p>For example,</p> <pre>cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "proj123" -suite "test1.testsuite" -exportstats "C:/Users/stats" -exportstatreportlist "http"</pre>
-exportstats	<p>Use this parameter to provide the complete path to a directory that you can use to store the exported report in a comma-separated values (CSV) format.</p> <p>For example,</p> <pre>cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "proj123" -suite "test1.testsuite" -exportstats "C:/Users/stats"</pre>
-exportstatsformat	<p>Use this parameter to specify a format for the result that you want to export along with the -exportstats parameter.</p>

Table 3. Optional parameters for CLI (continued)


Parameters	Description
	<p>You must use at least one of the following arguments with the -exportstats-format parameter:</p> <ul style="list-style-type: none"> • simple.csv • full.csv • simple.json • full.json • csv • json <p>For example,</p> <pre>cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "proj123" -suite "test1.testsuite" -exportstats "C:/Users/stats" -exportstatsformat "simple.json"</pre> <p>You can add multiple arguments separated by a comma.</p> <p>For example,</p> <pre>cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "proj123" -suite "test1.testsuite" -exportstats "C:/Users/stats" -exportstatsformat "simple.json,full.csv"</pre> <p>When you want to export both simple and full type of test results in a <code>json</code> or <code>csv</code> format, you can specify <code>json</code> or <code>csv</code> as the arguments in the command. When the test run completes, the test result exports to <code>simple.json</code> and <code>full.json</code> files.</p> <p>For example,</p> <pre>cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "proj123" -suite "test1.testsuite" -exportstats "C:/Users/stats" -exportstatsformat "json"</pre> <p>You can select the Command Line checkbox from the product Preferences (Window > Preferences > Test > Performance Test Reports > Export Reports) when you want to automatically export test results to one of the selected formats after the test run completes.</p> <p> Remember: When you run the test from CLI, and if you use the -exportstats parameter, then CLI preferences take precedence over the Preferences set in the product. Therefore, the default format of the exported test result is CSV.</p>

Table 3. Optional parameters for CLI (continued)



Parameters	Description
	 For example, when you select the Command Line option and Report format to json in the product Preferences, and run the test from CLI by using the -exportstats parameter, then the result is exported to a csv file after the test run is complete.
-exportstatshhtml	<p>Use this parameter to provide the complete path to a directory that you can use to export web analytic results.</p> <p>For example,</p> <pre data-bbox="625 699 1419 789">cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "proj123" -suite "test1.testsuite" -exportstatshhtml "C:/Users/reports"</pre> <p>The results are exported to the specified directory. You can analyze the results on a web browser without using Rational® Performance Tester.</p>
-history	<p>Use this parameter when you want to view a record of all events that occurred during a <code>test</code> or <code>schedule</code> run.</p> <p>You can use any of the following options:</p> <ul style="list-style-type: none"> • jaeger: To send test logs to the Jaeger UI during the <code>test</code> or <code>schedule</code> run. <p> Note: You must set the <code>JAEGER_AGENT_HOST</code> property as an environment variable by using the command line before you use the jaeger option to send test logs to the Jaeger UI.</p> <ul style="list-style-type: none"> • testlog: To send test logs as traditional test logs in Rational® Performance Tester during the <code>test</code> or <code>schedule</code> run. • null: To send no test logs either to the Jaeger UI or Rational® Performance Tester during the <code>test</code> or <code>schedule</code> run. <p>For example,</p> <pre data-bbox="625 1665 1419 1755">cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "proj123" -suite "test1.testsuite" -history "jaeger"</pre> <p>You can add multiple arguments separated by a comma to send test logs during the <code>test</code> or <code>schedule</code> run to Rational® Performance Tester and the Jaeger UI.</p>

Table 3. Optional parameters for CLI (continued)


Parameters	Description
	<p>For example,</p> <pre data-bbox="626 365 1382 453">cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "proj123" -suite "test1.testsuite" -history "jaeger,testlog"</pre> <p>For more information about how to view test logs in the Jaeger UI and Rational® Performance Tester, see the Related information section.</p>
ignoreunhealthytranscations	<p>Use this parameter if you do not want to record the elapsed time for the unhealthy transactions during the test run.</p> <p>The default value for the ignoreunhealthytranscations parameter is set to <code>false</code>. Therefore, the elapsed time is recorded for unhealthy transactions.</p> <p> Note: The value provided in the ignoreunhealthytranscations parameter always take precedence over the Discard time measurements for unhealthy transactions option set in the product preferences (Window > Preferences > Test > Test Execution).</p>
-import	<p>Use this parameter to import the project from your local computer to a workspace and then run the test from the command-line interface.</p> <p>When you want to run tests that are in a source control system such as Git, you can clone the project resources from the remote repository to your computer. You must use the -import parameter along with the -workspace parameter and use an empty workspace to import test assets.</p> <p>You can use the -import parameter when you do not want to use your existing workspace or the workspace that you cloned from a remote repository.</p> <p>For example,</p> <pre data-bbox="626 1488 1317 1572">cmdline -workspace "C:/Users/IBM/rationalsdp/workspace_new" -project "proj123" -suite "test1.testsuite" -import "D:/Performance/Project1"</pre>
-importzip	<p>Use this parameter to import the project as test assets with dependencies into your workspace.</p> <p>You can run test assets from the imported zip file, but you must specify the -importzip parameter along with the -schedule or -suite parameters.</p> <p>For example,</p>

Table 3. Optional parameters for CLI (continued)

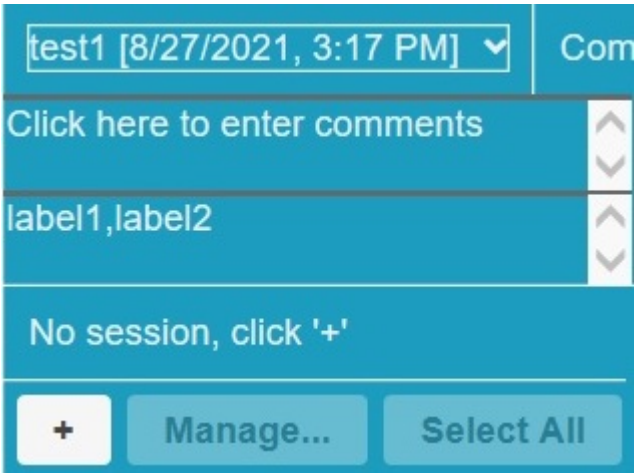
Parameters	Description
	<pre>cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "proj123" -importzip "C:\User\Desktop\test1.zip" -suite "test1.testsuite"</pre>
<p>-labels</p>	<p>Use this parameter to add labels to test results when you run test assets from CLI.</p> <p>For example,</p> <pre>cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "proj123" -suite "test1.testsuite" -publish "https://localhost:5443/#project.name=test&teamspace.name=ts1" -labels "label1,label2"</pre> <p>You can also use the -labels parameter along with the -publish parameter to add labels to a test result when you want to publish test results to Rational® Test Automation Server.</p> <p>You can add multiple labels to a test result separated by a comma. When you run test assets, then the labels that you added are displayed on the Performance Report in Rational® Performance Tester.</p>  <p>The Results page of Rational® Test Automation Server displays the label that you added, in the specific test asset in the following conditions:</p> <ul style="list-style-type: none"> • When you set Publish result after execution as Always or Prompt in the Preferences of Rational® Performance Tester (WindowsPreferencesTestRational Test Automation Server). • When you use the labels parameter along with the publish parameter.

Table 3. Optional parameters for CLI (continued)

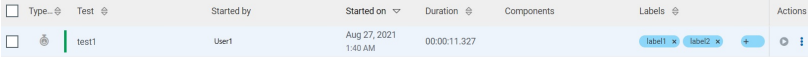

Parameters	Description
	 <p> Note: If the value of the -labels parameter contains double quotation marks (""), for example, "100" users, then CLI does not accept values for the parameter.</p> <p>To work around this problem, you must create a command-line config file, and then run the test by using the -configfile parameter. You can also use the <code>CommandLine.exe</code> file to run the command and add labels in a language that might not support Unicode characters on the Windows® operating system.</p> <p>You can locate the <code>CommandLine.exe</code> file in the following location:</p> <pre><installation_directory>\IBM\IBMIMShared\plug-ins\com.ibm.rational.test.lt.cmdlineexecute<time_stamp></pre>
<p>- overriderm1labels</p>	<p>Use this parameter to perform any of the following actions:</p> <ul style="list-style-type: none"> • Enable the Resource Monitoring from Service option for a performance schedule if the Resource Monitoring from Service option is not enabled from the schedule editor in Rational® Performance Tester. • Ignore Resource Monitoring sources that were set in the performance schedule and to change for a label matching mode. • Replace an existing set of Resource Monitoring labels that were set in the performance schedule and run the schedule with a new set of Resource Monitoring labels. <p>You must have added the Resource Monitoring labels to the Resource Monitoring sources on the Resource Monitoring page in your Rational® Test Automation Server project. You can then use those labels to collect data from the source while running the <code>schedule</code> from CLI.</p> <p>For example, if you have added a label in Rational® Test Automation Server for a Resource Monitoring source as <code>rm1</code>, then you can run the following command to collect data from the source:</p>

Table 3. Optional parameters for CLI (continued)

Parameters	Description				
	<pre data-bbox="630 317 1382 401">cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "/myproj/proj123" -schedule "sch1" -overriderm1labels "rm1"</pre> <p data-bbox="613 436 1406 506">You can add multiple labels to a schedule separated by a comma to collect data from multiple sources during the <code>schedule</code> run.</p> <p data-bbox="613 541 748 569">For example:</p> <pre data-bbox="630 594 1382 678">cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "/myproj/proj123" -schedule "sch1" -overriderm1labels "rm1,rm2"</pre> <p data-bbox="613 711 1422 821">If your label for resource monitoring contains a comma (,) then you must replace the single comma with the double comma in the -overriderm1labels parameter.</p> <p data-bbox="613 856 1398 966">For example, if you have added a label to a Resource Monitoring source as <code>rm1,test</code>, then you must run the following command to collect data from source as follows:</p> <pre data-bbox="630 991 1382 1075">cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "/myproj/proj123" -schedule "sch1" -overriderm1labels "rm1,,test"</pre>				
-overwrite	<p data-bbox="613 1100 1409 1169">Use this parameter to determine whether a result file with the same name is overwritten or not.</p> <p data-bbox="613 1205 1382 1274">The default value is <code>false</code>. If the value is set to <code>true</code>, the file is overwritten and retains the same file name.</p> <p data-bbox="613 1310 1308 1337">You must use double quotation marks (") for values <code>true</code> or <code>false</code>.</p> <p data-bbox="613 1373 748 1400">For example,</p> <pre data-bbox="630 1425 1382 1509">cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "proj123" -suite "test1.testsuite" -overwrite "true"</pre>				
-plugins	<p data-bbox="613 1537 1406 1646">Use this parameter to provide the complete path to the directory that contains the <code>plugins</code> directory, if Rational® Performance Tester is not installed in the default location.</p> <p data-bbox="613 1682 1219 1709">The default location of the <code>plugins</code> directory is as follows:</p> <table border="1" data-bbox="613 1738 1419 1883"> <thead> <tr> <th data-bbox="613 1749 881 1791">Operating system</th> <th data-bbox="881 1749 1419 1791">The default path to the directory</th> </tr> </thead> <tbody> <tr> <td data-bbox="613 1801 881 1883">Windows®</td> <td data-bbox="881 1801 1419 1883">C:\Program Files\IBM\IB-MIMShared\plugins</td> </tr> </tbody> </table>	Operating system	The default path to the directory	Windows®	C:\Program Files\IBM\IB-MIMShared\plugins
Operating system	The default path to the directory				
Windows®	C:\Program Files\IBM\IB-MIMShared\plugins				

Table 3. Optional parameters for CLI (continued)


Parameters	Description	
	Operating system	The default path to the directory
	Linux™	/opt/IBM/IBMIMShared/plugins
	Mac	/Application/IBM/IBMIMShared/plugins
	<p>For example,</p> <pre>cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "/myproj/proj123" -suite "test1.testsuite" -eclipsehome "D:\Testtool\IBM\SDP" -plugins "D:\Testtool\IBM\IBMIMShared\plugins"</pre>	
<p>-publish</p>	<p>Use this parameter to publish test results to Rational® Test Automation Server.</p> <p> Remember: Before you use the publish parameter, you must provide the offline user token of Rational® Test Automation Server by using any of the following methods:</p> <ul style="list-style-type: none"> • Set the value of the RTCP_OFFLINE_TOKEN environment variable to the offline user token of Rational® Test Automation Server by using the command-line interface. • Provide the offline user token of Rational® Test Automation Server in the preferences of Rational® Performance Tester (Window > Preferences > Test > Rational Test Automation Server) <p>You can use the following arguments along with the -publish parameter:</p> <ul style="list-style-type: none"> • no: Use this argument if you do not want to publish test results after the run. <p>This argument is useful if the product preferences are set to publish the results, but you do not want to publish them.</p> <ul style="list-style-type: none"> • You can use any of the following arguments to specify the project name: 	

Table 3. Optional parameters for CLI (continued)


Parameters	Description
	<ul style="list-style-type: none"> ◦ <code>serverURL #project.name=projectName&team-space.name=name_of_the_teamspace</code> ◦ <code>serverURL #project.name=projectName&team-space.alias=name_of_the_teamspace_alias</code> <p>For example,</p> <pre>cmdline -workspace "C:/Users/IBM/rationalSDP/workspace" -project "proj123" -suite "test1.testsuite" -publish "https://localhost:5443/#project.name=testproj"</pre> <p>Where:</p> <ul style="list-style-type: none"> ◦ <code>https://localhost:5443</code> is the URL of Rational® Test Automation Server. ◦ <code>testproj</code> is the name of the project in Rational® Test Automation Server. <p>For example,</p> <pre>cmdline -workspace "C:/Users/IBM/rationalSDP/workspace" -project "proj123" -suite "test1.testsuite" -publish "https://localhost:5443/#project.name=test&team-space.name=t s1"</pre> <p> Notes:</p> <ul style="list-style-type: none"> ◦ If you have a project with the same name in different team spaces, then you must append either the &team-space.name=name_of_the_teamspace or &team-space.alias=name_of_the_teamspace_alias options. ◦ If the name of the project or team space contains a special character, then you must replace it with <code>%<Hexvalue_of_special_character></code>. <p>For example, if the name of the team space is <i>Initial Team Space</i>, then you must provide it as <i>Initial%20Team%20Space</i>.</p> <p>Where <code>%20</code> is the Hex value of <code>space</code> character.</p>

Table 3. Optional parameters for CLI (continued)



Parameters	Description
	<div data-bbox="704 321 750 373" style="display: inline-block; vertical-align: top; margin-bottom: 10px;">  </div> <ul style="list-style-type: none"> ◦ The values provided in the publish parameter always take precedence over the Results options set in the product preferences (Window > Preferences > Test > Rational Test Automation Server > Results). <p>The Reports information section on the <code>Log</code> file displays the names of the report along with its corresponding URLs in the following conditions:</p> <ul style="list-style-type: none"> • When you configured the URL of in Preferences of Rational® Performance Tester (Window > Preferences > Test > Rational Test Automation Server). • When you set Publish result after execution as Always or Prompt in the Preferences of Rational® Performance Tester (Window > Preferences > Test > Rational Test Automation Server > Results). • When you used publish parameter or publish parameter along with publishreports parameter. <div data-bbox="626 1014 672 1066" style="display: inline-block; vertical-align: top; margin-bottom: 10px;">  </div> <p>Remember: If you provide Rational® Test Automation Server and the project details in the product Preferences (Window > Preferences > Test > Rational Test Automation Server) and if you use the -publish parameter, the server details in CLI take precedence over the product preferences.</p>
<p>-publish_for</p>	<p>Use this parameter to publish the test results based on the completion status of the tests:</p> <ul style="list-style-type: none"> • ALL - This is the default option. Use this argument to publish test results for any text execution verdict. • PASS - Use this argument to publish test results for the tests that have passed. • FAIL - Use this argument to publish test results for the tests that have failed. • ERROR - Use this argument to publish test results for the tests that included errors. • INCONCLUSIVE - Use this argument to publish test results for the inconclusive tests.

Table 3. Optional parameters for CLI (continued)

Parameters	Description
	<p>You must use the -publish_for parameter along with the -publish parameter. You can add multiple arguments for the -publish_for parameter separated by a comma.</p> <p>For example,</p> <pre data-bbox="630 506 1425 667">cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "proj123" -suite "test1.testsuite" -publish "https://localhost:5443/#project.name=test&teamspace.name=ts1" -publish_for "PASS,ERROR"</pre>
-publishreports	<p>Use this parameter to publish specific test results to Rational® Test Automation Server.</p> <p>The arguments that you can use with the -publishreports parameter are as follows:</p> <ul data-bbox="678 898 797 968" style="list-style-type: none"> • STATS • TESTLOG <p>You must use the -publishreports parameter along with the -publish parameter.</p> <p>For example,</p> <pre data-bbox="630 1167 1425 1283">cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "proj123" -suite "test1.testsuite" -publish "https://localhost:5443/#project.name=testproj" -publishreports "STATS"</pre> <p>You can prefix the -publishreports arguments with !" to publish all the reports except the specified one.</p> <p>For example,</p> <pre data-bbox="630 1472 1425 1587">cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "proj123" -suite "test1.testsuite" -publish "https://localhost:5443/#project.name=testproj" -publishreports "!"TESTLOG"</pre> <p> Note: When you run service tests, you cannot publish the functional report directly from the CLI to Rational® Test Automation Server.</p>

Table 3. Optional parameters for CLI (continued)



Parameters	Description
	 You must publish the functional report by using Rational® Performance Tester. For more information about generating and publishing the functional report, see the Related information section.
-quiet	<p>Use this parameter when you do not want to display the values and acceptance status of the parameters on the command-line interface.</p> <p>For example,</p> <pre data-bbox="625 661 1412 745">cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "proj123" -suite "test1.testsuite" -quiet</pre>
-rate	<p>Use this parameter to specify a rate that you want to achieve for a workload in the Rate Runner group.</p> <p>For example,</p> <pre data-bbox="625 934 1412 1018">cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "proj123" -suite "test1.testsuite" -rate "Rate Runner Group 1=1/s, 3/m"</pre> <p>Where, <code>Rate Runner Group1</code> is the name of the rate runner group that has two stages. The desired rate for the first stage is one iteration per second and the rate for the second stage is three iterations per minute.</p> <p> Notes:</p> <ul data-bbox="747 1281 1404 1438" style="list-style-type: none"> • The name of the Rate Runner group must match with the name in the rate schedule. • The -rate parameter creates a new copy of the rate schedule that contains the specified number of rate. <p>You can specify multiple Rate Runner groups separated by the semicolon (;). The time units that you can specify for the rate are as follows:</p> <ul data-bbox="673 1638 812 1795" style="list-style-type: none"> • s (second) • m (minute) • h (hour) • d (day)
-results	Use this parameter to specify the name of the results file.

Table 3. Optional parameters for CLI (continued)

Parameters	Description
	<p>The default name of the result file is the <code>test</code> or <code>schedule</code> name with a time-stamp appended. You must specify a folder name that is relative to the project to store the test results.</p> <p>For example,</p> <pre>cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "proj123" -suite "test1.testsuite" -results "Results/testresults1"</pre>
-stdout	<p>Use this parameter to display the information about the <code>test</code> or <code>schedule</code> on CLI.</p> <p>After you run the <code>test</code> or <code>schedule</code> from CLI, the following outputs are displayed to give you the overall information of the <code>test</code> or <code>schedule</code>:</p> <ul style="list-style-type: none"> • --VERDICT: The verdict of the <code>test</code> or <code>schedule</code>. • --REMOTE_RESULT: The URL of the result published to Rational® Test Automation Server. • --REMOTE_RESULT_UI: The URL of the result published to Rational® Test Automation Server and can be opened in a browser to analyze the result. • --LOCAL_RESULT: The path of the result saved locally. <p>For example,</p> <pre>cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "proj123" -suite "test1.testsuite" -publish "https://localhost:5443/#project.name=testproj" -stdout</pre>
-swapdatasets	<p>Use this parameter to replace the dataset values during a <code>test</code> or <code>schedule</code> run.</p> <p>If the <code>test</code> or <code>schedule</code> is associated with the dataset, you can replace the dataset at run time while initiating the run from CLI.</p> <p>You must ensure that both original and new datasets are in the same workspace and have the same column names. You must also include the path to the dataset when you use the -swapdatasets parameter. You must provide the values for the -swapdatasets parameter in the following format:</p> <pre>/project_name/ds_path/original_ds.csv:/project_name/ds_ path/new_ds.csv</pre> <p>For example,</p>

Table 3. Optional parameters for CLI (continued)



Parameters	Description
	<pre data-bbox="630 317 1382 432">cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "proj123" -suite "test1.testsuite" -swapdatasets "/proj1023/Datasets/ds1.csv:/proj1023/Datasets/new_ds1.csv"</pre> <p data-bbox="613 464 1357 533">You can swap multiple datasets that are saved in a different project by adding multiple paths to the dataset separated by a semicolon (;).</p>
-timerange	<p data-bbox="613 573 1398 642">Use this parameter along with -exportstats and -exportstatshtml to export test results within one or more time ranges.</p> <p data-bbox="613 678 1398 747">The value of the -timerange parameter is the time range that you specify in the schedule.</p> <p data-bbox="613 783 748 810">For example,</p> <pre data-bbox="630 835 1382 919">cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "proj123" -suite "test1.testsuite" -exportstats "C:/Users/stats" -timerange "10 Users"</pre> <p data-bbox="613 951 1373 1020">You must separate time ranges with a comma and use double quotation marks (") when there is space in a time range.</p>
-usercomments	<p data-bbox="613 1087 1365 1157">Use this parameter to add text that you want to display in the user comments row of the report.</p> <p data-bbox="613 1192 748 1220">For example,</p> <pre data-bbox="630 1245 1409 1329">cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "proj123" -suite "test1.testsuite" -exportstats "C:/Users/stats" -usercomments="test run with dataset"</pre> <p data-bbox="630 1371 1398 1486"> Note: If the value of the -usercomments parameter contains double quotation marks ("), for example, <code>test run with "dataset"</code>, then CLI does not accept values for the parameter.</p> <p data-bbox="683 1528 1398 1759">To work around this problem, you must create a command-line configuration file by using Rational® Performance Tester, and then run the test by using the -configfile parameter. You can also use the <code>CommandLine.exe</code> file to run the command and to add comments in a language that might not support Unicode characters on the Windows® operating system.</p> <p data-bbox="683 1791 1398 1816">You can locate the <code>CommandLine.exe</code> file in the following location:</p>

Table 3. Optional parameters for CLI (continued)

Parameters	Description
	 <pre data-bbox="678 323 1403 426"><installation_directory>\IBM\IBMIMShared\plug- ins\com.ibm.rational.test.lt.cmdlineexecute<time_- stamp></pre>
-users	<p data-bbox="610 499 1416 562">Use this parameter to override the default number of virtual users in the <code>test</code> or <code>schedule</code> run.</p> <p data-bbox="610 604 1406 667">For a schedule, the default is the number of users specified in the schedule editor and for a test, the default is one user.</p> <p data-bbox="610 709 1399 772">The -users parameter creates a new copy of the <code>schedule</code> that contains the specified number of users.</p> <p data-bbox="610 814 750 846">For example,</p> <pre data-bbox="625 863 1377 940">cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "proj123" -suite "test1.testsuite" -users 50</pre>
-varfile	<p data-bbox="610 974 1382 1037">Use this parameter to specify the complete path to the XML file that contains the variable initialization.</p>
-vmargs	<p data-bbox="610 1073 1370 1136">Use this parameter to specify the Java maximum heap size for the Java process that controls the command line playback.</p> <p data-bbox="610 1178 1269 1209">You must use the -vmargs parameter with the <code>-Xmx</code> argument.</p> <p data-bbox="610 1241 1386 1304">For example, when you use -vmargs -Xmx4096m, specifies the maximum heap size as 4Gb.</p> <p data-bbox="610 1346 1424 1451">This method is similar to specifying <code>-Xmx4096m</code> in the <code>eclipse.ini</code> file for Rational® Performance Tester when playing back the test from the user interface.</p> <p data-bbox="610 1493 750 1524">For example,</p> <pre data-bbox="625 1541 1377 1619">cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "proj123" -suite "test1.testsuite" -vmargs "-Xmx4096m"</pre>

Running a test or schedule from a command line

You can run a test or schedule without opening Rational® Performance Tester by using the command-line interface (CLI).

Before you begin

- You must have read and been familiar with command line parameters. See [Command-line parameters on page 375](#).
- You must have completed the following tasks:
 - Recorded the tests in Rational® Performance Tester.
 - Set the **Web Reports** preferences in Rational® Performance Tester to view the reports remotely. See [Accessing reports remotely on page 468](#).
 - **Optional:** Created a configuration file to use the **-configfile** parameter. See [Creating a command-line config file on page 370](#).
 - **Optional:** Added password of encrypted datasets to secure storage. See [Activating secure storage of dataset passwords on page 371](#).
 - Closed Rational® Performance Tester.

About this task

The command-line syntax with the `checkmark_perf.jpg` parameters is as follows:

```
cmdline -workspace <workspace_full_path> -project <proj_name or proj_relative_path> -eclipsehome
<eclipse_full_path> -plugins <plugin_full_path> -schedule <sched_name or sched_rel_path> -suite <suite_name
or suite_relative_path> -importzip <full_path_to_zip_file> -varfile <variable_file_full_path> -configfile
<file_full_path> -results <result_file> -overwrite <"true" | "false"> -quiet -users <number_of_users> -vmargs
<JVM_args> -rate <RateRunnerGroupName=iterationNumber1/duration, iterationNumber2/duration> -duration
<Stage1=durationOfStage; Stage2=durationOfStage> -execsummaryreport <id_of_report> -stdout -publish
<serverURL>#project.name=<projectName>&teamspace.name=<name_of_the_teamspace> -publish_for
<ALL,PASS,FAIL,ERROR, INCONCLUSIVE> -labels <labelname1, labelname2> -exportlog <log_full_path> -exportstats
<local_dir_path> -exportstatshhtml <local_dir_path> -exportstatsformat <name of the file format> -compare
<"result_path1, result_path2"> -exportstatreportlist <stats_list> -usercomments <"any user comment"> -publishreports
<"STATS, TESTLOG"> -swapdatsets <existing_dataset_file_path:new_dataset_file_path> -timerange <"all, 5 Users, 10
Users"> -history <jaeger,testlog,null> -overridermLabels <"rmlabelname1,rmlabelname2"> -ignoreunhealthytranscations
<"true" | "false"> -execsummary <path_to_local_dir>
```

1. Open the command prompt.
2. Run the following command to change the directory to the `cmdline` directory:

```
cd <cmdline_directory_path>
```

For example,

```
cd C:\Program Files\IBM\SDP\cmdline
```

Based on the operating system that you used to install Rational® Performance Tester, you can locate the `cmdline` directory as follows:

Operating system	The default path to the directory
Windows®	C:\Program Files\IBM\SDP\cmdline
Linux™	/opt/IBM/SDP/cmdline
Mac	/Application/IBM/SDP/cmdline

3. Run the following command to run the test:

```
cmdline -workspace workspace_full_path -project proj_rel_path -suite suite_name or suite_rel_path
```

For example,

```
cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "/myproj/proj123" -suite "test1.testsuite"
```

OR

```
cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -project "/myproj/proj123" -suite "Tests/test1.testsuite"
```



Note: You can add an additional parameter to the preceding command according to your testing requirements and then run the test.

Result

The test run starts and the status is displayed on CLI.

Results

You have run the test from CLI.

What to do next

You can analyze the test results from Rational® Performance Tester. See [Evaluating results in web analytic reports on page 437](#).

If you have published test results to Rational® Test Automation Server, then refer to the [Rational® Test Automation Server](#) documentation.

Running multiple tests from the command line

When you want to run multiple tests, schedules or compound tests that are in a project folder, you can select and run those tests by using any of the command options such as "*" or testType from the command line.

Before you begin

You must have completed the following tasks:

- Recorded multiple tests or created multiple compound tests that are contained in your project folder.
- Read and been familiar with the command-line arguments to use with the cmdline command. See [Command-line parameters on page 375](#).

1. Navigate to the directory that contains the `cmdline.bat` or `cmdline.sh` file.
2. Enter one the following commands to run multiple tests:

```
cmdline -workspace <workspacename> -project <projectname> -suite "*"
```

```
cmdline -workspace <workspacename> -project <projectname> -testType <testtype_value>
```

Where,

- "*" is the wildcard character enclosed within double quotes that denotes all test assets in the specified project folder.
- <testtype_value> indicates the test type to be selected for test run. Rational® Service Tester for SOA Quality supports **test** and **compound** as <testtype_value>. You can specify the value as **test** for all individual tests such as Web UI, mobile, HTTP, and Windows tests. For all compound tests, you can specify the value as **compound**.

You can also use the exclude option "!" along with the wildcard option "*" to exclude the specified test assets.

The following table lists the sample commands to run multiple tests.

Option	Sample command	Description
Wildcard character '*'	<code>cmdline -workspace "workspace" -project myProject -suite "*"</code>	This command runs all the tests and schedules in myProject .
	<code>cmdline -workspace "workspace" -project myProject -suite "test*"</code>	This command runs all the tests in myProject that have the file names starting with 'test'.
	<code>cmdline -workspace "workspace" -project myProject -suite "Compound Tests/*"</code>	This command runs all the tests in the Compound Tests folder under myProject .
	<code>cmdline -workspace "workspace" -project myProject -suite "Web*.testsuite:/myProject2/Test1*.testsuite"</code>	This command runs all the tests in myProject that have the file names starting with 'Web' and all the tests in myProject2 that have the file names starting with 'Test1'.
	<code>cmdline -workspace "workspace" -project "PT1" -schedules "sch1*"</code>	This command runs the schedules that have schedules starting with 'sch1'
Wildcard character '*' with '!' to exclude a specified value	<code>cmdline -workspace "workspace" -project myProject -suite "Web*.testsuite:!Web2"</code>	This command runs all the tests in myProject that have the file names starting with 'Web', but not 'Web2'.

Option	Sample command	Description
testType	cmdline -workspace "workspace" -project myProject -testType compound	This command runs all the compound tests in myProject .
	cmdline -workspace "workspace" -project myProject -testType test	This command runs all the Web UI, HTTP, mobile and Windows tests in myProject .

Results

You have run multiple tests that are contained in your project folder by using the wildcard or testType option from the command line.

Stopping the test run from a command line

After you run the test from the command-line interface (CLI), you can stop the test run to use additional parameters in your test run.

Before you begin

You must have initiated a test run from CLI. See [Running a test or schedule from a command line on page 395](#).

1. Open the command prompt.
2. Run the following command to change the directory to the `cmdline` directory:

```
cd <cmdline_directory_path>
```

For example,

```
cd C:\Program Files\IBM\SDP\cmdline
```

Based on the operating system that you used to install Rational® Performance Tester, you can locate the `cmdline` directory as follows:

Operating system	The default path to the directory
Windows®	C:\Program Files\IBM\SDP\cmdline
Linux™	/opt/IBM/SDP/cmdline
Mac	/Application/IBM/SDP/cmdline

3. Run the following command to stop the test run after a while:

```
cmdline -workspace workspace_full_path -stoprun
```

For example,

```
cmdline -workspace "C:/Users/IBM/rationalSDP/workspace" -stoprun
```

Result

The test run stops after a few seconds of execution of the command and the status is displayed on CLI. The test log is generated and you can view it from Rational® Performance Tester.



Note: If you have used the **-exportlog** parameter when running the test, the test logs are stored in the specified location and also in Rational® Performance Tester.

4. Run the following command to stop the test run immediately:

```
cmdline -workspace workspace_full_path -abandonrun
```

For example,

```
cmdline -workspace "C:/Users/IBM/rationalsdp/workspace" -abandonrun
```

Result

The test run stops immediately and the status is displayed on CLI. However, the test log is not generated.

Results

You have stopped the test run from CLI.

Related information

[Viewing test logs on page 467](#)

Controlling cache sizes

If you use an infinite loop and the number of cached responses in a test increases exponentially, you can set a limit to cache for a user group in the schedule.

About this task

When the cache limit is reached, the least-recently accessed cached entry is released to accommodate a new entry. Also, when a test follows another test in the schedule, you can clear the cache before a test starts.

1. To clear the cache before a test starts, from the Test Navigator, open a test.
2. Click the **HTTP Options** tab and select the **Clear page cache when the test starts** check box.
3. To set a limit to the number of cache entries, in the Test Navigator, navigate to a schedule and double-click it to open it.
4. Click the user group for which you want set the cache limit.
5. Click the **Options** tab and then click **Edit Options**.
6. Select the **Set cache size limit** check box and, in the **Maximum cache size** field, type a numeric value.
This value indicates the number of entries allowed for a user.
7. Click **OK** and save the schedule.

Increasing memory allocation

The virtual users that access your web server require memory to prepare requests, send requests, and receive responses. Because the amount of memory is not automatically set on remote computers, you might receive an out-of-memory error. To correct this situation, increase the memory allocation for that computer.

About this task

If you receive an out-of-memory error when you run a test or schedule, override the default amount of memory that is allocated for that computer. To do this, set the RPT_VMARGS property, which overrides RPT_DEFAULT_MEMORY_SIZE. After the first successful execution, IBM® Rational® Performance Tester automatically sets value for RPT_DEFAULT_MEMORY_SIZE, which represents the maximum heap that will be specified by Rational® Performance Tester in subsequent executions.



Note: Ensure there is at least one successful execution after all locations are created so RPT_DEFAULT_MEMORY_SIZE exists.



Tip:

If you see out-of-memory issues, it is a good practice to first check the `javacore*` file. You can also look at the results and verify that the server is responding correctly because many times errors can lead to excessive resource consumption. You can also monitor memory usage with Task Manager or other tools at varying user load levels such as 10, 50, 100, 500 or 1000 users and use that data to make an estimate of the memory needs per virtual user and then project memory requirements for larger user loads. In some cases the best solution is to add another agent.

Rational® Performance Tester sets heap size for RPT_DEFAULT_MEMORY_SIZE based on the bit-type of the JRE:

- For 32-bit Java Runtime Environment (JREs), Rational® Performance Tester sets 70% of the size of physical memory to RPT_DEFAULT_MEMORY_SIZE. Typically, the maximum limit is set to 1200m.
- For 64-bit JREs, some workloads might perform better with a lesser heap size than 70% of physical memory up to a maximum of 12000m.

To increase the memory allocation on a remote computer:

1. In the Test Navigator (from your local computer), expand the project until you find the deployment location that you want to change.

Deployment locations are represented by the  icon.

2. Right-click the deployment location, and then click **Open**.
3. Under **Property Groups**, click the **General Properties** link, and then click **Add**.
4. In the New Property window:

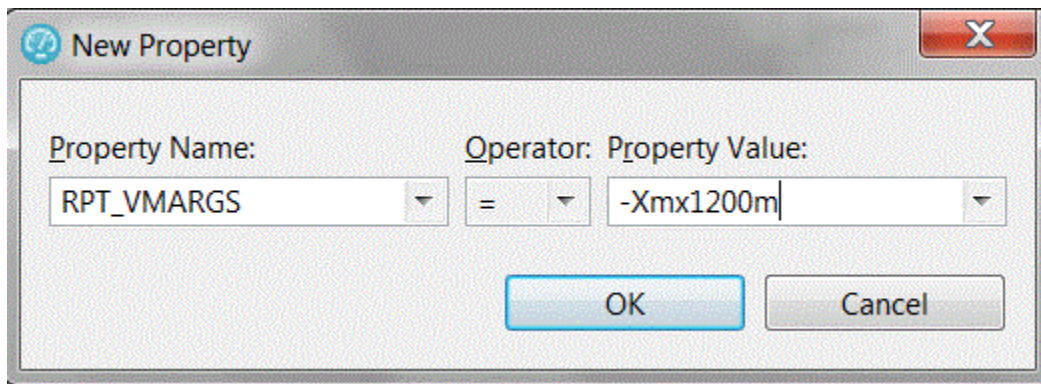
- a. In the **Property Name** field, type `RPT_VMARGS`.
- b. In the **Operator** field, confirm that the operator is `=`.
- c. In the **Property Value** field, type `-Xmxnnnm`, where `nnn` is the amount of memory, in megabytes, and then click **OK**.

Example

If you need to set multiple `RPT_VMARGS` values for a location, place them in the same property entry and separate them with a space. Do not use multiple property entries to set multiple `RPT_VMARGS` values for a location.

Result

The following **New Property** window sets maximum heap to 1200 megabytes:



- i** **Tip:** It is a good practice is to monitor memory usage with Task Manager or other tools at varying user load levels such as 10, 50, 100, 500 or 1000 users and use that data to make an estimate of the memory needs per virtual user and then project memory requirements for larger user loads. In some cases the best solution is to add another agent.

What to do next

If you have increased the available memory and you still receive out-of-memory errors, add more remote computers for your user groups. For information about how to do this, see [Running a user group at a remote location](#).

Extending test execution with custom code

You can extend how you run your tests by writing custom Java™ code and calling the code from the test. You can also specify that results from the tests that are affected by your custom code be included in reports.

Creating custom Java™ code

Custom code uses references in the test as input and returns modified values to the test. Use the `ICustomCode2` interface to create custom code and the `ITestExecutionServices` interface to extend test execution. These interfaces are contained in the `com.ibm.rational.test.lt.kernel.services` package.

About this task



Note: When you use the `ITestExecutionServices` interface in your custom code to report test results, the results for the custom code are displayed in the test log. If you log custom verification point verdicts, these are reflected in the overall schedule verdict.

Custom code input values can be located in references or field references. You can also pass a text string as an argument to custom code. References that are used as input to custom code must be included in the same test as the custom code. In the test, the reference must precede the code that it affects. Verify that the test contains the references that are required for customized inputs to your code. For details about creating references and field references, see [Creating a Reference or a Field Reference](#).

If your custom code uses external JAR files, you might need to change the Java™ build path. In some cases, you can avoid changing the build path manually by running the test before adding your custom code to it. The first time a test runs, classes and libraries that are required for compilation are added to the build path. For example, you can import Test and Performance Tools Platform (TPTP) classes that are required to create custom events in the test log if the test, to which you have added your custom code, has run previously. However, if the test has never been run, import errors occur because the classes are not named in the build path for the project until the test has run.

If your code uses external resources, for example, an SQL database or a product that manages customer relationships, you must configure the custom code to work on every computer on which your test runs.

Custom code is saved in the `src` folder of the project that contains the test that calls the code. By default, custom code is located in a package named `test` in the `src` folder.

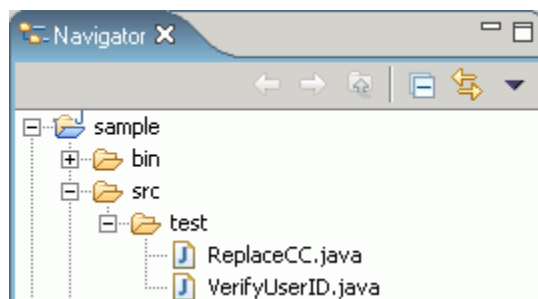
You can reuse a custom code package for tests that are located in multiple projects. The projects must be in one workspace. To reuse custom code across projects, use the project name before the custom code package. For

Test Element Details

Class name:

example,

The following example shows the standard Navigator view of two custom code classes. (The Test Navigator does not display Java™ source files.)



When you add the `ReplaceCC.java` and `VerifyUserID.java` custom code classes to the test and return a value to the test, **Substitute** lists these two classes.

The test package also contains the generated Java™ code for tests in the project.

You can put custom code in a different package (for example, `custom`). Separate custom code from generated code, especially if you use a source-control system.

To add custom code:

1. Open the test, and select a test element.
2. Click **Add** or **Insert**, and select **Custom Code**.
Add appends the custom code to the bottom of the selected test element. **Insert** adds the custom code above the selected test element.



Note: After you add or insert custom code, the Problems view displays an error stating that the new custom code element has no Java™ file. This error message remains until you click **View Code** or **Generate Code**, to remind you that the custom code test element is not yet associated with any Java™ code.

3. Inspect the **Class name** field, and complete one of these steps:
 - If the code to call already exists, change the class name to match its name. Click **View Code** to open the code in the Java™ editor.
 - If the code does not exist, change the class name to describe the purpose of the code. Click **Generate Code** to generate a template class for logging results and to open it in the Java™ editor. If a class with this name exists, you are warned that it will be overwritten.
4. In the **Arguments** field, click **Add**.
5. In the **Custom Code** window, select all inputs that your code requires.
The **Custom Code** window lists all values in the test that can be used as inputs to your code (references or field references in the test that precede the code).
6. Click **OK**.

Result

The window closes, the selected references are added to the **Arguments** field.

7. To add text strings as inputs to your custom code, click **Text**, and then type the text string to use.
8. In the test, after your custom code, locate a value that your code returns to the test.
9. Highlight the value.
10. Right-click the highlighted value, click **Substitute**, and select the class name of your custom code.

Result

The custom code classes that you have added are listed. After you have made your selection, the value to be returned to the test is highlighted in orange, and the **Used by** table is updated with this information.

What to do next

Custom code is not displayed in the **Test Navigator** view. To view custom code, open the **Package Explorer** view and use the Java™ tools to identify the custom code that you added.

Test execution services interfaces and classes

You use the test execution services interfaces and classes to customize how you run tests. These interfaces and classes are located in the `com.ibm.rational.test.it.kernel` package. Each interface and class is described briefly in this topic and in detail in the Javadoc information.

The Javadoc for the test execution services interfaces and classes can be accessed from the product by clicking **Help > Help Contents > IBM Rational Performance Tester API Reference**.

Interface	Description
ICustom-Code2	Defines customized Java™ code for test execution services. Use this interface to create all custom code.
ITestExecutionServices	<p>Provides information for adding custom test execution features to tests. Replaces the IKLog interface. All the methods that were available in IKLog are contained in ITestExecutionServices, along with several newly exposed objects and interfaces. This interface is the primary interface for execution services. ITestExecutionServices contains the following interfaces:</p> <ul style="list-style-type: none"> <li data-bbox="380 1020 500 1041">• IDataArea <li data-bbox="380 1083 456 1104">• IARM <li data-bbox="380 1146 591 1167">• IDatasetController <li data-bbox="380 1209 532 1230">• ILoopControl <li data-bbox="380 1272 565 1293">• IPDLogManager <li data-bbox="380 1335 607 1356">• IStatisticsManager2 <li data-bbox="380 1398 581 1419">• ITestLogManager <li data-bbox="380 1461 456 1482">• ITime <li data-bbox="380 1524 532 1545">• ITransaction <li data-bbox="380 1587 456 1608">• String
IDataArea	Defines methods for storing and accessing objects in data areas. A data area is a container that holds objects. The elements of a data area are similar to program variables and are scoped to the owning container. To use objects specific to a protocol, you should use objects provided by that protocol that are stored in the protocol-specific data area.

Interface	Description
IARM	Provides information about defining ARM (Application Response Measurement) specifications. You use this interface if your virtual users are being sampled for ARM processing.
ILoop- Control	Provides control over loops in a test or schedule. For example, you can use this interface to break loops at specific points in a test. The loop that is affected is the nearest containing loop found in either the test or the schedule.
IPDLog- Manager	Provides logging information such as problem severity, location levels, and error messages.
IStatistic- sManag- er	Provides access to performance counters in the ICustomCode2 interface (used for defining custom code). Performance counters are stored in a hierarchy of counters. Periodically, all the counter values in the hierarchy are reported to the testing workbench and collected into test run results, where they are available for use in reports and graphs. Each counter in the hierarchy has a type (defined in class <code>StatType</code>). The operations that are available on a counter depend on the counter's type.
ITestLog- Manager	Logs messages and verification points to the test log. Use this interface for handling error conditions, anomalies in expected data or other abstract conditions that need to be reported to users, or for comparisons or verifications whose outcome is reported to the test log. ITestLogManager can also convey informational or status messages after the completion of a test.
ITime	Defines basic time services, such as the current system time in milliseconds (adjusted so that all systems are synchronized with the schedule controller), the time the test begins, and the elapsed time from the beginning of the test.
ITransac- tion	Provides support for transactions. A collection of named transactions is maintained for each virtual user. Transactions created in custom code can be started and stopped wherever custom code can be used. These transactions can span several tests. Performance counters are kept for custom code transactions and appear in reports. An example of how you could use ITransaction is to create transactions for one virtual user but not another, to help verify responses from tests.
IEngineIn- fo	Provides information about the testing execution engine; for example, the number of virtual users running in this engine, the number of virtual users that have completed, the local directory in which test assets are deployed, and the host name of the computer on which the engine runs.
ITestInfo	Provides information about the test that is running; for example, the test name and information about the current problem determination log level for this test.
IVirtual- UserInfo	Provides information about virtual users; for example, the virtual user's name, problem determination log level, TestLog level, globally unique ID, and user group name.
IScalar	Provides methods for simple integer performance counters. It is used for counters of <code>SCALAR</code> and <code>STATIC</code> types. Use this interface to decrement and increment counters.
IStat	Defines observational performance counters. It defines the method for submitting a data point to performance counters of type <code>RATE</code> , <code>AVERAGE</code> , and <code>RANGE</code> .

Interface	Description
IStat-	Retrieves the performance counter tree associated with the current statistics processor. Stops the deliv-
tics	ery of performance counters. Changes the priority of the statistics delivery thread.
IStatTree	Provides methods that can retrieve child counters, create the XML fragments that define counters, and set the description field of counters.
IText	Contains text-based performance counters. Performance counters that do not fit any of the other counter types can be created as type <code>TEXT</code> . <code>TEXT</code> counters are not assigned definitions, but they are collected in the test results.

Class	Description
Data-	Throws an exception whenever an attempt is made to modify a locked DataArea key.
Area-	
Lock-	
Ex-	
cep-	
tion	
Out-	Indicates that an object created by <code>ITestExecutionServices</code> has been referenced outside of its intended
OfS-	scope.
cope-	
Ex-	
cep-	
tion	
Trans-	Throws an exception when a transaction is misused. The following conditions lead to a <code>TransactionException</code>
ac-	exception: attempting to start a transaction that has already been started, attempting to stop a transaction
tion-	that has not been started, and getting the start time or the elapsed time of a transaction that has not been
Ex-	started. Any operation (except <code>abort()</code>) on a transaction that has been aborted will throw a <code>TransactionExcep-</code>
cep-	<code>tion</code> exception.
tion	
Stat-	Provides a list of valid performance counter types. The performance counter types are: <code>AVERAGE</code> , <code>iAVERAGE</code> ,
Type	<code>iRANGE</code> , <code>iRATE</code> , <code>iSCALAR</code> , <code>iSTATIC</code> , <code>iSTRUCTURE</code> , <code>iTEXT</code> , <code>RANGE</code> , <code>RATE</code> , <code>SCALAR</code> , <code>STATIC</code> , <code>STRUCTURE</code> , and <code>TEXT</code> .

Reducing the performance impact of custom code

If custom code runs inside a page, it can affect that page's response time.

HTTP pages are containers of HTTP requests. On a given HTTP page, requests run in parallel across all of the connections between the agent computer and the system under test.

Page response time is the interval between *page start* and *page end*, which are defined as follows: Page start is the first timestamp associated with the client-server interaction. This interaction is either the first byte sent or the first connect of the first HTTP request. Page end is the last timestamp associated with the client-server interaction. This interaction is the last byte received of the last HTTP request to complete. Because of parallelism, the last HTTP request to complete might not be the last one listed for the page.

Typically, you should not insert custom code inside a page. While custom code that runs for only a few milliseconds should have little effect on page response time, the best practice is to place custom code outside a page. Custom code placed outside a page has no effect on page response time, and its execution time can overlap with think time delays.

Do not use custom code for data correlation if you can instead use the data correlation features built into the product. The built-in data correlation code takes advantage of requests running in parallel, whereas custom code actions do not begin until all earlier actions are completed.

You might need to place custom code inside a page to correlate a string from the response of a request inside that page to another request inside the same page. Even in this case, if you split the page into two pages, you can use the built-in data correlation features instead of custom code.

If you still want to run tests with custom code inside HTTP pages, use the Page Element report to evaluate performance. The Page Element report shows the response time and throughput for individual HTTP requests. Custom code does not affect the response time measurement of individual HTTP requests.

Related information

Performance testing tips

Custom code examples

Custom code enables you to perform such tasks as managing loops, retrieving virtual user information, running external programs from tests, and customizing data correlation.

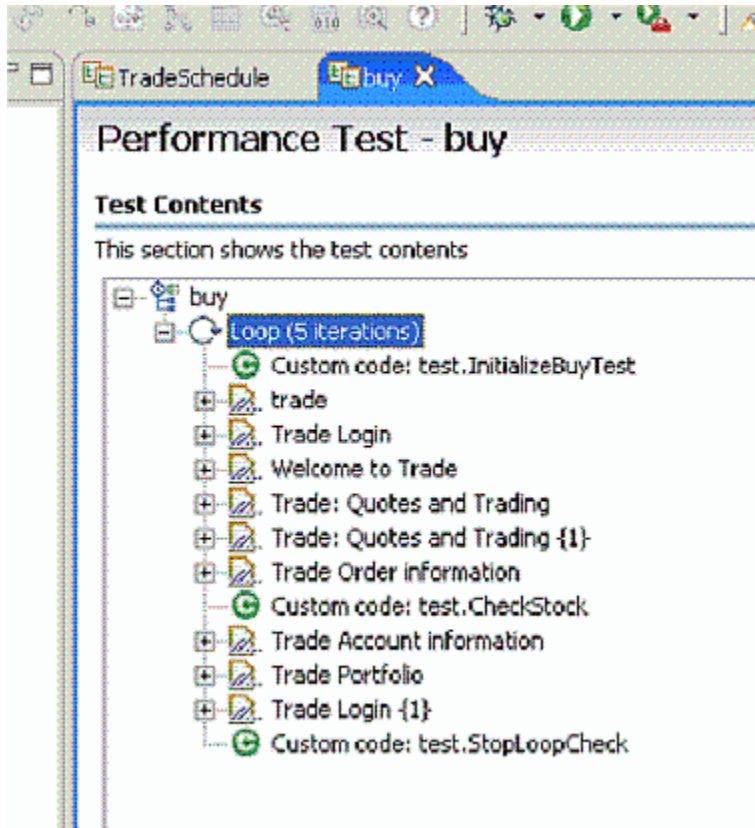
Controlling loops

This example demonstrates extending test execution by using custom code to control loops. It provides sample code that shows how you can manipulate the behavior of loops within a test to better analyze and verify test results.

The Javadoc for the test execution services interfaces and classes can be accessed from the product by clicking **Help > Help Contents > IBM Rational Performance Tester API Reference**.

This example uses a recording of a stock purchase transaction using the Trade application. The concepts shown here can be used in tests of other applications.

The test begins with a recording of a stock purchase transaction, using dataset substitution for the login IDs. The pages are wrapped in a five-iteration loop, as shown in the following figure:



Notice that among the various pages of the test, three items of custom code exist (indicated by the green circles with "C"s in them). This example explores these items of custom code.

The Javadoc for the test execution services interfaces and classes can be accessed from the product by clicking **Help > Help Contents > IBM Rational Performance Tester API Reference**.

The first piece of custom code, `InitializeBuyTest`, is mentioned here:

```
package customcode;

import java.util.Random;

import com.ibm.rational.test.lt.kernel.IDataArea;
import com.ibm.rational.test.lt.kernel.services.ITestExecutionServices;
import com.ibm.rational.test.lt.kernel.services.IVirtualUserInfo;

/**
 * @author unknown
 */
public class InitializeBuyTest implements
    com.ibm.rational.test.lt.kernel.custom.ICustomCode2 {

    /**
     * Instances of this will be created using the no-arg constructor.
     */
    public InitializeBuyTest() {
```

```

}

/**
 * For description of ICustomCode2 and ITestExecutionServices interfaces,
 * see the Javadoc.. */
public String exec(ITestExecutionServices tes, String[] args) {
    // Get the test's data area and set a flag indicating that nothing
    // has failed yet. This flag will be used later to break out
    // of the schedule loop as soon as a failure is encountered.
    IDataArea dataArea = tes.findDataArea(IDataArea.TEST);
    dataArea.put("failedYet", "false");

    // Get the virtual users's data area
    IDataArea vda = tes.findDataArea(IDataArea.VIRTUALUSER);

    // Randomly select a stock to purchase from the set of s:0 to s:499.
    IVirtualUserInfo vuInfo = (IVirtualUserInfo) vda.get(IVirtualUserInfo.KEY);
    Random rand = vuInfo.getRandom();
    String stock = "s:" + Integer.toString(rand.nextInt(499));

    // Persist the name of the stock in the virtual user's data area.
    vda.put("myStock", stock);

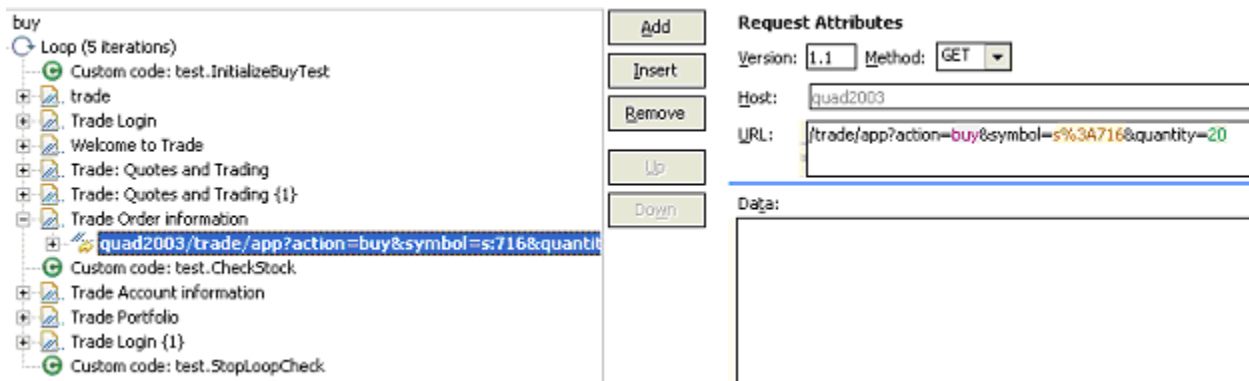
    return stock;
}

```

This custom code is located in the method `exec()`.

First, the data area for the test is acquired to store a flag value, in this case a string of text, to be used later to stop the test loop when an error is discovered. Data stored in this way can be persisted across tests.

Then a randomly generated stock string is created. The value is stored as the variable `stock`, and is passed back as the return value for the method. This return value is used as a substitute in a request later, as shown in the following figure:



The highlighted item uses a substitution (`s:716`), which is the value returned by the `InitializeBuyTest` custom code item. We are using custom code to drive the direction of our test.

The next lines of code in `InitializeBuyTest` use the Virtual User data area to store the name of the stock for later reference. Again, data stored in this way can persist across tests.

The second piece of custom code is called `checkStock`. Its contents are as follows (listing only the `exec()` method this time):

```
public String exec(ITestExecutionServices tes, String[] args) {

    // Get the actual and requested stock purchased.
    String actualStock = args[0].replaceAll("<B>", "");
    actualStock = actualStock.substring(0, actualStock.indexOf("<"));
    String requestedStock = args[1];

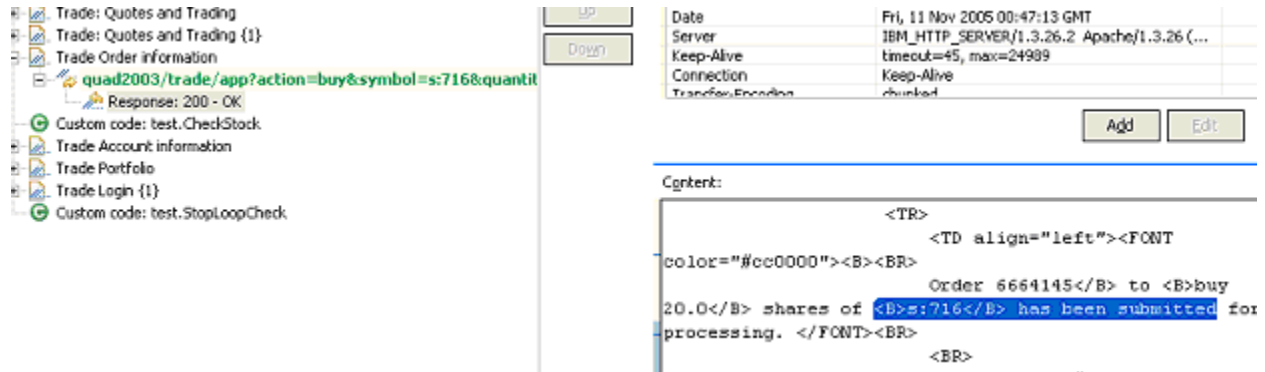
    // Set the log level to ALL.
    IDataArea dataArea = tes.findDataArea(IDataArea.TEST);
    ITestInfo testInfo = (ITestInfo)dataArea.get(ITestInfo.KEY);
    testInfo.setTestLogLevel(ITestLogManager.ALL);

    // If the log level is set to ALL, report the actual and requested stock
    // purchased.
    ITestLogManager testLogManager = tes.getTestLogManager();
    if (testLogManager.wouldReport(ITestLogManager.ALL)) {
        testLogManager.reportMessage("Actual stock purchased: "
            + actualStock + ". Requested stock: " + requestedStock
            + ".");
    }

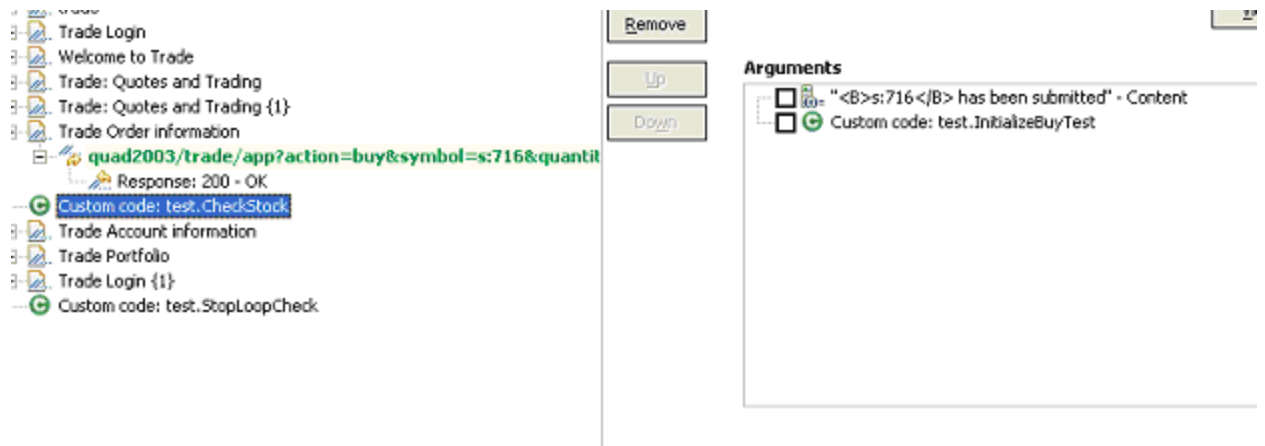
    // If the actual and requested stock don't match, submit a FAIL verdict.
    if (testLogManager.wouldReport(ITestLogManager.ALL)) {
        if (!actualStock.equalsIgnoreCase(requestedStock)) {
            testLogManager.reportVerdict(
                "Actual and requested purchase stock do not match.",
                VerdictEvent.VERDICT_FAIL);

            // Use the test's data area to record the fact that an error has
            // occurred.
            dataArea.put("failedYet", "true");
        }
    }
    return null;
}
```

This code begins by extracting two arguments that have been passed to the code. A part of the response in the original recording is highlighted and used as a reference, as shown in the following figure.



Some string manipulation is needed to acquire the text of interest; in this case, the name of the stock that was actually purchased. This newly created reference is then passed into `CheckStock` as an argument, as shown in the following figure:



Note that the return value of `InitializeBuyTest` is passed in as an argument as well.

The `CheckStock` custom code item uses these values to verify that the randomly chosen stock generated by `InitializeBuyTest` is actually purchased during the execution of the test.

`CheckStock` then sets the test log level, reports the actual and requested stock purchase, and raises a FAIL verdict if they do not match. `CheckStock` also stores a `true` value associated with the tag `failedYet` in the test's data area.

The third piece of custom code (`exec()` method only) is mentioned here:

```
public String exec(ITestExecutionServices tes, String[] args) {

    // Get the test log manager.
    ITestLogManager testLogManager = tes.getTestLogManager();

    // Get the test's data area and get a flag indicating to
    // see if anything has failed yet. If so, stop the loop.
    IDataArea dataArea = tes.findDataArea(IDataArea.TEST);
    String failedYet = (String) dataArea.get("failedYet");

    // Break out of the loop if an error has been encountered.
```

```

if (failedYet.equalsIgnoreCase("true")) {
    tes.getLoopControl().breakLoop();

    if (testLogManager.wouldReport(ITestLogManager.ALL)) {
        testLogManager.reportMessage("Loop stopped.");
    }
}

return null;
}

```

This code uses the test's data area to determine the user-defined value associated with the tag `failedYet`. If `failedYet` is `true`, `StopLoopCheck` breaks out of the test loop.

Retrieving the IP address of a virtual user

This example shows how to retrieve the local IP address of a virtual user. Retrieving IP addresses is particularly useful when virtual users are using IP aliases.

The Javadoc for the test execution services interfaces and classes can be accessed from the product by clicking **Help > Help Contents > IBM Rational Performance Tester API Reference**.

The following custom code retrieves the IP address that was assigned to a virtual user:

```

import java.net.InetAddress;
import com.ibm.rational.test.lt.kernel.IDataArea;
import com.ibm.rational.test.lt.kernel.services.ITestLogManager;
import com.ibm.rational.test.lt.kernel.services.IVirtualUserInfo;

public String exec(ITestExecutionServices tes, String[] args) {
    IVirtualUserInfo vui = (IVirtualUserInfo)
        tes.findDataArea(IDataArea.VIRTUALUSER).get(IVirtualUserInfo.KEY);
    ITestLogManager tlm = tes.getTestLogManager();

    if (vui != null) {
        String localAddr = null;
        InetAddress ipAddr = vui.getIPAddress();
        if (ipAddr != null)
            localAddr = ipAddr.toString();
        tlm.reportMessage("IPAlias address is " + (localAddr != null ? localAddr : "not set"));
        return localAddr;
    }
    else
        return ("Virtual User Info not found");
}

```

Printing input arguments to a file

The `PrintArgs` class prints its input arguments to the file `C:\arguments.out`. This class could be used, for example, to print a response returned by the server.

The Javadoc for the test execution services interfaces and classes can be accessed from the product by clicking **Help > Help Contents > IBM Rational Performance Tester API Reference**.

Example

```

package customcode;

import com.ibm.rational.test.lt.kernel.services.ITestExecutionServices;

import java.io.*;

/**
 * The PrintArgs class prints its input arguments to the file
 * C:\arguments.out. This example could be used to print a response
 * returned by the server.
 */

/**
 * @author IBM Custom Code Samples
 */

public class PrintArgs implements
    com.ibm.rational.test.lt.kernel.custom.ICustomCode2 {

    /**
     * Instances of this will be created using the no-arg constructor.
     */
    public PrintArgs() {
    }

    public String exec(ITestExecutionServices tes, String[] args) {
        try {
            FileWriter outFile = new FileWriter("C:\\arguments.out");
            for (int i = 0; i < args.length; i++)
                outFile.write("Argument " + i + " is: " + args[i] + "\n");
            outFile.close();
        } catch (IOException e) {
            tes.getTestLogManager().reportMessage(
                "Unable to write to C:\\arguments.out");
        }
        return null;
    }
}

```

Counting the number of times that code is executed

The CountAllIterations class counts the number of times code is executed by all virtual users. The CountUserIterations class counts the number of times code is executed by an individual virtual user.

The Javadoc for the test execution services interfaces and classes can be accessed from the product by clicking **Help > Help Contents > IBM Rational Performance Tester API Reference**.

Example

The CountAllIterations class counts the number of times it is executed by all virtual users running in a particular JVM and returns this count as a string.

```

package customcode;

import com.ibm.rational.test.lt.kernel.services.ITestExecutionServices;

/**
 * The CountAllIterations class counts the number of times it is executed
 * by all virtual users running in a particular JVM and returns this count
 * as a string.  If all virtual users on an agent are running in the same
 * JVM (as would typically be the case), this class will count the number of
 * times it is run on the agent.
 */

/**
 * @author IBM Custom Code Samples
 */

public class CountAllIterations implements
    com.ibm.rational.test.lt.kernel.custom.ICustomCode2 {
    private static int numJVMLoops = 0;

    /**
     * Instances of this will be created using the no-arg constructor.
     */
    public CountAllIterations() {
    }

    public String exec(ITestExecutionServices tes, String[] args) {
        return Integer.toString(++numJVMLoops);
    }
}

```

Example

The CountUserIterations class counts the number of times code is executed by an individual virtual user.

```

package customcode;

import com.ibm.rational.test.lt.kernel.services.ITestExecutionServices;
import com.ibm.rational.test.lt.kernel.IDataArea;

/**
 * The CountUserIterations class counts the number of times it is executed
 * by an individual virtual user and returns this count as a string.
 */

/**
 * @author IBM Custom Code Samples
 */

public class CountUserIterations implements
    com.ibm.rational.test.lt.kernel.custom.ICustomCode2 {

    /**
     * Instances of this will be created using the no-arg constructor.
     */
    public CountUserIterations() {
    }
}

```

```

public String exec(ITestExecutionServices tes, String[] args) {
    IDataArea userDataArea = tes.findDataArea(IDataArea.VIRTUALUSER);
    final String KEY = "NumberIterationsPerUser";

    Number numPerUser = (Number)userDataArea.get(KEY);
    if (numPerUser == null) {
        numPerUser = new Number();
        userDataArea.put(KEY, numPerUser);
    }

    numPerUser.value++;
    return Integer.toString(numPerUser.value);
}

private class Number {
    public int value = 0;
}
}

```

Setting and clearing cookies for a virtual user

The `SetCookieFixedValue` class sets a Cookie for a virtual user, and the `ClearCookies` class clears all cookies for a virtual user.

The Javadoc for the test execution services interfaces and classes can be accessed from the product by clicking **Help > Help Contents > IBM Rational Performance Tester API Reference**.

Exemple

The `SetCookieFixedValue` class sets a Cookie, defined in the `newCookie` variable, for a virtual user just as if the server had returned a Set-Cookie.

```

package customcode;

import com.ibm.rational.test.lt.kernel.services.ITestExecutionServices;
import com.ibm.rational.test.lt.execution.http.cookie.IHTTPVirtualUserInfo;
import com.ibm.rational.test.lt.kernel.IDataArea;

import java.text.ParseException;

/**
 * The SetCookieFixedValue class sets a Cookie, defined in the newCookie
 * variable, for a virtual user just as if the server had returned a Set-Cookie.
 */

/**
 * @author IBM Custom Code Samples
 */

public class SetCookieFixedValue implements
    com.ibm.rational.test.lt.kernel.custom.ICustomCode2 {

    /**
     * Instances of this will be created using the no-arg constructor.
     */
}

```



```

public SetCookieFixedValue() {
}

public String exec(ITestExecutionServices tes, String[] args) {
    String newCookie = "MyCookie=CookieValue;path=/;domain=.ibm.com";
    IDataArea dataArea = tes.findDataArea(IDataArea.VIRTUALUSER);
    IHTTPVirtualUserInfo httpInfo =
        (IHTTPVirtualUserInfo)dataArea.get(IHTTPVirtualUserInfo.KEY);

    try {
        httpInfo.getCookieCache().setCookie(newCookie);
    } catch (ParseException e) {
        tes.getTestLogManager().reportMessage("Unable to parse Cookie " +
            newCookie);
    }

    return null;
}
}

```

The ClearCookies class clears all Cookies for a virtual user. For information on how cookies are treated in tests and schedules, see How loops affect the state of virtual users.

```

package customcode;

import com.ibm.rational.test.lt.kernel.services.ITestExecutionServices;
import com.ibm.rational.test.lt.execution.http.util.CookieCacheUtil;

/**
 * The ClearCookies class clears all Cookies for a virtual user.
 */

/**
 * @author IBM Custom Code Samples
 */

public class ClearCookies implements
    com.ibm.rational.test.lt.kernel.custom.ICustomCode2 {

    /**
     * Instances of this will be created using the no-arg constructor.
     */
    public ClearCookies() {
    }

    public String exec(ITestExecutionServices tes, String[] args) {
        CookieCacheUtil.clearCookieCache(tes);
        return null;
    }
}

```

Determining where a test is running

The ComputerSpecific class determines where a test is running

The Javadoc for the test execution services interfaces and classes can be accessed from the product by clicking **Help > Help Contents > IBM Rational Performance Tester API Reference**.

Exemple

```

package customcode;

import com.ibm.rational.test.lt.kernel.services.ITestExecutionServices;

import java.net.InetAddress;
import java.net.UnknownHostException;

/**
 * The ComputerSpecific class determined the hostname on which the test is
 * running, prints the hostname and IP address as a message in the test log,
 * and returns different strings based on the hostname.
 */

/**
 * @author IBM Custom Code Samples
 */

public class ComputerSpecific implements
    com.ibm.rational.test.lt.kernel.custom.ICustomCode2 {

    /**
     * Instances of this will be created using the no-arg constructor.
     */
    public ComputerSpecific() {
    }

    public String exec(ITestExecutionServices tes, String[] args) {
        String hostName = "Unknown";
        String hostAddress = "Unknown";

        try {
            hostName = InetAddress.getLocalHost().getHostName();
            hostAddress = InetAddress.getLocalHost().getHostAddress();
        } catch (UnknownHostException e) {
            tes.getTestLogManager().reportMessage(
                "Not able to obtain host information");

            return null;
        }
        tes.getTestLogManager().reportMessage("The hostname is " + hostName +
            "; IP address is " + hostAddress);

        if (hostName.equals("host-1234"))
            return "Special";
        else
            return "Normal";
    }
}

```

Determining where a test is running

The ComputerSpecific class determines where a test is running

The Javadoc for the test execution services interfaces and classes can be accessed from the product by clicking **Help > Help Contents > IBM Rational Performance Tester API Reference**.

Exemple

```

package customcode;

import com.ibm.rational.test.lt.kernel.services.ITestExecutionServices;

import java.net.InetAddress;
import java.net.UnknownHostException;

/**
 * The ComputerSpecific class determined the hostname on which the test is
 * running, prints the hostname and IP address as a message in the test log,
 * and returns different strings based on the hostname.
 */

/**
 * @author IBM Custom Code Samples
 */

public class ComputerSpecific implements
    com.ibm.rational.test.lt.kernel.custom.ICustomCode2 {

    /**
     * Instances of this will be created using the no-arg constructor.
     */
    public ComputerSpecific() {
    }

    public String exec(ITestExecutionServices tes, String[] args) {
        String hostName = "Unknown";
        String hostAddress = "Unknown";

        try {
            hostName = InetAddress.getLocalHost().getHostName();
            hostAddress = InetAddress.getLocalHost().getHostAddress();
        } catch (UnknownHostException e) {
            tes.getTestLogManager().reportMessage(
                "Not able to obtain host information");
            return null;
        }
        tes.getTestLogManager().reportMessage("The hostname is " + hostName +
            "; IP address is " + hostAddress);

        if (hostName.equals("host-1234"))
            return "Special";
        else
            return "Normal";
    }
}

```

Extracting a string or token from its input argument

The ParseResponse class extracts a string from its input argument. The ExtractToken class extracts a particular token (string) from its input argument. Both classes can be useful for handling certain types of dynamic data correlation.

The Javadoc for the test execution services interfaces and classes can be accessed from the product by clicking **Help > Help Contents > IBM Rational Performance Tester API Reference**.

Example

The ParseResponse class extracts a string from its input argument, using a regular expression for pattern matching.

```
package customcode;

import com.ibm.rational.test.lt.kernel.services.ITestExecutionServices;

import java.util.regex.*;

/**
 * The ParseResponse class demonstrates using Custom Code to extract a
 * string from its input argument using a regular expression for pattern
 * matching.
 *
 * In this sample, the args[0] input string is assumed to be the full
 * response from a previous request. This response contains the day's
 * headlines in a format such as:
 *
 * <a class=f href=r/d2>In the News</a><small class=m>&nbsp;<span id=nw>
 * </span></small></h2>
 * <div class=ct>
 * &#149;&nbsp;<a href=s/213231>Cooler weather moving into eastern
 * U.S.</a> * <br>&#149;&nbsp;<a href=s/262502>Digital camera shipments
 * up</a><br> *
 * Given the above response, the extracted string would be:
 *     Cooler weather moving into eastern U.S.
 */

/**
 * @author IBM Custom Code Samples
 */

public class ParseResponse implements
    com.ibm.rational.test.lt.kernel.custom.ICustomCode2 {

    /**
     * Instances of this will be created using the no-arg constructor.
     */
    public ParseResponse() {}

    public String exec(ITestExecutionServices tes, String[] args) {
        String HeadlineStr = "No Headline Available";
        String RegExpStr = ".*In the News[^;]*;[^;]*;[^;]*;<a
href=([>]*)>([<]*)<";
        Pattern pattern =
Pattern.compile(RegExpStr, Pattern.DOTALL);
        Matcher matcher =
pattern.matcher(args[0]);
        if (matcher.lookingAt())
```

```

        HeadlineStr = matcher.group(2);
    else
        tes.getTestLogManager().reportMessage("Input does not match
pattern.");
    return HeadlineStr;
}

```

The ExtractToken class extracts a particular string from its input argument.

```

package customcode;

import com.ibm.rational.test.lt.kernel.services.ITestExecutionServices;

/**
 * The ExtractToken class demonstrates using Custom Code to extract a particular
 * token (string) from its input argument. This can be useful for handling
 * certain types of dynamic data correlation.
 *
 * In this sample, the args[0] input string is assumed to be comma-delimited
 * and the token of interest is the next-to-last token. For example, if
 * args[0] is:
 * javascript:parent.selectItem('1010','[Negative]1010','1010','','IBM',
 * '30181','Rational','1','null','1','1','6fd8e261','RPT')
 * the class will return the string 6fd8e261.
 */

/**
 * @author IBM Custom Code Samples
 */

public class ExtractToken implements
    com.ibm.rational.test.lt.kernel.custom.ICustomCode2 {

    public ExtractToken() {
    }

    public String exec(ITestExecutionServices tes, String[] args) {
        String ArgStr;
        String NextToLastStr;
        String[] Tokens = args[0].split(",");

        if (Tokens.length > 2) {
            ArgStr = Tokens[Tokens.length - 2]; // Extract next-to-last token

            // Remove enclosing ''
            NextToLastStr = ArgStr.substring(1, ArgStr.length() - 1);
        } else {
            tes.getTestLogManager().reportMessage("Could not extract value");
            NextToLastStr = null;
        }
        return NextToLastStr;
    }
}

```

Retrieving the maximum JVM heap size

The JVM_Info class retrieves the maximum heap size of the JVM.

The Javadoc for the test execution services interfaces and classes can be accessed from the product by clicking **Help > Help Contents > IBM Rational Performance Tester API Reference**.

Exemple

```
package customcode;

import com.ibm.rational.test.lt.kernel.services.ITestExecutionServices;

import java.net.*;

/**
 * The JVM_Info class retrieves the maximum heap size of the JVM.
 * It writes a message in the test log with the hostname where the
 * JVM is running and the JVM's maximum heap size in megabytes.
 */

/**
 * @author IBM Custom Code Samples
 */

public class JVM_Info implements
    com.ibm.rational.test.lt.kernel.custom.ICustomCode2 {

    public JVM_Info() {
    }

    public String exec(ITestExecutionServices tes, String[] args) {
        Runtime rt = Runtime.getRuntime();
        long maxMB = rt.maxMemory()/(1024*1024); // maxMemory() size is in bytes
        String hostName = "Unknown";

        try {
            hostName = InetAddress.getLocalHost().getHostName();
        } catch (UnknownHostException e1) {
            tes.getTestLogManager().reportMessage("Can't get hostname");
            return null;
        }

        tes.getTestLogManager().reportMessage("JVM maximum heap size for host "
            + hostName + " is " + maxMB + " MB");

        return null;
    }
}
```

Running an external program from a test

The ExecTest class runs a program, defined in the execName variable, on the system where the test is running.

The Javadoc for the test execution services interfaces and classes can be accessed from the product by clicking **Help > Help Contents > IBM Rational Performance Tester API Reference**.

Exemple

```
package customcode;
```

```

import com.ibm.rational.test.lt.kernel.services.ITestExecutionServices;
import com.ibm.rational.test.lt.kernel.services.ITestLogManager;
import org.eclipse.hyades.test.common.event.VerdictEvent;

import java.io.IOException;

/**
 * The ExecTest class runs a program, defined in the execName variable,
 * on the system where the test is running.
 * The test verdict is set to PASS if the program return code is 0.
 * The test verdict is set to FAIL if the program doesn't execute or
 * if the program return code is non-zero
 * In this sample, the program is perl.exe.
 */

/**
 * @author IBM Custom Code Samples
 */

public class ExecTest implements
    com.ibm.rational.test.lt.kernel.custom.ICustomCode2 {

    /**
     * Instances of this will be created using the no-arg constructor.
     */
    public ExecTest() {
    }

    public String exec(ITestExecutionServices tes, String[] args) {
        ITestLogManager logger = tes.getTestLogManager();
        int rtnval = 1;
        Process p = null;
        String execName = "C:/Windows/System32/perl.exe C:/Perl/true.pl";

        Runtime rt = Runtime.getRuntime();
        // Execute test
        try {
            p = rt.exec(execName);
        } catch (IOException e) {
            logger.reportMessage("Unable to run = " + execName);
            logger.reportVerdict("Execution of " + execName + " failed",
                VerdictEvent.VERDICT_FAIL);

            return null;
        }

        // Wait for the test to complete
        try {
            rtnval = p.waitFor();
            logger.reportMessage("Process return value is " +
                String.valueOf(rtnval));
        } catch (InterruptedException e1) {
            logger.reportMessage("Unable to wait for " + execName);
            logger.reportVerdict("WaitFor on " + execName + " failed",
                VerdictEvent.VERDICT_FAIL);

            return null;
        }
    }
}

```

```

    // Check the test return code and set the test verdict appropriately
    if (rtnval != 0)
    {
        logger.reportVerdict("Execution failed", VerdictEvent.VERDICT_FAIL);
    } else {
        logger.reportVerdict("Execution passed", VerdictEvent.VERDICT_PASS);
    }

    return null;
}
}

```

Adding custom counters to reports

When you want to monitor the specific requirement, you can add custom counters to performance report by using the custom code. After running tests, the results from the custom counters are automatically aggregated in the same way that the default performance testing counters.

Starting from 10.1.0, you can view and monitor the counter information generated by the custom code on a graph when the custom code starts in the test run.

After running tests, you can view the custom counter in the report. You can also view the custom counter information on a different page by creating a custom report. For more information about customizing the report, see related links.

The Javadoc for the test execution services interfaces and classes can be accessed from the product by clicking **Help > Help Contents > IBM Rational Performance Tester API Reference**.

You can add the following custom code in your test to create a custom counter in a report.

```

package test;

import org.eclipse.hyades.test.common.event.VerdictEvent;

import com.ibm.rational.test.lt.kernel.services.ITestExecutionServices;
import com.ibm.rational.test.lt.kernel.services.stats.CountAggregationLevel;
import com.ibm.rational.test.lt.kernel.services.stats.CounterUnits;
import com.ibm.rational.test.lt.kernel.services.stats.ICounterFolder;
import com.ibm.rational.test.lt.kernel.services.stats.ICounterRegistry;
import com.ibm.rational.test.lt.kernel.services.stats.IStatisticsManager2;
import com.ibm.rational.test.lt.kernel.services.stats.IValueCounter;
import com.ibm.rational.test.lt.kernel.services.stats.ValueAggregationLevel;

import database.DatabaseAccess;
import database.TransactionResult;

public class DatabaseStats implements com.ibm.rational.test.lt.kernel.custom.ICustomCode2 {

    private static boolean registerDone;

    /**
     * This method declares the counters that will be produced during execution.
     * Declaring counters is optional, but it allows to customize some of their
     * attributes, such as the label and unit, and what level of statistical information
     * will be available in reports.
     */
}

```



```

*/
private static synchronized void registerCounters(ICounterRegistry registry) {
    if (registerDone) return;
    registry.path("Database", "Transaction", "Attempts")
        .count()
        .aggregationLevel(CountAggregationLevel.RATE_RANGE)
        .label("Started Transactions")
        .unit("transactions")
        .register();

    registry.path("Database", "Transaction", "Commits")
        .verificationPoint()
        .label("Transaction Commits VP")
        .register();

    registry.path("Database", "Transaction", "Response Time", "Network")
        .value()
        .aggregationLevel(ValueAggregationLevel.RANGE)
        .unit(CounterUnits.MILLISECONDS)
        .register();

    registry.path("Database", "Transaction", "Response Time", "Commit")
        .value()
        .aggregationLevel(ValueAggregationLevel.DISTRIBUTION)
        .unit(CounterUnits.MILLISECONDS)
        .register();

    registry.path("Database", "Error")
        .text()
        .label("Database Error Message")
        .register();
    registerDone = true;
}

private DatabaseAccess database = DatabaseAccess.INSTANCE;

/**
 * This custom code adds a record in database. It produces a couple of counters,
 * such as the database transaction attempts, successes/failures, and response time.
 */
public String exec(ITestExecutionServices tes, String[] args) {
    String product = args.length > 0 ? args[0] : "Default";
    IStatisticsManager2 mgr = tes.getStatisticsManager2();
    registerCounters(mgr.registry());

    database.startTransaction();
    mgr.getCountCounter("Database", "Transaction", "Attempts").increment();

    database.executeQuery("INSERT INTO TABLE Purchases VALUES('" + product + "', 1000)");
    TransactionResult result = database.commit();

    mgr.getVerificationPointCounter("Database", "Transaction", "Commits")
        .increment(result.isSuccess() ? VerdictEvent.VERDICT_PASS : VerdictEvent.VERDICT_FAIL);
    if (!result.isSuccess()) {
        mgr.getTextCounter("Database", "Error").addMeasurement(result.getErrorMessage());
    }
}

```

```

ICounterFolder times = mgr.getFolder("Database", "Transaction", "Response Time");
times.getValueCounter("Network").addMeasurement(result.getNetworkTime());
times.getValueCounter("Commit").addMeasurement(result.getCommitTime());

IValueCounter value = tes.getStatisticsManager2().getValueCounter("MyStats", "Value");
value.addMeasurement(System.nanoTime() % 2000);

return null;
}
}

```

Related information

[Creating custom Java code on page 402](#)

[Creating custom reports on page 446](#)

Using transactions and statistics

You can use custom code to start transactions, gather additional statistics during a transaction, and stop a transaction.

The Javadoc for the test execution services interfaces and classes can be accessed from the product by clicking **Help > Help Contents > IBM Rational Performance Tester API Reference**.

The following code shows how to start a transaction. Transactions that are generated by test execution services automatically create and manage statistics.

```

package customcode;

import com.ibm.rational.test.lt.kernel.services.ITestExecutionServices;
import com.ibm.rational.test.lt.kernel.services.ITransaction;

/**
 * @author IBM Custom Code Samples
 */
public class BeginTransaction implements
    com.ibm.rational.test.lt.kernel.custom.ICustomCode2 {

    /**
     * Instances of this will be created using the no-arg constructor.
     */
    public BeginTransaction() {
    }

    /**
     * For Javadoc information on the ICustomCode2 and ITestExecutionServices interfaces,
     * see the 'Test execution services interfaces and classes' help topic.
     */
    public String exec(ITestExecutionServices tes, String[] args) {
        // the name of the transaction could have been passed in via data correlation mechanism.
        ITransaction foo = tes.getTransaction("foo");
        foo.start();
    }
}

```

```

    return null;
}
}

```

The following code shows how to gather additional statistics during a transaction.

```

package customcode;

import com.ibm.rational.test.lt.kernel.ITime;
import com.ibm.rational.test.lt.kernel.services.ITestExecutionServices;
import com.ibm.rational.test.lt.kernel.statistics.IScalar;
import com.ibm.rational.test.lt.kernel.statistics.IStat;
import com.ibm.rational.test.lt.kernel.statistics.IStatTree;
import com.ibm.rational.test.lt.kernel.statistics.impl.StatType;

/**
 * @author IBM Custom Code Samples
 */
public class BodyTransaction implements
    com.ibm.rational.test.lt.kernel.custom.ICustomCode2 {

    /**
     * Instances of this will be created using the no-arg constructor.
     */
    public BodyTransaction() {
    }

    /**
     * For Javadoc information on the ICustomCode2 and ITestExecutionServices interfaces,
     * see the 'Test execution services interfaces and classes' help topic.
     */
    public String exec(ITestExecutionServices tes, String[] args) {
        IStatTree tranStat;
        IStatTree timeStat;
        IStatTree countStat;

        IStat timeDataStat = null; // counter for the time RANGE
        IScalar countDataStat = null; // counter for the count SCALAR

        ITime timer = tes.getTime();

        IStatTree rootStat = tes.getStatisticsManager().getStatTree();
        if (rootStat != null) {
            // these counters set up the hierarchy
            tranStat = rootStat.getStat("Transactions", StatType.STRUCTURE);
            timeStat = tranStat.getStat("Body Time", StatType.STRUCTURE);
            countStat = tranStat.getStat("Bocy Count", StatType.STRUCTURE);

            // the name of the counters could have been passed in via data correlation mechanism
            timeDataStat = (IStat) timeStat.getStat("foo", StatType.RANGE);
            countDataStat = (IScalar) countStat.getStat("foo", StatType.SCALAR);
        }

        // get the start time
        long startTime = timer.timeInTest();

        // do the work
    }
}

```

```

// whatever that work might be

// get the end time
long endTime = timer.timeInTest();

// update timeDataStat with the elapsed time
if (timeDataStat != null)
    timeDataStat.submitDataPoint(endTime - startTime);

// update the countDataStat
if (countDataStat != null)
    countDataStat.increment();

return null;
}
}

```

The following code shows how to stop a transaction.

```

package customcode;

import com.ibm.rational.test.lt.kernel.services.ITestExecutionServices;
import com.ibm.rational.test.lt.kernel.services.ITransaction;

/**
 * @author IBM Custom Code Samples
 */
public class EndTransaction implements
    com.ibm.rational.test.lt.kernel.custom.ICustomCode2 {

    /**
     * Instances of this will be created using the no-arg constructor.
     */
    public EndTransaction() {
    }

    /**
     * For Javadoc information on the ICustomCode2 and ITestExecutionServices interfaces,
     * see the 'Test execution services interfaces and classes' help topic.
     */
    public String exec(ITestExecutionServices tes, String[] args) {
        // the name of the transaction could have been passed in via data correlation mechanism.
        ITransaction foo = tes.getTransaction("foo");
        foo.stop();
        return null;
    }
}

```

Reporting custom verification point failures

You can use custom code to report a custom verification point failure.

The Javadoc for the test execution services interfaces and classes can be accessed from the product by clicking **Help > Help Contents > IBM Rational Performance Tester API Reference**.

The following code shows how to report a custom verification point failure.

```
package customcode;

import org.eclipse.hyades.test.common.event.VerdictEvent;
import org.eclipse.hyades.test.common.runner.model.util.Verdict;

import com.ibm.rational.test.lt.execution.core.IVerificationPoint;
import com.ibm.rational.test.lt.kernel.services.ITestExecutionServices;

/**
 * @author IBM Custom Code Samples
 */
public class Class implements
    com.ibm.rational.test.lt.kernel.custom.ICustomCode2 {

    /**
     * Instances of this will be created using the no-arg constructor.
     */
    public Class() {
    }

    /**
     * For javadoc of ICustomCode2 and ITestExecutionServices interfaces, select 'Help Contents' in the
     * Help menu and select 'Extending
        Rational® Performance
        Tester functionality' -> 'Extending test execution with custom code'
     */
    public String exec(ITestExecutionServices tes, String[] args) {
        tes.getTestLogManager().reportVerificationPoint("CustomVP", VerdictEvent.VERDICT_FAIL);
        return null;
    }
}
}
```

Debugging custom code

This example demonstrates debugging custom code by adding a breakpoint. It provides sample code to add a breakpoint. This way of debugging custom code is applicable only for a schedule.

1. Start IBM® Rational® Performance Tester and create a performance test project `MyProject`.
2. Create an HTTP test, **MyTest**, by recording a visit to `http://<hostname>:7080/`.



Note: Before accessing the URL, ensure that Rational® Performance Tester is running. The URL returns an HTTP 404 error, which is expected.

Result

3. Expand the first request and click the response element.
4. In the Test Element Details section, right-click in the **Content** field and click **Create Field Reference**.
5. Type the reference name and click **OK**.
6. Click the first page, and then click **Add > Custom Code**.
7. In the **Arguments** section of Test Element Details, click **Add**.
8. Expand the data source for the search results page, select the reference name that you created in step 5, and click **Select**.
9. Click **Generate Code**.

Result

A new tab with the generated code is displayed.

10. Insert the following the code into the `exec()` method:

```
ITestLogManager history = tes.getTestLogManager();
if (args.length > 0) {
    if (args[0].indexOf("Investor Relations") != -1) {
        history.reportMessage("First page failed. Bail loop!");
        tes.getLoopControl().continueLoop();
    }
}
```

Important:

- Fix the double quotation marks, if any, so they are straight and the compiler no longer gives warning.
- To resolve compiler warnings related to importing a class, press **Ctrl + Shift + O**.

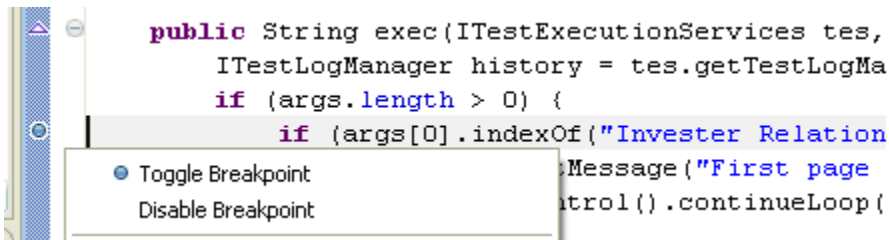
The code will look like this:

```

public String exec(ITestExecutionServices tes, String[] args) {
    ITestLogManager history = tes.getTestLogManager();
    if (args.length > 0) {
        if (args[0].indexOf("Investor Relations") != -1) {
            history.reportMessage("First page failed. Bail loop!");
            tes.getLoopControl().continueLoop();
        }
    }
    return null;
}

```

11. To set a breakpoint, click anywhere on the `args[0].indexOf` line. Move the pointer to the left-most portion of the text editor window and double-click with the pointer horizontally on the same line. A blue button is displayed in this left-most portion of the window indicating the breakpoint is set.



12. Save the custom code and then the test.

13. Create a new schedule, Schtest.

- a. In Schtest, set the number of users to run to 1.
- b. Click **User Group 1** and click **Add > Test**. Select the MyTest test and click **OK**.
- c. Click **User Group 1** and click the **Run this group on the following locations** button.
- d. Click **Add > Add New**.
- e. In the **New Location** window, type the following information:
 - i. In **Host name**, type localhost.
 - ii. In **Name**, type debuglocation.
 - iii. In **Deployment directory**, type C:\mydeploy.
 - iv. Click **Finish**.
- f. Save the schedule.

14. In the Test Navigator, right-click **debuglocation** and click **Open**.

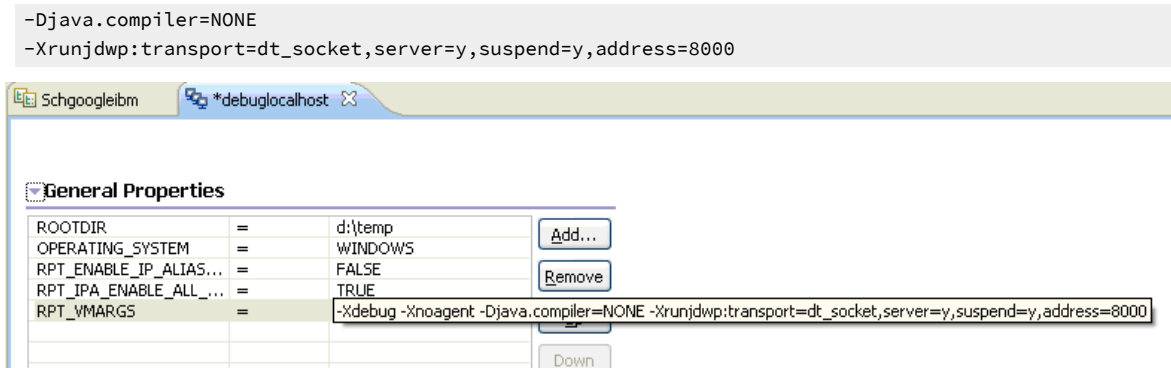
15. Click the **General Properties** tab and click **Add**.

16. In the **Property name** field, type `RPT_VMARGS` and in the **Property value** field, add the following values each separated by a space.

```

-Xdebug
-Xnoagent

```



17. Save the location.

18. Attach the debugger to the schedule execution process.

a. Run the schedule.

Because the schedule is using **debuglocation**, it will pause at the beginning to let you attach the debugger to the execute process.

b. Click **Window > Open Perspective > Other > Debug**.

c. Click **Run > Debug Configurations**.

d. In the **Debug Configurations** window, right-click **Remote Java Application** and click **New**.

e. Click **Debug**.

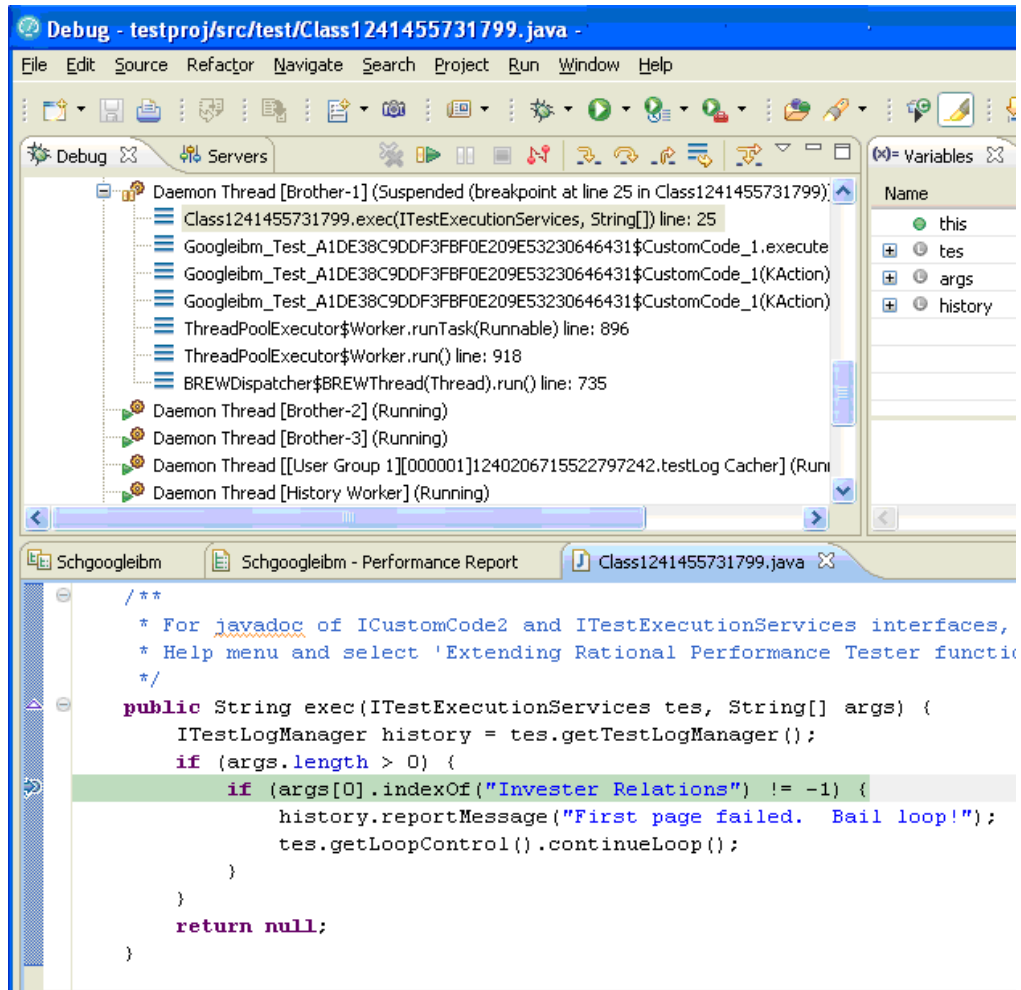
A list of running threads are displayed in the Debug window and the schedule execution pauses at the debug breakpoint.

f. If you are doing it for the first time, you might need to provide the source location to see the custom Java code. You do this by taking the following steps:

i. Click **Edit Source Lookup Path** and click **Add**.

ii. Click **Workspace Folder > OK**.

- iii. Now, expand MyProject, select the src folder, and click **OK**. The schedule run stops at the specified breakpoint.



Accessing the actual schedule name from the custom code

When there are any special characters in the name of the schedule, you can write the actual name of the schedule in the test log by using the appropriate interface in the custom code.

Example

The following sample custom code shows how to write the name of the schedule in the test log:

```
package customcode;

import com.ibm.rational.test.lt.kernel.engine.impl.Engine;
import com.ibm.rational.test.lt.kernel.services.ITestExecutionServices;

/**
 *
 * @author Custom Code Samples
 */
public class GetActualScheduleName implements com.ibm.rational.test.lt.kernel.custom.ICustomCode2 {
```

```

/**
Instances of this will be created using the no-arg constructor.
*/
    public GetActualScheduleName() {
        }
/**

For javadoc of ICustomCode2 and ITestExecutionServices interfaces, select 'Help Contents' in the
Help menu and select 'Extending Rational Performance Tester functionality' -> 'Extending test execution
with custom code'
*/
    public String exec(ITestExecutionServices tes, String[] args) {
        String schName = Engine.getInstance().getSchedule().getScheduleName();
        tes.getTestLogManager().reportMessage("Schedule Name is "+schName);
        return null; }
}

```

Reading and writing data from a dataset

When a test is associated with a dataset, you can extend the test either by reading or writing the dataset values from the custom code.

The data that you write into the dataset is saved only when you set **Open mode** to **Shared (for all test executions)** in the **Edit Dataset** window. In other open modes, the modified data is used only for the test run.

The following sample custom code reads and writes the data from the dataset:

```

package datasets;

import java.awt.List;

/**
 * @author IBM Custom Code Samples
 */

public class myds implements com.ibm.rational.test.lt.kernel.custom.ICustomCode2 {

    public myds() {
    }

    public String exec(ITestExecutionServices tes, String[] args) {
// the name of the dataset is the same as what is shown in the test. The dataset must be added to the
// test in order
// to get a controller for it.
        IDatasetController control = tes.getDataSetController("/testproj/myds.csv");
        try {
// once you have the controller you can get a row
            DataSetRow row = control.getNextRow();
// returns a string representation of the row
            row.getEntireRow();
// alternatively you can get individual values by the column name
            row.getValue("Column1");

// you can also write a new row to the dataset
// -1 means append to the end

```

```

    // alternatively you can specify a row number and whether to overwrite that row or to insert a
    new row at the spot
    control.writeRow(-1, Arrays.asList("a", "b", "c"), false);
  } catch (Exception e) {
    tes.getTestLogManager().alwaysReportMessage( e.toString());
    // TODO Auto-generated catch block
    e.printStackTrace();
  }
  return null;
  // or whatever you want to return here
}

```

Getting the properties of a dataset

When a test asset is associated with a dataset, you can get the properties of a dataset through a custom code. You can also insert a new row into the dataset.

Exemple

The properties that you can get through a custom code are as follows:

- Total number of rows
- Total number of columns
- Name of the columns
- The value that is set for the **Open mode** option
- The value that is set for the **Access mode** option

After the test run is complete, you can view the properties of the dataset in test logs. See [Viewing test logs on page 467](#).

The following sample custom code provides the properties of the dataset in test logs:

```

package datasets;

import java.awt.list;

/**
 * @author IBM Custom Code Samples
 */

public class myds implements com.ibm.rational.test.lt.kernel.custom.ICustomCode2 {

    public myds() {
    }

    public String exec(ITestExecutionServices tes, String[] args) {
        // the name of the dataset is the same as what is shown in the test.
        //The dataset must be added to the test in order to get a controller for it.
        IDatasetController control = tes.getDataSetController("/testproj/myds.csv");
        try
        {

```

```

        tes.getTestLogManager().alwaysReportMessage(" Total Rows in DS
"+(Integer.toString(control.getTotalRows())));
        control.getAccessMode();
        tes.getTestLogManager().reportMessage(" DS Access Mode is "+(control.getAccessMode()));
        tes.getTestLogManager().reportMessage("DS Open Mode is "+(control.getOpenMode()));
        tes.getTestLogManager().alwaysReportMessage("Column headers in DS
"+control.getColumnHeaders().toString());
        control.writeRow(-1, Arrays.asList("a","b","c"), false);
        tes.getTestLogManager().alwaysReportMessage("Total Rows after adding new row in DS
"+Integer.toString(control.getTotalRows()));
    }

    catch (Exception e)
    {
        tes.getTestLogManager().alwaysReportMessage( e.toString());
// TODO Auto-generated catch block
e.printStackTrace();
}

```

Migrating custom code from previous versions

You can run scripts that contain custom code from previous releases and edit tests to make new calls to old or new custom code classes.

About this task

You can perform the following tasks without any additional steps:

- Run a script that contains custom code that was created in a previous release.
- Edit a test to make a new call to an old custom code class.
- Add new custom code to a test that contains old custom code.

To edit a class in existing custom code so that it can call new `TestExecutionServices` methods, type cast the `IKlog` argument in the old custom code to the `ITestExecutionServices` interface.

When you migrate the custom code from the previous versions, you must use `getStatisticsManager2()` as `getStatisticsManager()` API is deprecated from 10.1.0.

Chapter 8. Test Manager Guide

This guide describes how to keep track of the performance of the application by evaluating the test results. This guide is intended for test managers.

After the test or schedule run completes, the performance data is displayed in the results. The results include various reports. You can customize the reports based on your needs.

Evaluating results in web analytic reports

After the test run, evaluate the results in the web analytic reports. Web analytics collect data using new technologies thereby providing better user experience.

Comparing results among runs

To analyze the difference between two or more reports, you can compare them. For example, to analyze the performance of the application at different time slots or different milestone builds, you can compare two runs.

About this task

You can compare the test runs from the Test Navigator view or from the web analytics report itself. To compare test runs from the Test Navigator view, select the runs, right-click, and click **Compare Results**.


You can compare the test runs that are in the same project or in the different projects. When comparing multiple runs, you cannot compare multiple time-ranges or stages.

To compare runs from the web analytics report:

1. Open the run or report to serve as the basis for comparison.

2. Click the name of the run next to the **Menu** option



3. Click **Add**  and navigate to the run to compare with.

Multiple runs are displayed in the report.

4. **Optional:** To add, remove, or move the position of the runs, click **Manage**



Related information



[Comparing schedule stages on page 437](#)

Comparing schedule stages

When you are running a schedule, in this context, is used to refer to VU Schedule and Rate Schedule that contains stages, time ranges are automatically created for each stage. You can view a report that compares these stages, and you also can set preferences to display the report automatically at the end of a staged run.

About this task

In addition to comparing stages, you can add time ranges and compare them. To view the compare report automatically at the completion of a run, click **Window > Preferences > Test > Performance Test Reports**, and select **Launch Compare report when staged run completes**.

1. Open the run that consists of stages.
By default, reports are displayed for the entire run.
2. Click the **Entire Run** menu  and select the stages to compare.
Both the running and completed stages show up in the list.
3. To add a new time range, click **Add**  in the **Entire run** menu.
4. In the **Time Range** dialog, specify a name, start time, and end time of the run and click **Apply**.



Note: When you compare stages in a run, you cannot compare data from various geographies at the same time.

Comparing results from various regions or agent locations

When you run a schedule that includes agents from different regions, use the Web Analytic reports to compare the performance data from these regions.

Before you begin

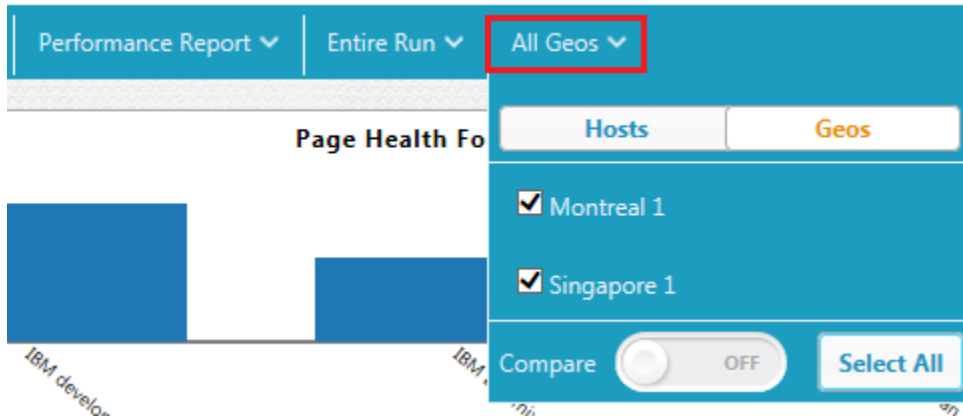
Run a schedule with on premise agents on different geographic regions. See [Running schedules on page 366](#).

About this task

When comparing agents, open the Location asset and in the **General Properties** tab, add an `RPT_GEO` property with any value. This value is then displayed in the **All Geos** menu of the report.

To compare performance data:

1. From the Test Navigator view, open the schedule run that includes remote agents. The name of the run corresponds with the name of the schedule and has a timestamp.
2. On the report toolbar, click **All Geos**.

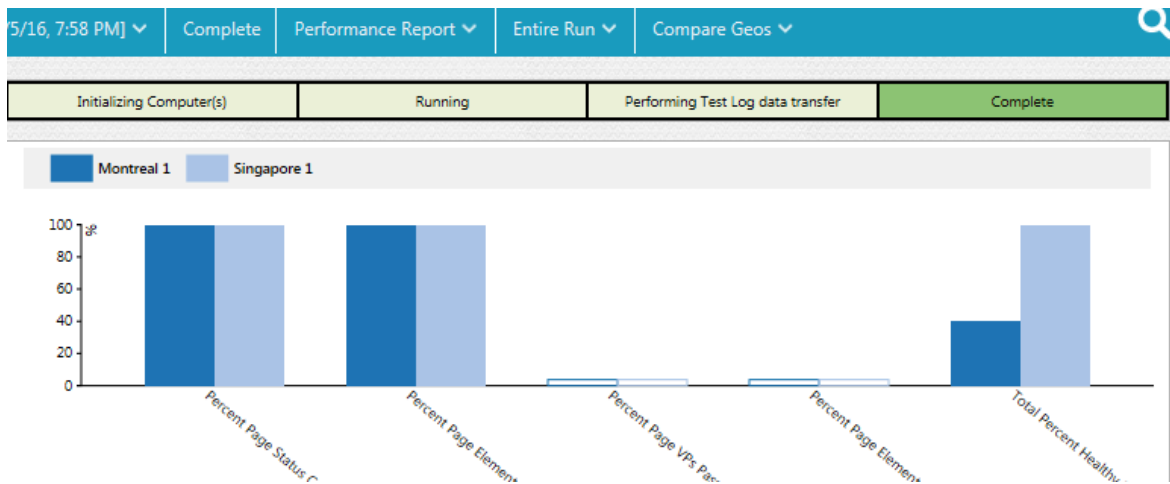


3. Select the regions that you want to compare and click **Compare**.



Note: When you compare data from various regions, you cannot compare stages in a run at the same time.

The report shows the data in the compare mode.



Generating functional reports

You can generate functional reports of your tests, which summarize the pass or fail verdicts of elements in the test log. Functional reports are generated from the test run as HTML files that use predefined report designs.

Before you begin

Before you can generate a functional report, you must successfully run a test or schedule and produce a test run.

The following report types are available:

- Extensible Stylesheet Language Transformation (XSLT) reports: These reports are faster to generate, but do not contain graphs.
- Business Intelligence and Reporting Tools (BIRT) report: These reports contain graphs but are slower to generate. You can customize and create your own BIRT report designs in the Report Design perspective of the workbench. BIRT report generation is not checkmark_perf.jpg when the workbench is running in a VMWare Windows™ image.



Note: If you use your own XSLT style sheets, verify that the style sheets contain this line: `<xsl:param`

```
name="languagePack" select="'default'"/>
```

1. In the **Test Navigator**, select a test run or runs.

You can use the Ctrl key to select multiple test runs or schedule runs. You cannot generate a functional report that contains more than 5000 calls or objects.

2. In the **Test Runs** view, right-click the test runs and select **Generate Functional Report**.

Result

This opens the **Generate HTML Functional Report** wizard.

3. Select the location in the workspace where you want to generate the functional report, and type the **Functional report base name**. A time stamp and the type of report is appended to this base name when the report is generated.

If you want to keep the temporary XML file that is created to generate the report for debugging purposes, select **Keep intermediate XML data**.

4. Click **Next**.

5. Select a predefined report designs or click **Add** to add a custom BIRT report design or an XSLT style sheet.

Choose from:

- **Common Functional Report:** This produces a generic functional report for all test protocols.
- **SAP Functional Report:** This produces a functional report for SAP tests.
- **Services - Failed events:** This produces a functional report for web service tests. The report contains only failed events. Events with other verdicts are not shown in the report.
- **Services - Failed tests:** This produces a functional report for web service tests. The report contains only failed tests. Tests with other verdicts or other event types are not shown in the report.
- **Services - Full:** This produces a functional report for web service tests. The report contains detailed information on all events.
- **Services - Summary:** This produces a brief summary functional report for web service tests.
- **Services - Truncated:** This produces a functional report for web service tests. The report contains detailed information on all events, but truncates XML contents after 500 characters.

One functional report is generated for each selected report design. Report designs marked with **(xslt)** use XSLT style sheets and are more suitable for larger reports.

6. Click **Finish**.

Results

The functional reports are generated as HTML files in the specified location in the workspace.

Publishing functional reports to the server

If you have generated functional reports for your test results, you can associate functional reports to test results and then publish them to IBM® Rational® Test Automation Server so that others can view them from the web browser.

Before you begin

You must have completed the following tasks:

- Accessed Rational® Test Automation Server.
- Created an offline user token to connect to Rational® Test Automation Server from Rational® Performance Tester. For more information, refer to [Managing access to the server](#).
- Created or joined a project in Rational® Test Automation Server.
- Upgraded legacy reports to the Web Analytics report format.



Note: You can right-click the report and select **Upgrade** to upgrade the legacy report to the Web Analytics report.

- Generated functional reports in Rational® Performance Tester. See [Generating functional reports on page 439](#).
1. Open Rational® Performance Tester, and then go to **Functional Reports** in the **Test Navigator**.
 2. Select the functional report that you want to publish to Rational® Test Automation Server.
 3. Right-click the selected report, and then click **Publish Report**.
 4. Expand the project folder, and then select the test result to associate with the functional report.

For example, if you have generated a functional report for the test result `test1`, then you must select the `test1` to associate the test result with functional report.

5. Click **OK**.
6. Click **New Server** and specify the URL of Rational® Test Automation Server.



Note: If you had added the URL of Rational® Test Automation Server in the product **Preferences**, then the **Server** field auto-populates the URL of Rational® Test Automation Server.

7. Enter an offline user token that you created on Rational® Test Automation Server.
8. Select a project from the **Project Name** drop-down list.

The **Project Name** drop-down list displays all the projects on Rational® Test Automation Server. The name of the team space for the project is displayed within parenthesis. You can select the appropriate project when there are identical project names in different team spaces.

You cannot create a project from the desktop client. If there are no projects or if you are not a member of any project or team space, then you must create a project or become a member of a project or team space on the server.

9. Enter the name of the label in the **Labels** field to add labels to the test result.



Note: If you run the test assets by using the **-labels** parameter from the Command-line interface (CLI), then the **Labels** field contains the name of the labels that you provided during the run time and labels that you added to the performance report.

10. Select the **Functional Report** checkbox from the **Reports** field.

11. Click **Publish**.

Results

You have published the functional report to Rational® Test Automation Server.

What to do next

You can log in to Rational® Test Automation Server and analyze the test results. For more information refer to [Test results and reports overview](#).

Publishing test results to the server

The test results indicate the quality of the application under test. Different stakeholders might want to check the quality of the application but do not have the desktop client installed. As a desktop client user, you can publish test results to IBM® Rational® Test Automation Server so that others can view them from the web browser.

Before you begin

You must have completed the following tasks:

- Accessed Rational® Test Automation Server.
- Created an offline user token to connect to Rational® Test Automation Server from Rational® Performance Tester. For more information, refer to [Managing access to the server](#).
- Created or joined a project in Rational® Test Automation Server.
- Upgraded legacy reports to the Web Analytics report format.



Note: You can right-click the report and select **Upgrade** to upgrade the legacy report to the Web Analytics report.

- Configured the firewall so that Rational® Test Automation Server enables connection on port number 443.

About this task

You can publish both performance and functional reports. You can set the publish parameters in the **Preference** page so that you do not have to do it after every run or you can set the parameters every time for the specific result that you want to publish. Based on the parameters, the test result is published to Rational® Test Automation Server after the test run is complete.

If you select **Prompt** from the drop-down list for the **Publish result after execution** option, after each test run, the **Publish Result** dialog is displayed to publish test results to Rational® Test Automation Server. You can modify the following options before publishing the results:

- If you want to publish reports to other than the default server added in the **Preferences** window, you can change the URL of Rational® Test Automation Server.



Note: If you change the server URL, you must enter an offline token to enable the publishing of test results.

- The default value for the **Result Name** field is the test result that you selected. You can provide a different name that you want to use.
- To identify specific test results, you can enter a name in the **Labels** field to associate it with the test result.



Note: The **Results** page of Rational® Test Automation Server displays the labels that you added in the **Labels** field.

- You can change the project name if you want to publish reports to a different project.



Note: The **Project Name** drop-down list displays all the projects on Rational® Test Automation Server. The name of the team space for the project is displayed within parenthesis. You can select the appropriate project when there are identical project names in different team spaces.

If there are no projects or if you are not a member of any project or team space, then you must create a project or become a member of a project or team space on the server.

1. Click **Window > Preferences > Test > Rational Test Automation Server**.
2. Specify the URL of the server and click **Test Connection**.



Note: In Linux operating systems, the **Password recovery setup** pop-up window is displayed, but it is hidden behind the **Preferences** dialog and you might not be able to close it. To work around this problem, you must press **ALT+TAB** repeatedly until the **Password recovery setup** pop-up window comes to the foreground.

3. Enter the offline user token that you generated on the server and click **OK**.

4. **Optional:** Click **Manage Offline Tokens** to view and remove the tokens that are associated with the desktop client, and click **Apply and Close**.

For example, if there is one instance of the desktop client for multiple testers to publish reports, each tester must remove the token created by other testers and add a new token.

5. Click the **Results** page from the navigation to apply settings for publishing reports.
6. Clear the **Use default Rational Test Automation Server URL** checkbox if the URL of the server is different than that is specified at **Window > Preferences > Test > Rational Test Automation Server**.

The format of the URL is `https://fully-qualified-domain-name:443`.

7. In **Publish result after execution** field, select when to publish test result.

In the initial stage when you are debugging a test, you might not want to publish the test result. Select one of the following options based on the requirement:

- Select **Never** to never publish the test results to the server.
- Select **Prompt** to prompt you to publish the test results after every test run.



Notes:

- A command-line interface always publishes test results to the server even if the product preference is set to **Prompt**.
- After each test run, the **Publish Result** dialog is displayed to publish reports to Rational® Test Automation Server.

- Select **Always** to publish test results after every test execution.

8. In **Publish to project** field, select a project that you are a member of on the server.

The **Publish to project** drop-down list displays all the projects on Rational® Test Automation Server. The name of the team space for the project is displayed within parenthesis. You can select the appropriate project when there are identical project names in different team spaces.

You cannot create a project from the desktop client. If there are no projects or if you are not a member of any project or team space, then you must create a project or become a member of a project or team space on the server.

9. In **Reports**, select the reports that you want to publish to the server.
10. Click **Apply and Close**.

Results

Test results are published to the Rational® Test Automation Server, depending on the parameters that you have set.

What to do next

You can log in to Rational® Test Automation Server and analyze the test results. For more information refer to [Test results and reports overview](#).

Related information

[Publishing specific results to the server on page 445](#)

Publishing specific results to the server

If you have a single test result or multiple test results that are not published to IBM® Rational® Test Automation Server, you can publish a single or all of them simultaneously.

Before you begin

You must have completed the following tasks:

- Accessed Rational® Test Automation Server.
- Created an offline user token to connect to Rational® Test Automation Server from Rational® Performance Tester. For more information, refer to [Managing access to the server](#).
- Created or joined a project in Rational® Test Automation Server.
- Upgraded legacy reports to the Web Analytics report format.



Note: You can right-click the report and select **Upgrade** to upgrade the legacy report to the Web Analytics report.

- Configured the firewall so that Rational® Test Automation Server enables connection on port number 443.

About this task

You can publish the statistical reports to Rational® Test Automation Server at this moment.

1. Open Rational® Performance Tester, and then go to **File > Export**.
2. Expand the **Test** folder, and then select **Execution Result to Rational Test Automation Server**.
3. Click **Next**.
4. Expand the project, and then select one or more test results that you want to publish.

Alternatively, you can right-click on all the test results that you want to publish from **Test Navigator**, and then select **Publish Results**.



Tip: You can press the Ctrl key to select the results from across projects.

5. Click **New Server** and specify the URL of Rational® Test Automation Server.



Note: If you had added the URL of Rational® Test Automation Server in the product **Preferences**, then the **Server** field auto-populates the URL of Rational® Test Automation Server.

6. Enter an offline user token that you created on Rational® Test Automation Server.
7. Select a project from the **Project Name** drop-down list.

The **Project Name** drop-down list displays all the projects on Rational® Test Automation Server. The name of the team space for the project is displayed within parenthesis. You can select the appropriate project when there are identical project names in different team spaces.

You cannot create a project from the desktop client. If there are no projects or if you are not a member of any project or team space, then you must create a project or become a member of a project or team space on the server.

8. Click a row against the result and type the name of the label in the **Labels** column to add labels to the result.



Note: If you run the test assets by using the **-labels** parameter from the Command-line interface (CLI), then the **Labels** field contains the name of the labels that you provided during the run time and labels that you added to the performance report.

9. **Optional:** Click **Add Common Labels** and type the name of the label to apply a common label to the selected results.
10. Select the reports that you want to publish to the server from the **Reports** section.
11. Click **Publish**.

Results

You have published the test results to Rational® Test Automation Server.

What to do next

You can log in to Rational® Test Automation Server and analyze the test results. For more information refer to [Test results and reports overview](#).


Customizing reports

You can customize reports to specifically investigate a performance problem in more detail than what is provided in the default reports.

Creating custom reports

If the default reports do not address your needs, you can create your own reports.

About this task



Before you create a custom report, determine the ways in which the custom report will be different from or similar to the system-supplied reports. You can use a default report as a template, modify the counters, and save it with a different name. You can create a copy of pages or charts in a report that are based out of existing pages or charts. To copy the pages or charts, go to the Edit view and click the **Duplicate** icon. 

You can also create a report from scratch and add the required counters. Counters for test protocol are explained in the topics in the [Reports and counters on page 471](#) section.



Note: If a counter has a lot of data, the graph is not represented properly. To ensure that the graph is displayed properly, you must filter out some of the data.



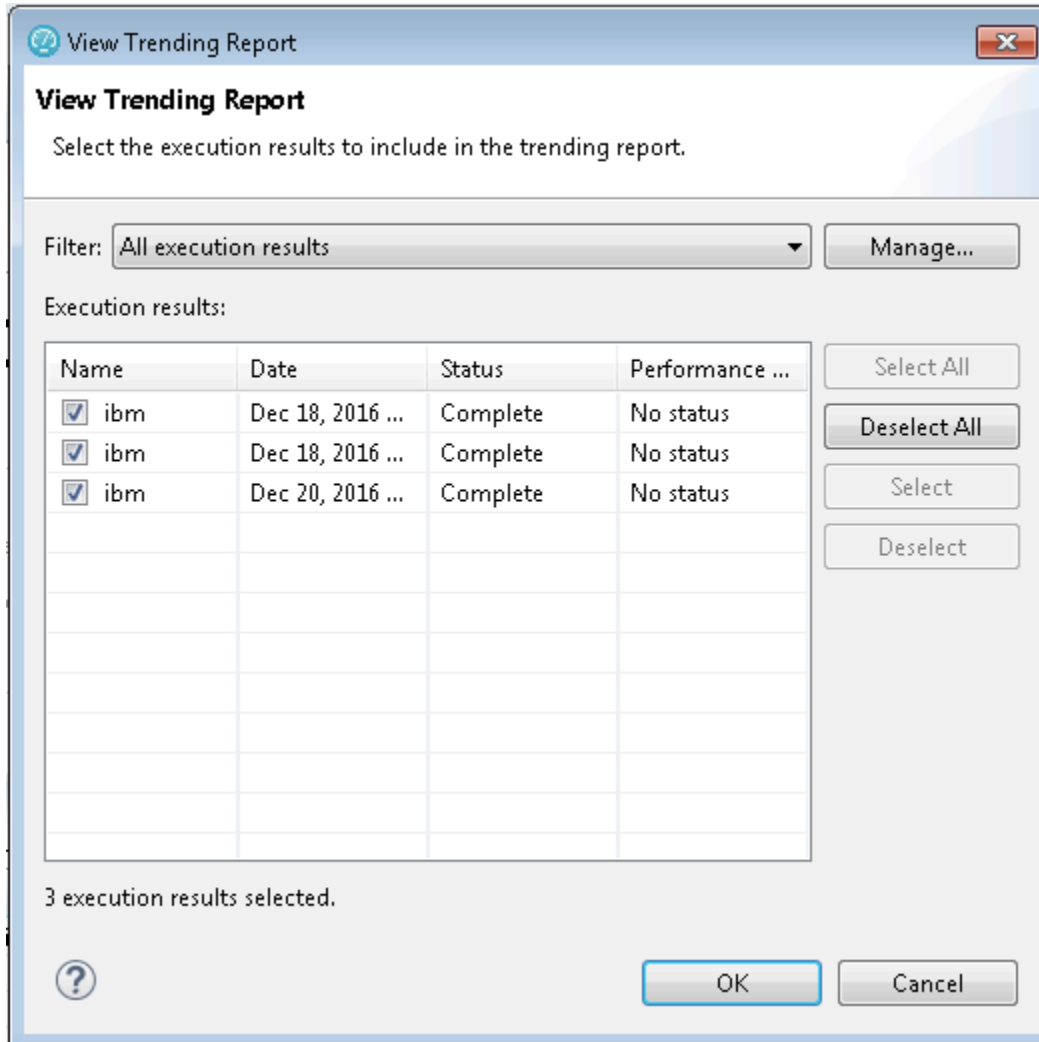
1. From the report, click **Menu**  and click **New**.
2. In **Create a new report** dialog, specify a name and description about the new report and click **Create**.
3. To change the page title, click the default page title and specify a different name.
4. Click **Click to insert a row** and specify the number of columns to add the views.
Each view represents a bar chart, line chart, or pie chart.
5. Select a view. To add counters to the view, click **Settings** .
6. On the View Settings page, select a counter and add its details.
7. Click **Apply** and from the **Menu**, click **Save** to save the report.
8. To add more views to the report, repeat steps 4 through 7 again.

Viewing trending reports

To view the trend of response time for an application over a period of time, open the trend report for a run. In addition to the response time, you can view the trend for the loops, transactions, and performance requirements for the application.

About this task

The trend report can help you determine the response times of the application at different milestones. For instance, you can run the performance test for sprint or milestone builds and tag them. When generating the trend report, you can specify conditions such as results that are less than 60 days old and include 'milestone' tag.



You cannot save a trend report. So, if you close the report, you have to generate it again.

To view the trend report:

1. In the Test Navigator view, select the run for which to open the trend report.
2. Right-click the run and click **View Trend Report**.
3. To view the trend that is based on certain criteria, in **Filter**, select a filter criteria.
If there is no customized filtering criteria, create one by clicking **Manage** and then **Add**.
4. To save the criteria, click **Save**, specify a name to the filter, and click **OK**.
The results in the execution results table are filtered out according to the specified criteria.
5. Click **OK**.

Result



The trend report is generated.



Filtering data in test results


You can filter the data in a test result that is displayed in a report so that you can remove the unnecessary data and focus on the data that is significant for the analysis.

Before you begin

You must have a test result.

1. Double-click the test result from the **Test Navigator**.
2. Select a report from the drop-down list.
For example, the Performance Report.
3. Click the **Menu** icon , and then click **Edit**.
4. Select a page from the left pane in which you want to filter the data.
For example, the Page Performance page.
5. Click the **Settings** icon  on a specific graph or table.
6. Click the **Filters** tab on the **View Settings** page.
7. Perform any of the following actions described in the following table to filter the data:

Op- tions	Actions
Fil- ter by count	<p>Perform the following steps:</p> <ol style="list-style-type: none"> a. Clear the Show highest values check box to display the smallest values for the pages.  Note: By default, the Show highest values check box is selected. b. Enter a value in the Number to display field to display the items on the graph or table based on the specified value for the selected counter.  Note: The title of the page is updated with the value that you specified along with the Show highest values field. For example, if you selected the Show highest values check box and entered 10 in the Number to display field for the Performance Summary page, then the title is displayed as follows: Performance Summary (10 Highest). c. Select the counter from the Primary counter for table filtering field by using the drop-down list if you want to filter the data for the other counter. d. Select the component for the counter that you selected from the Component drop-down list.

Op- tions	Actions
	<p>For example, consider that you performed the following actions to filter the data:</p> <ul style="list-style-type: none"> ◦ Selected the Show highest values check box. ◦ Entered <i>5</i> as a value in the Number to display field. ◦ Selected <i>Page Response Time</i> as Primary counter for table filtering and <i>Average</i> as Component. <p>Then, the graph or table displays 5 pages that include the highest <i>Average Page Response Time</i> during the test run.</p>
Fil- ter by value	<p>Perform the following steps:</p> <ol style="list-style-type: none"> a. Clear the Show counters above value check box to display the lower values for the pages. <p> Note: By default, the Show counters above value check box is selected.</p> <ol style="list-style-type: none"> b. Enter a value in the Filter value field to display the items on the graph or table based on the specified value for the selected counter. c. Select the counter from the Primary counter for table filtering field by using the drop-down list if you want to filter the data for the other counter. d. Select the component for the counter that you selected from the Component drop-down list. <p>For example, consider that you performed the following actions to filter the data:</p> <ul style="list-style-type: none"> ◦ Cleared the Show counters above value check box. ◦ Entered <i>800</i> as a value in the Filter value field. ◦ Selected <i>Page Response Time</i> as Primary counter for table filtering and <i>Average</i> as Component. <p>Then, the graph or table displays the pages that include the <i>Average Page Response Time</i> lesser than <i>800 ms</i> during the test run.</p>
Fil- ter by name	<p>Perform the following steps:</p> <ol style="list-style-type: none"> a. Enter a label name in the Filter value field. <p>The label name is the name that you provided for a page when you recorded the test.</p> <ol style="list-style-type: none"> b. Select the Case sensitive check box to find the pages that exactly match with the letter case of the name that you entered in the Filter value field. c. Select any of the following options to find pages more effectively:

Options	Actions
	<ul style="list-style-type: none"> ▪ Include counters whose label contains filter value ▪ Include counters whose label equals filter value ▪ Exclude counters whose label contains filter value ▪ Exclude counters whose label equals filter value



Note: The fields **Cumulated**, **Label**, **Path**, and **Unit** are non-editable and display the preconfigured values for the selected counter.

Result

In the **Preview** section, the values in the graph or table change as and when you change the filter options.

8. Click **Apply** to apply the changes that you made for the filters.
9. Click **Save** from the menu to save the data that you filtered.
10. Click **Edit** from the menu to exit the edit mode.

Results

You have filtered the data on the specific page for the report.

Customizing the appearance of graphs in a report



To display the data in a table, bar chart, or line chart in a manner that caters to your test requirements, use the controls that are available in the View Options of a report.

1. In the Test Navigator, expand the project until you locate the run.
Each run begins with the name of the schedule or test, and ends with the date of the run in brackets.

2. Double-click the run.

Result

The default report opens.

3. Click the **Menu** icon  and click the **Edit** icon.
4. Click the **Settings** icon  for the graph or table to modify.
5. The controls that are available in the View Options section depend on the graph type: bar chart, line chart, or table. For each graph type, only the applicable controls are displayed. You can adjust the following controls:

Option	Description
Adapt Y Scale	To compute minimum and maximum limit on the Y axis, select the check box. (all charts)
Title	Specify a title to the graph.
Show title	To hide the title, clear the check box.

Option	Description
X Axis Main items	Select the item to view on the X Axis.
Stacked Items	Select the item such as Pages or Time Ranges to view them in stack instead of separate bars.
Adapt Y scale on zoomed data	To adjust the Y scale according to zoomed data, select the check box. (line charts)
Show time ranges	To display the time range in the background of the chart, select the check box.
Line smoothing	To apply corners, clear the check box.
Orientation	To view bar charts horizontally or vertically, select an orientation.
Labels display policy	To hide the labels in a bar chart, select Hidden . To be able to accommodate labels within the frame of a bar chart, select Adaptative . If you select Fixed , long labels might not be visible.
Time line visibility	To view the time line of the chart in partial or full view, select Small or Full options. Drag the time line to create a new time range. If those options are specified, you can drag and create a new time range on the chart itself. If you select None , the time line is not visible and you cannot create a new time range on the chart.

6. After making the changes, click **Apply** and from the Menu click **Save**.

To apply the changes to other reports, you can export the report definition and import it back. See [Exporting report metadata on page 463](#).

When you install Rational® Performance Tester 10.5.1 or later on a Windows Server 2022 computer, and open the Web Analytics reports for editing, the icons are not displayed.

To workaround the problem, you can perform the following steps on browsers other than Internet Explorer:

- a. Open Rational® Performance Tester.
- b. Go to **Window > Preferences**.
- c. Select **Test > Performance Test Reports**.
- d. Select **In external browser** under **Open Statistics Report**.
- e. Click **OK**.
- f. Close the report and then double-click the result.

Perform the following steps in Internet Explorer:

- a. Enter Internet options in the start menu.
- b. Select **Local Intranet** in the Security tab.
- c. Click **Sites**.
- d. Type <http://127.0.0.1>, and then click Add.
- e. Click **Close**, and then click **OK**.
- f. Restart Rational® Performance Tester

Changing the report displayed during a run

Use this page to select the default report that opens during a run. Typically, you select **Determine default report based on protocols in test**, which determines the protocols that you are testing and automatically opens the appropriate protocol-specific reports.






1. Open the Default Report Preferences page. Click **Window > Preferences > Test > Performance Test Reports > Default Report**.
2. In the Default Report window, select **Determine default report based on protocols in test** or a specific default report to display a customized report or if the default reports do not meet your needs. Note, however, that you will have to change this setting when you record other protocols.
3. Click **Apply**, and then click **OK**.

Modifying counters in a graph

To gather additional information for diagnosing performance problems, you can modify the counters that are displayed in a graph.

About this task

Counters are specific in-built queries that gather statistical information from the recorded test. The information can be the number of page hits, response time, and user load. By default, each report has pre-defined counters. You can add or remove the counters from the graphs in the report.

1. Double-click the report from the **Test Navigator** to modify the counters.
2. Click the **Menu** icon , and then click **Edit**.
3. Click the **Settings** icon  to modify counters on a specific graph.
4. Select the **Counters** tab on the **View Settings** page, and then perform the following steps to add, remove, or move the counters in a graph:
 - a. Click the **Plus** button , and then select the counters from the drop-down list to add a counter.
 - b. Click the **Remove** button  to remove the selected counter.
 - c. Use the **up-down** control buttons  to move a counter.

The **Preview** section displays the result of the actions.


5. **Optional:** For a selected counter, you can change the component of the counter. Based on the counter selection, the **Component** field shows the options available for that counter.

6. Perform the following steps to define a percentile value as decimal number for the counter:
 - a. Select the **Percentile** as component from the **Component** drop-down list.
 - b. Enter a new value in the **Percentile value** field.
For example, 99.9.
7. **Optional:** You can change the **Cumulated** value for the selected counter if you want to show the cumulation values on a graph. Select one of the following options based on the requirement:
Choose from:
 - Select **No** to display the value of the last interval on the current time range.
 - Select **From the beginning of the time range** to display the cumulation of all values of the current time range.
 - Select **From the beginning of the run** to display the cumulation of all values from the beginning of the run to the end of the current time range.



Notes:

- For line charts, the default value is **No**.
- For bar chart, pie chart, and tables, the default value is **From beginning of time range**.
- The fields **Label**, **Path**, and **Unit** are non-editable.

8. Click **Apply**.
9. Click **Save** from the menu.
10. **Optional:** Click **Save As** to create another report with these changes.
11. Click the **Edit** icon  to exit the edit mode.

Results

You have updated the counter information for the specific report.

Correcting time offset

Response time breakdown and resource monitoring data is time stamped using the system clock of the host computer. If there are differences between the system clocks of the host computers that you include in a test, then response time breakdown and resource monitoring data are skewed in reports. The best practice is to synchronize the system clocks on all computers that you include in a test. When this is not possible, you can correct the time offset of each host computer after a test run. Typically, correct the time offset on all computers to match the system clock of the workbench computer.

After you run tests with resource monitoring or response time breakdown enabled, follow these steps to correct the time offset:

1. In the **Test Runs** view, right-click the host where you want to correct the time offset; then click **Correct Time Offset**.
2. Select a **Shift Direction** of positive or negative. A positive shift moves the response time breakdown and resource monitoring data on the selected host to the right. A negative shift moves the response time breakdown and resource monitoring data on the selected host to the left.
3. Type the hours, minutes, or seconds of the time offset you want to use, and click **OK**.

Results

The response time breakdown and resource monitoring data on the selected host displays with a corrected time offset.

Export test results

You can export the test result in different formats to share it with different stakeholders.

Creating an executive summary from the workbench

To create a printable report that summarizes the findings of the performance test run on a single view, create an executive summary. You can export the data of the test run as an executive summary from a single report or from multiple reports such as Performance Report, Mobile and Web UI Statistical Report, Transaction Report, and Loop Report. You can then open the summary in a word-processing program to further format and annotate the data.

About this task

You export the executive summary to a local or a shared directory. You can export a test run from the Web Analytics report, from the test workbench, and from the command line.

When you use the workbench approach to create an executive summary, you can choose to create the summary for multiple runs and multiple report types at the same time. When you use the Web Analytics reports or the command line, you create executive summary for a particular run and a report at a time.

To create an executive summary from the workbench:

1. Click **File > Export > Test > Executive Summary**. You can also right-click the runs to create executive summaries for from the Test Navigator view and click **Export > Test > Executive Summary**. Each run would have one executive summary.
2. In **Export Directory**, specify the folder path to save the executive summary and click **Next**.
3. Select the runs to create the executive summary for. To create an executive summary for comparing two runs, select the **Generate a compare report** check box and select the main run to compare the report with and click **Next**.
4. Select a report to export and click **Finish**.

What to do next

A folder with the name of the run is created on the specified folder. To view the executive summary, open the `index.html` file.



Creating an executive summary from the Web Analytics report

To create a printable report that summarizes the findings of the performance test run on a single view, create an executive summary. You can choose to view the executive summary on a web browser or save it on a computer.

About this task

To generate an executive summary for a particular report such as Transaction report or Performance report, open that report and then follow the steps in this topic. To generate an executive summary for multiple reports or test runs at the same time, see [Creating Executive Summary from Workbench on page 455](#).

To create an executive summary from the Web Analytics report:

1. Open the test run to create executive summary for. The test run opens in a web browser.
2. From the dropdown, open the report for which to create executive summary.
3. Click the **Menu** icon  , click the **Share** icon  , and click **Executive Summary**.
4. To view the executive summary of the report in another browser tab, click **View on another tab or page of the browser**. To save the executive summary, click **Save as an HTML file on the local computer**.
5. Click **Generate**.

Exporting reports to HTML format



When you export a test run and share it, people can analyze test data without using the test workbench. You can also email the test run or post it on a web server. The exported run can be displayed and printed from any browser. A test run contains multiple reports. You can choose to export any or all of the reports.

About this task

You can export a single run to a local directory or multiple runs in the compare mode to a directory. In addition to exporting a test run from Web Analytics, you can export it from the test workbench itself and from command line.

To export from the workbench, select a single run or multiple runs and click **Export > Test > Performance Test Run Statistics as HTML application** . To generate a single report comparing multiple runs, in the Export wizard, select the **Generate a compare report** check box and select a base run from the dropdown. To generate one report for each run, do not select the check box.

To export from Web Analytics:

1. Open the test run to export.
The test run opens in an external or internal web browser.
2. Click the **Menu** icon  , click the **Share** icon  , and click **Export Session to HTML**.
3. Select the type of report to export and click **Export**.

- When you export from the workbench, specify a path to the folder to save the exported report.
Your current project is the default save location. You can create a folder outside of the project to store exported reports.

When you export from an external browser, the report is compressed and saved to the default download location of the browser.

What to do next

You can now share the test run with others. You can also export the test run from command line.

Related information

[Running a test or schedule from a command line on page 395](#)

Exporting results to a CSV file



To further analyze test results, you can export all statistics or specific statistics captured during a run to a CSV file.

About this task

You can export a single run to a local directory or multiple runs in the compare mode to a directory. You can export the runs from Web Analytics report, workbench, and command line. To export from the workbench, select a single run or multiple runs and click **Export > Test > Performance Test Run Statistics as CSV File**. To export data of specific time ranges, on a subsequent page select a time range.

To export the run from command line, see the parameters in the [Running a test or schedule from a command line on page 395](#) topic.

- Open the test run to export.

- Click the **Menu** icon  , click the **Share** icon  , and click **Export Session to CSV**.
- Select the encoding system for the export.
- Complete either one of the following steps:

Choose from:

- To export only the last value of each counter from the results or to export data of specific time ranges, select **Simple**.



Note: When you export data of specific time ranges, for example, 5 Users or 15 Users, a separate column is created in the CSV file for each time range.

- To create multiple CSV files if the number of columns exceed the specified value, select the **Split output if column exceeds** check box and specify a value.
- To export all of the data for the run, select **Full**.

To include description about the name of the run, node name, and time range for each counter, select the **Include per instance counters**.

- To export data of each location (agent) in a separate section in the CSV file, select the **Export each agent separately**

To export data of each location (agent) to separate CSV files, select the **One file per agent** check box.

5. Click **Export**. If you export from the workbench, the report is saved in the specified folder. If you export from an external browser, the report is downloaded in a compressed format to the default download location of the browser.

What to do next

You can now analyze and share the report with people who are not using the workbench.

Related information

[Exporting reports to HTML format on page 456](#)

Methods to export test results into a JSON file

You can export test results to a JSON file in different ways. You can then share the exported test results with different stakeholders and they can analyze test data without using the product.

You can export the test results into a JSON file by using any of the following methods:

- Product
- Web analytics report from the product
- Command-line interface

When you use the product method to export test results, you can select multiple test results simultaneously. However, when you use the web analytics report or the command line, then you can export a report of the test result.

Related information

[Running a test or schedule from a command line on page 395](#)

Exporting results to a JSON file by using the product

When you want to analyze test results in a text format, you can export all test results or specific test results that are captured during a test run to a JSON file by using the product.

Before you begin

You must have at least one or more test results.

About this task

You can automatically export reports to a comma-separated values (CSV) or JSON file at the end of a test run. You can select the appropriate options to export reports from the command line, workbench, or both from the product preferences (**Window > Preferences > Test > Performance Test Reports > Export Reports**).

1. Open the workbench, and then go to **File > Export**.
2. Expand the **Test** folder, and then select **Performance Test Run Statistics as JSON file**.
3. Click **Next**.

Result

The **Export Performance Test Run Statistics** window is displayed.

4. Specify the folder path to save exported test results in the **Export Directory** field.

Alternatively, you can click **Browse** and select a directory.



5. Select the encoding system for the exported results from the **Exported encoding** drop-down list.
6. Expand the project, and then select one or more test results that you want to export.




Note: When you select a test result, you can export data of specific time ranges. For example, Entire Run, 5 Users, or 15 Users. By default, the report includes the data for the Entire Run.

7. Click **Next** and perform any of the following actions described in the following table to include the type of results into the JSON file:

Requirements	Options available	Actions
When you select a test result	Simple	<ol style="list-style-type: none"> a. Select Simple to export only the last value of each counter from results. b. Optional: Select Time Range Comparison to export data of specific time ranges. c. Optional: Select the time range that is displayed depending on the test result that you selected.

Requirements	Options available	Actions
		<p>For example, Entire Run, 5 Users or 15 Users.</p> <p> Note: By default, the report includes the data for the Entire Run.</p>
When you select one or more test results	Simple	Select Simple to export only the last value of each counter from results.
	Full	<p>a. Select Full to export all the data for every sample interval during the test run.</p> <p>b. Optional: Select the Split output if counters number exceeds check box, and then specify a value to create multiple JSON files if the number of counters in the exported file exceed the specified value.</p> <p> Note: The default value is set to 250.</p>

8. Select the **Include per instance counters** check box to include a description of the name of the result, node name, and time range for each counter.
9. Select the **Export each agent separately** check box to group the data in the exported JSON file by prefixing the name of the counters with the name of the agent.
10. **Optional:** Select the **One file per agent** check box to export data that was run on the agent in separate JSON files.

 **Note:** The **One file per agent** option is available only when you select the **Export each agent separately** option.

11. Click **Finish** to save the exported results to a JSON file.

Results

You have exported test results in to the JSON file by using the product.

What to do next

You can now analyze the result and share the file with stakeholders for further analysis of the results.

Related information

[Running a test or schedule from a command line on page 395](#)

Exporting results to a JSON file by using a web analytic report

When you want to analyze test results in a text format, you can export all test results or specific test results that are captured during a test run to a JSON file by using a web analytic report.

Before you begin

You must have at least one or more test results.



About this task

You can automatically export reports to a comma-separated values (CSV) or JSON file at the end of a test run. You can select the appropriate options to export reports from the command line, workbench, or both from the product preferences (**Window > Preferences > Test > Performance Test Reports > Export Reports**).

1. Open the test result that you want to export from the **Test Navigator**.

Result

The test result is displayed in a browser.

2. Click the **Menu** icon  and then click the **Share** icon .
3. Click **Export Session to JSON file**.
4. Select the encoding system from the drop-down list for the exported results.

You can select the default encoding unless the exported JSON file is shared by multiple applications that recognize a specific encoding.

5. Perform any of the following actions described in the following table to include the type of results into the JSON file:

Op-tions	Actions
Simple	Select Simple to export only the last value of each counter from results.

Options	Actions
Full	<ol style="list-style-type: none"> a. Select Full to export all the data in the results. b. Optional: Select the Split output if counters number exceeds check box, and then specify a value to create multiple JSON files if the number of counters in the exported file exceed the specified value.

6. Select the **Include per instance counters** check box to include a description of the name of the result, node name, and time range for each counter.
7. Select the **Export each agent separately** check box to group the data in the exported JSON file by prefixing the name of the counters with the name of the agent.
8. **Optional:** Select the **One file per agent** check box to export data that was run on the agent in separate JSON files.



Note: The **One file per agent** option is available only when you select the **Export each agent separately** option.

9. Click **Export** to specify the folder path to save the exported result.
10. Click **OK** to save the exported results to a JSON file.

Results

You have exported test results in to the JSON file by using the web analytic report.

What to do next

You can now analyze the result and share the file with stakeholders for further analysis of the results.



Related information

[Running a test or schedule from a command line on page 395](#)

Sharing URL of test run

When you share the URL of the test run with other people, they can view and analyze the test results on a browser on their computer if the test workbench is running on your computer at that time.

To share the URL of the test run:

1. Open the test run to share.
2. Click the **Menu** icon  and click the **Share** icon  and select **Share Execution Result URL**.
A unique URL is created for the test run.
3. Copy the URL and click **Close**.

What to do next

You can now share the URL of the test run with anybody.

Exporting report metadata

To share report metadata with another test workbench user, export the report definition. Use this option to share customized report formats with other users. The recipient imports the metadata with Eclipse's **Import** option and views the report from the Test Navigator or in the list of reports in the web report.

To export report metadata:

1. Click **File > Export**.
2. In the **Export** window, expand the **Test** folder, select **Report Definitions**, and click **Next**.
3. In **Save to File**, select the file that will contain the report. This file is created if it does not exist.
4. In **Select Report**, select the report to export, and then click **Finish**.

The file is saved in the `.report` format.

What to do next

To apply another report definition to your reports, import that report metadata by clicking **File > Import > Report Definition**, and browse to the `.report` file.

Viewing response time breakdown

You can do detailed analysis of the response time to find bottlenecks in the HTTP traffic of the application.

Viewing page element responses

You can view the response times for individual page elements in reports, to determine which elements are the slowest.

About this task

Page element response times do not include client delay or connection time. Because page elements can be returned in parallel from the server under test, the page response time is not necessarily the sum of the page element response times.

1. Open the web analytics reports.
2. On the Page Performance report, click a page (represented by a bar) and click **Page Element Responses**.
The Page Element Responses report displays response time for all of the elements of the page.
3. To return to the original report, click the Page Performance link in the breadcrumb.

Viewing page response time contributions

You can view the response time contributions for individual pages to determine how much time was actually taken by the page to load and the time taken for the connection to go through and the delay on the client side of each page.

Before you begin

Because page elements can be returned in parallel from the server under test, the page response time is not necessarily the sum of the page element response times. Client delay and connection time also contribute to page response time. The page response time can be greater than the sum of the page element response times if, for example, a lengthy connection time adds a delay. Connection time includes the time required for Domain Name Services (DNS) lookups. Conversely, the page response time can be less than the sum of the page element response times if multiple page elements are returned in parallel.

1. Open the web analytics report.
2. On the Page Performance report, click a page (represented by a bar) and click **Page Response Time Contributions**.

The Page Response Time Contributions report shows the average response time taken for Connection Time, Client Delay Time, and Page Element Response Time.

3. To return to the original report, click the Page Performance link in the breadcrumb.

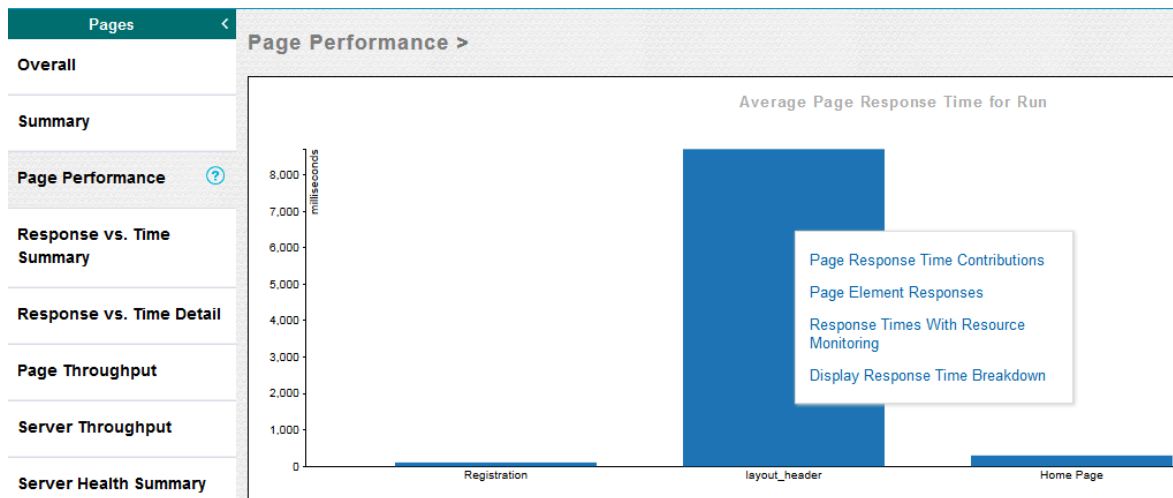
Viewing response time breakdown data

In addition to viewing page element and page response time, you can do further analysis to view the response time data for each method.

Before you begin

You must have instrumented the servers. See [Instrumenting local servers](#).

1. Open the web analytics reports.
2. On the Page Performance report, click a page (represented by a bar) and click **Display Response Time Breakdown**.



You can also directly access the response time breakdown data from the Page Element Responses report.

3. In the workbench, the **Page Element Selection** dialog is open.
4. Select a page element to view response time breakdown for and click **Finish**.

Comparing service test response contents

After running a service test with equal or contain verifications points, you can use the **Raw transaction data** view to compare the actual XML or text contents of the message returns with the expected results of the verification points in the test log.

Before you begin

Before comparing results you must have run a service test with equal or contain verification points.

To compare the actual and expected results of a service message return:

1. After running the test, open the test log.
2. Make sure that the **Raw Transaction Data** view is open. If necessary, click **Window > Show View > Raw Transaction Data** to open the view.
3. Select an equal or contain verification point.

Result

The **Raw Transaction Data** view displays a comparison between the expected data from the verification point and the actual data that is received during the run.

Related information

[Viewing message content on page 362](#)

Logs overview

Rational® Performance Tester uses logs to store different types of information, which you can use to determine the reason for a test failure.

Rational® Performance Tester has the following logs:

Test logs

The test log contains a historical record of events that occurred during a test run or a schedule run, as well as the status of each verification point. The test log sets a verdict for each run as follows:

- **Pass** indicates that all verification points matched or received the expected response and all the test steps successfully completed.
- **Fail** indicates that at least one verification point did not match the expected response or that the expected response was not received or a Web UI step did not run successfully.
- **Error** indicates one of the following results: a primary request was not successfully sent to the server, no response was received from the server for a primary request, or the primary request response was incomplete or could not be parsed.
- The verdict is set to **Inconclusive** only if you provide custom code that defines a verdict of **Inconclusive**.

The verdict is rolled up from the child elements to the test level. For example, if a user group contains 25 virtual users, and five virtual users have failed verdicts, that user group has only one failed verdict, not five.

The test log file is stored in binary format with a `.executiondlr` file name extension in the project directory of your workspace. You can also view the test log in the user interface.

For more information about viewing test logs, see .

Problem determination logs

You can set the level of information that is saved in the problem determination log during a run. By default, only warnings and severe errors are logged. Typically, you change this log level only when requested to do so by the Support person.

The problem determination logs contain internal information about the playback engine. These logs are particularly useful for debugging problems such as Kerberos authentication, SSL negotiation, and resource constraints on an agent. The log files are named `CommonBaseEvents00.log` and are located in the deployment directory. For example, if you play back a schedule on an agent and set `C:\Agent` as the deployment directory, the problem determination log files are in a directory similar to `C:\Agent\deployment_root\\A1E14699848784C00D2DEB73763646462\CommonBaseEvents00.log`. If a large amount of log information is generated, multiple `CommonBaseEvents` files are created.

For more information about setting problem determination level, see [Setting the problem determination level](#).

Agent logs

Look in `%TEMP%` directory for the `majordomo.log` file. This file contains information about the attempts to contact the workbench including information about any failures and the reason for the failures.

On the Microsoft™ Windows operating system, the `%TEMP%` directory is typically at `%USERPROFILE%\AppData\Local\Temp`.

If the majordomo service is configured to log in as Local System Account, then the `%TEMP%` directory is at `%SystemRoot%\TEMP`, typically `C:\Windows\TEMP`.

Error logs

If an error message is displayed when you run tests, try looking up the error message in the *Performance testing error messages* section of the online help. Only the most common error messages are listed. If no error message is displayed when you encounter a problem, open the error log by clicking **Window > Show View > Error Log**. If the workbench shuts down while running tests, restart the workbench and examine the error log. By default, warning and error messages are logged. You can increase the default logging level by clicking **Window > Preferences > Logging**. The log file is stored in the `.metadata` directory of your workspace. To avoid excessive logging, the Logging Level should be adjusted for individual Logger Names in the Loggers tab. For example, to get more information about a problem connecting with IBM® Engineering Test Management, increase the Logging Level for `com.ibm.rational.test.it.rqm.adapter` Logger Name. For the licensing issue, adjust the level for




com.ibm.rational.test.It.licensing Logger Name. When you no longer need the extra logging, use the **Restore Default** button in the Logging Preferences to reset all the levels to their recommended defaults.

Viewing test logs

To see a record of all the events that occurred during a test run or a schedule run, as well as the status of each verification point, open the test log for that run. You can also compare an event from the test log with the request or response in the test to view the differences between the recording and the playback of the test.

About this task

The test log file is stored in binary format with a `.executiond1r` file name extension in the project directory of your workspace. You can also view the test log in the user interface.

1. In the Test Navigator view, right-click the executed test; then click **Display Test Log**.
2. On the **Overview** tab, view the verdict summary for the executed test. To see the potential data correlation errors in a separate view, click **Display Potential Data Correlation Errors**.
3. On the **Events** tab, view the errors, failures, and passes for each event in the test.
 - To navigate to the verdict type, click the **Select the verdict type**  icon.
4. On the **Data Correlation** tab, see all the references and substitutions that occurred during a test execution, as well as the data correlation errors. By default, you view both references and substituters. To view only substituters, click the **Show References**  icon. To view the correlation data for each virtual user that was executed, click the **Merge Users**  icon. This icon is enabled only for a schedule. In the **Data Correlation** section, when you click an event, you can see the correlation data in either the Content View or the Table View.

What to do next

From the test log, you can submit, search, and open defects in a defect tracking system. For details on configuring the test log preferences and working with defects, see [Associating defects with a test log](#).

Viewing reports after a run

Reports are generated and displayed automatically after a run. Each test result begins with the name of the schedule or test, and ends with the timestamp of the run in brackets.

About this task

In version 8.5.1 or later, for a service test report, you can choose not to generate a report automatically after a run by clicking **Window > Preferences > Test > Performance Test Reports > Service Test Reports** and clearing the **Functional Test Report generation after Test execution**.

1. In the Test Navigator, expand the project until you locate the run.
2. Do either of the following:

- To view the default report, double-click the run. To change the default report, Open the Default Report Preferences page. Click **Window > Preferences > Test > Performance Test Reports > Default Report**
- To view another report, right-click the test run, click **Display Report**, and then select the report to display.



Note: You can also view reports remotely from a web browser. For information about viewing reports remotely, see [Accessing reports remotely on page 468](#).

Accessing reports remotely

Before executing a schedule or test, you can enable an option so that you can access reports remotely from a web browser. When you make changes to a report, the changes are saved to the workspace where the workbench is running.

1. Open Rational® Performance Tester
2. Click **Windows > Preferences > Test > Performance Test Reports > Web Reports**.
3. Select the **Allow remote access from a web browser** checkbox to enable remote access to reports.
4. Select the **Allow control of schedule execution from the web browser** checkbox to enable the remote control of schedule execution tasks.
5. **Optional:** Enter the port number in the **Secure port** field if the default port number is used by another service. The default non-secure port number for web reports is 8443.
6. Select the **User authentication is required to access reports** checkbox to provide security for web reports.
7. Enter the credentials in the **Username** and **Password** fields.



Note: You must use the same login credentials to access reports remotely.

8. **Optional:** Select the **Also enable non-secured access** checkbox to access the web reports by anyone who has the URL of web report.
9. **Optional:** Enter the port number in the **Non-secure port** field if the default port number is used by another service. The default non-secure port number for web reports is 8080.
10. Click **Apply**.

Results

You have accessed reports remotely.

What to do next

The URL of the web reports is displayed in the **Reports will be available at** section. You can click **Copy** and paste the URL `http://host_name:8080/analytics/web/index.html` in a web browser to access it. To access a secured report, you must specify the login credentials that you have set in the preference of Rational® Performance Tester.

Exporting test logs

To process data from a performance test in another application or to use search tools to locate text in a test log, export the test log to a text file.

1. In the Test Navigator, right-click the run, and select **Export Test Log**.
 - a. **Optional:** To export only a portion of the test log, open the test log by right-clicking the test run and then selecting **Display Test Log**. Right-click the elements to export, and then select **Export Log Element**.

Result

The **Export Test Log** window opens.

2. In the Export Test Log window, specify a location for saving the file, and then select options as follows:

Option	Description
Export format	Select default encoding or Unicode encoding.
Include event time stamps	Select to include event time stamps.
Include detailed protocol data	Select to include detailed protocol data. This option is available only for HTTP test runs.
Include response content	Select to include response content. This option is available only for HTTP test runs.
Include known binary data	Select to export binary data. This option is available only for HTTP test runs.

3. Click **Finish**.

Result


The test log is exported to a text file.

Exporting event log

To view all the events that occurred during the run of a test from another file, you can export this data from the Event Log panel, to an XML, CSV, or text file.

Before you begin


You must run a test to view data in the Event Log panel.

1. On the Event Log panel toolbar click the **View Menu** arrow icon  and select **Export Event Log**.
2. In the Save dialog, specify the location and format in which you want to save the events.

Exporting event console output

To view errors and other events of a test run from another file, you can export this data from the Execution Event Console view to an XML, CSV, or text file.

Before you begin

- Ensure that the Execution Event Console view is open by clicking **Window > Show View > Execution Event Console**.
 - Ensure that the test is run and the Execution Event Console view contains data.
1. From the Execution Event Console view toolbar, click the **View Menu** arrow icon  and select **Export**.
 2. In the Save dialog, specify the location and format in which you want to save the events.

Viewing resource monitoring data





You can analyze the performance of the computer resources, application server, or database servers by viewing the resource monitoring data in web analytics reports.

Adding resource counters to reports

To view performance data of resource counters that are not shown in the report by default, you can add the resource counters.

Before you begin

You must have enabled capturing of resource monitoring data in the schedule. See [Enable resource monitoring](#).

1. Open the Performance Report, and from the list of Pages, click **Resources**.
2. Click the **Menu** icon  and then click the **Edit** icon .
3. To update the graph, click the **Settings** icon  on the graph. There would be multiple Settings icons for a report.
4. On the View Settings page, click **Counters**, and click the **Add** icon .
5. From the dropdown, select **Resource Monitoring** and then from the **Component** dropdown, select a unit of measurement such as Min, Max, or Average.
6. Click **Apply**, click **Save**, and then click the **Edit** icon.

Results




The changes are reflected in the resource monitoring graph.

Filtering resource counters

To view the performance data that is more important for you, filter the resource monitoring counters. You can filter resource counters the same way you filter other results displayed in reports.

Before you begin

To view the Resources page in the Performance Report, you must have enabled capturing of resource monitoring data in the schedule. See [Enable resource monitoring](#).

1. Open the Performance Report, and from the list of Pages, click **Resources**.
2. Click the **Menu** icon  and then click the **Edit** icon .
3. To update the graph, click the **Settings** icon  on the graph. There would be multiple Settings icons for a report.
4. On the View Settings page, click **Filters**.
5. From the dropdown, click one of the following filter types:

Option	Description
Filter by count	Display the specified number of items. For example, if you select this option and then type 15, the report will show the 15 items with the highest values (Show highest values) or the 15 items with the lowest values (Show lowest values).
Filter by value	Display items based on a comparison with the specified value. For example, if you select this option and then type 15, the report will show all of the items that are higher than 15 (Show counters above filter value) or lower than 15 (Show counters below filter value).
Filter by label	Display items that match the specified label. If you are filtering a table, the label is typically a page, and is listed in the left column. If you are filtering a graph, the label is a legend in the graph.

6. From the **Primary counter for table filtering** dropdown, click a primary resource counter. Typically, this is the **Resource Monitoring** counter.
7. Click **Apply**, click **Save**, and then click the **Edit** icon.

Results

The changes are reflected in the resource monitoring graph.

Reports and counters

After the tests or schedules are run and completed, you can view the results and open the reports to analyze the verdict, the performance, and statistics.

Requirements report

The Performance Requirements report is renamed to Requirements report. The Requirements report validates the performance and functional requirements that you set in a test or in a schedule.

Validation is accomplished by comparing the data in the run to the requirements that you set on the data.

Status Summary page

With the Status Summary page, you can quickly analyze the requirements that are defined in a test result. The page contains two tables.

- The first table provides the overall status of the run and the percentage of performance requirements that passed.
- The second table lists all of the standard requirements that you defined.

Each requirement has a row in the table that explains the target of the performance requirement (for example, an HTTP page), specification, and status.

Overall Summary page

The Overall Summary page provides a high-level, graphical analysis of requirements, both standard and supplemental. The Overall Summary page contains two tables and two pie charts.

The first table presents the pass/fail status and percentage pass data for requirements and supplemental requirements.

Both requirements and supplemental requirements are represented in the center of the report by a pie chart. In both cases, failures are shown in red and passes are shown in green.

The final graphic on this page presents numeric details for each requirement type and for the union of both types.

Details page

The Details page provides a detailed analysis of standard requirements. It contains two tables and a bar chart.

As with the Status Summary page, the overall status of the run and percent passed value for the run are presented in a table at the top of the page. The bar chart at the center of the report displays the margin of each performance requirement. The *margin* is a percentage value that allows improvement or regression analysis with regards to a requirement. When a performance requirement is in the passed state, it will have a margin ≥ 0 .

The margin is calculated as follows:

$$\% \text{ Margin} = \text{abs}(\text{specification} - \text{observed})/\text{specification}$$

When a performance requirement is in the "failed" state, it will have a value ≤ 0 . The margin is calculated as follows:

$$\% \text{ Margin} = -\text{abs}(\text{specification} - \text{observed})/\text{specification}$$

Margins are useful in comparison mode because you can detect improvement or regressions before they are significant enough to change the requirement from pass to fail. The table at the bottom of the Details page contains a representative row for each performance requirement. Each requirement is presented with an explanation of its target, observed result, specification, margin, and status. Passed statuses are shown in green and failed statuses are shown in red.

Supplemental Details page

The Supplemental Details page shows a detailed analysis of supplemental requirements. It contains the same data as the Details page, except that it pertains to supplemental requirements.

Synchronization Point report

This report provides information about the synchronization points in test runs and lets you manually release a user from a synchronization point. To release a user, right-click in the report and select **Manage Synchronization Points**.

The Synchronization Point report contains the following information for each synchronization point in the run:

- The name of the synchronization point.
- Time-out Value. The time after which the synchronization point is automatically released. A value of 0 means that the synchronization point will be released after the arrival of the first user.
- Users Late. The number of virtual users that have arrived "late"; that is, after the synchronization point was released.
- Users Arrived. The number of virtual users that have arrived at the synchronization point.
- Current® State. The state of the synchronization point. The state can be:
 - Inactive. No users have arrived or all locations are inactive.
 - Active. At least one user has arrived.
 - Released. All users have arrived, been release manually, or have timed out.
- Users Expected. The number of virtual users still expected to arrive.
- Run Duration. The time between the first user's arrival and the first user's release. The count begins as soon as any location reports an Active state, and stops as soon as any location reports a released state. The accuracy of the time might be adversely affected by a large statistics interval.

Loops report

This report summarizes the functionality of loops in a test.

Loop Invocation Details

This tab displays the number of times the loop was invoked in a test and loops that were invoked but did not complete.

Loop Iteration Details

This tab displays how many iterations does each loop run and how many iterations in a loop were successful.

Loop Iteration Health

This tab displays the health of each iteration of a loop. You can also view the error that cause the failure of an iteration of a loop.

Agents Health Report

With the Agents Health Report, you can view the usage data of CPU, Memory, Threads, and JVM Heap for the agent machines involved in the run. The report shows usage data for the Agent Host and Agent Process on the agent

machines. By default, the **Enable Agent Health Report** check box at **Window > Preferences > Test > Test Execution** is selected. Also, by default, the report opens in the Compare mode. You cannot disable the compare mode.

Report

CPU

The CPU usage graph shows the usage data for the Agent Host and Agent processes. Note that the SAP, SOA, and Rational® Integration Tester test extensions runs as external processes on the agent locations and therefore are not counted in the Agent process but are counted in the Agent Host with all others system processes (including the agent).

Memory

The Memory usage graph shows the memory consumed by the Agent processes in terms of average and percentage and by the Agent Host. The average memory consumption of the agent is usually very little. Therefore, the graph also displays the memory consumption in terms of percentage. If the CPU and Memory consumption for the agents is high, you can decide to add more locations to the execution of the performance tests.

JVM Heap

The JVM Heap graph shows the percentage of memory used compared to the total memory allocated and the total JVM memory allocated. If the memory used is high, you can decide to increase the JVM Heap size by adding the `-Xmx` property to the agent location.

Threads

Threads Usage graph shows the percentage of threads used for the currently running tasks compared to the maximum of threads that the tool can create. It also shows the number of tasks waiting for the threads to be made available to execute them.

Rate Runner report

You can use the Rate Runner report to view how all the Rate Runner groups in the Rate Schedule have performed during the run.

Rate Runners

The **Rate Runners** page displays the graph with X-axis showing the time and Y-axis showing the rate or workload generated per second. The top section of the report displays the following fields:

State: Status of the run for rate generators. One of the following statuses is displayed:

- Not started
- Arriving: At least one user has arrived at the synchronization point.
- Active: At least one iteration of workload.
- Inactive: No users or clients are running.

Users or Clients: Number of users or clients in the Arriving or Active state.

Target Rate: Number of iterations specified in Rate Generator or Rate Schedule. The report shows the value in seconds. For example, if the iteration rate is 2 every minute, the Target Rate would show 0.032 per second.

Workloads Started - First time: Time taken for the first workload to start after the test run. The time taken includes the time to create the users or clients, begin the workload, meet at the rate synchronization point, run any iteration delay, and then start the workload.

Workloads Started - Elapsed time: Time taken for the first workload to start and the last workload to end.

Workloads Started: Number of workloads already started at any given point of time during the run. For example, for a target rate of 4 every minute, at the end of one minute, the number of workloads started should be 4. However, it should increment throughout the run based on the target rate. So, at a target rate of 4 every minute and a total duration of 10 minutes, the number of workloads at the end of the run should show `Workloads Started = 40` and `Workloads Completed = 40`.

Workloads Started - Rate: Actual rate achieved for the run. Compare it with the Target Rate to determine the performance of the application.

Percent Target: Comparison of the Target Rate with the Actual Rate (Workloads Started - Rate) to indicate by percentage how close the workload is to reach the specified target rate.

Percent Late - Percent: The percent of workloads in a user group or Rate Runner group that did not begin to run at the specified time.

Health

Use this page to view the number of workloads that did not meet the target rate.

Throughput

Use this page to view the number of workloads that passed successfully.

Transaction report

This report summarizes the success of transactions in the run, plots on a graph the response trend of each transaction during a specific sample interval, and shows the transaction throughput.

This report plots the sample intervals within a run. You set the **Statistics sample interval** value in the schedule, as a schedule property.

Overall page

The **Overall Transaction Rate** graph shows the average elapsed time for all transactions during a specific sample interval. *Elapsed time* is the actual time spent within the transaction container. If you have staged loads in the schedule, this graph also delineates the stages with time range markers, in various colors, at the top.

The table under the graph lists the following information:

- The average elapsed time for all transactions in the entire run.
- The standard deviation of the elapsed time. The standard deviation tells you how tightly the data is grouped about the mean. For example, assume that *System A* and *System B* both have an average elapsed time of 12 ms. However, this does not mean that the elapsed times are similar. *System A* might have elapsed times of 11, 12, 13, and 12 ms. *System B* might have elapsed times of 1, 20, 25, and 2 ms. Although the mean time is the same, the standard deviation of *System B* is greater and the elapsed time is more varied.
- The longest transaction in the entire run.
- The shortest transaction in the entire run.
- The average net server time for all transactions in the entire run.
- The standard deviation of the net server time.
- The longest net server time in the entire run.
- The shortest net server time in the entire run.

Elapsed Time vs. Time

The **Elapsed Time vs. Time** graph shows the average response of each transaction during a specific sample interval. Each transaction is represented by a separate line. If you have staged loads in the schedule, this graph also delineates the stages with time range markers, in various colors, at the top.

The table under the graph lists the following information for each transaction:

- The minimum elapsed time for the entire run.
- The average elapsed time for the entire run. This is similar to the graph above, but the information in the table is the average for the entire run rather than the average per sample interval.
- The standard deviation of the average response time. The standard deviation tells you how tightly the data is grouped about the mean. For example, *System A* and *System B* both have an average response time of 12 ms. However, this does not mean that the response times are similar. *System A* might have response times of 11, 12, 13, and 12 ms. *System B* might have response times of 1, 20, 25, and 2. Although the mean time is the same, the standard deviation of *System B* is greater and the response time is more varied.
- The maximum elapsed time for the entire run.
- The rate, per second, at which the transaction was completed.
- The number of attempts for the transaction.

Net End-to-End time vs. Time

Net end to end time for a transaction is a measured time of interactions with the server and a client such as a browser or a device. Typically, this does not include think times or processing time by the workbench.

Net Server Time vs. Time

Net server time for a transaction is a measured time of interactions with the server. Typically, this does not include think times or processing time by the product. The server interaction calculation is protocol specific. For example, in HTTP protocol, the net server time is exactly the sum of all page response times. The elapsed time (wall clock time) includes think time and other product processing overhead. For a Web UI test, the net server time includes the sum of time spent on the server and network.

Transaction Throughput

These line graphs provide an overview of the transaction frequency and the number of users that are adding load, both over the course of a run.

- The **Transaction Hit Rate** graph shows the overall rates for starting and completing transactions during a specified sample interval. If you have staged loads in the schedule, this graph also delineates the stages with time range markers, in various colors, at the top. The summary table under the graph lists the transaction rate per second and the number of transactions that were completed for the entire run.
- The **User Load** graph shows the number of active users and the number of users that completed testing over the course of the run. The summary table under the graph lists the number of active users, the number of users that completed testing, and the total number of users for the latest sample interval.

Transaction Health

The **Transaction Health** page displays whether the transaction is healthy or unhealthy. When you define error conditions for a transaction, you can set whether the behavior of the errors affect the health of the run. If any one condition is met in a transaction, the transaction is marked unhealthy. The **Transaction Health** page displays the percentage of healthy and unhealthy transactions. The graphic bar is displayed only if there are no errors.



Note:

During the test run, the elapsed time is recorded for unhealthy transactions. You can select the **Discard time measurements for unhealthy transactions** check box from the **Statistics tab** of **VU Schedule Details**, if you do not want to record the elapsed time for the unhealthy transactions.

Transaction Details

The **Transaction Completion Percentage** graph in the **Transaction Details** page displays the overall percentage of the successful transactions. The Transaction Details section shows the transactions that were attempted, completed, exited, and the percentage of completed transactions.

Transaction Percentile report

This report shows the 85th, 90th, and 95th percentile elapsed times for all users, the union of all transactions in a run, and for the 10 slowest transactions in a run.

The default percentiles in this report, 85, 90, and 95, are sufficient for most purposes. If you need a report on a different percentile set, click **Window > Preferences > Test > Percentile Analysis Targets** to change the percentiles in this report and in the Page Percentile report.

The Summary page of this report has a graph with three bars, which represent the 85th percentile, 90th percentile, and 95th percentile elapsed times for all users and for all transactions in the run. For the 85th percentile bar, 85% of all users achieved the indicated elapsed time or better. For the 90th percentile bar, 90% of all users achieved the indicated elapsed time or better. And for the 95th percentile bar, 95% of all users achieved the indicated elapsed time or better.

The 85%, 90%, and 95% pages show the elapsed time percentiles of the 10 slowest transactions in the run. For example, if you click the tab for the 85th percentile, and the total for a transaction is 110 (the total is beneath each bar), you know that 85 percent of the elapsed times for that transaction are less than or equal to 110 milliseconds (ms).

This graph provides an overall idea of the elapsed times for each transaction. For example, the Transaction report might indicate that a login transaction is one of the 10 slowest transactions. However, it is possible that only one instance of the login transaction was extremely slow while the other instances of the login transaction were within acceptable range. The Transaction Percentile report shows which transactions have slow elapsed time averages because they were slow in general, not because a few elapsed times (out of many) were extremely slow.

The table beneath the graph provides the following information for each transaction:

- The minimum elapsed time for the run.
- The average elapsed time for the run.
- The standard deviation of the elapsed time. The standard deviation tells you how tightly the data is grouped about the mean. For example, assume that *System A* and *System B* both have an average elapsed time of 12 ms. However, this does not mean that the elapsed times are similar. *System A* might have elapsed times of 11, 12, 13, and 12 ms. *System B* might have elapsed times of 1, 20, 25, and 2 ms. Although the mean time is the same, the standard deviation of *System B* is greater and the elapsed time is more varied.
- The maximum elapsed time for the run.
- The 85th percentile for the run. That is, for this particular transaction, 85% of the elapsed times were equal to or faster than this time.
- The 90th percentile for the run. That is, for this particular transaction, 90% of the elapsed times were equal to or faster than this time.
- The 95th percentile for the run. That is, for this particular transaction, 95% of the elapsed times were equal to or faster than this time.
- The number of attempts in the run.

Transaction Net Server Time Percentile report

This report shows the 85th, 90th, and 95th percentile net server times for all users and the union of all transactions in a run. The report also shows the 10 slowest transactions in a run. The net server time corresponds to the cumulative server response times within a transaction. Net server time does not include think times and delays, which are included in the elapsed time.

The default percentiles in this report, 85th, 90th, and 95th, are sufficient for most purposes. However, if you must report on a different percentile set, click **Window > Preferences > Test > Percentile Analysis Targets** to change the percentiles in this report and in the Page Percentile report.

The Summary page of this report has a graph with three bars, which represent the 85th percentile, 90th percentile, and 95th percentile net server times for all users and for all transactions in the run. For the 85th percentile bar, 85% of all users achieved the indicated net server time or better. For the 90th percentile bar, 90% of all users achieved the

indicated net server time or better. And for the 95% percentile bar, 95% of all users achieved the indicated net server time or better.

The 85%, 90%, and 95% pages show the net server time percentiles of the 10 slowest transactions in the run. For example, if you click the tab for the 85th percentile, and the total for a transaction is 110 (the total is beneath each bar), you know that 85 percent of the net server times for that transaction are less than or equal to 110 milliseconds (ms).

This graph provides an overall idea of the net server times for each transaction. For example, the Transaction report might indicate that a login transaction is one of the 10 slowest transactions. However, it is possible that only one instance of the login transaction was extremely slow while the other instances of the login transaction were within acceptable range. The Transaction Net Server Time Percentile report shows which transactions have slow net server time averages because they were slow in general, not because a few net server times (out of many) were extremely slow.

The table beneath the graph provides more detailed information for each transaction:

- The minimum net server time for the run.
- The average net server time for the run.
- The standard deviation of the net server time. The standard deviation tells you how tightly the data is grouped about the mean.
- The maximum net server time for the run.
- The 85th percentile for the run. That is, for this particular transaction, 85% of the net server times were equal to or faster than this time.
- The 90th percentile for the run. That is, for this particular transaction, 90% of the net server times were equal to or faster than this time.
- The 95th percentile for the run. That is, for this particular transaction, 95% of the net server times were equal to or faster than this time.
- The number of attempts in the run.

Rate Generator report

Use this report to determine if the transactions run at the rate that you specify.

The report contains three pages. See the description of each of the page.

Rate Generators

State: Status of the run for rate generators. One of the following statuses is displayed:

- Not started
- Arriving: At least one user has arrived at the synchronization point.
- Active: At least one iteration of workload.
- Inactive: No users or clients are running.

Users or Clients: Number of users or clients in the Arriving or Active state.

Target Rate: Number of iterations specified in Rate Generator or Rate Schedule. The report shows the value in seconds. For example, if the iteration rate is 2 every minute, the Target Rate would show 0.032 per second.

Workloads Started - First time: Time taken for the first workload to start after the test run. The time taken includes the time to create the users or clients, begin the workload, meet at the rate synchronization point, run any iteration delay, and then start the workload.

Workloads Started - Elapsed time: Time taken for the first workload to start and the last workload to end.

Workloads Started: Number of workloads already started at any given point of time during the run. For example, for a target rate of 4 every minute, at the end of one minute, the number of workloads started should be 4. However, it should increment throughout the run based on the target rate. So, at a target rate of 4 every minute and a total duration of 10 minutes, the number of workloads at the end of the run should show `Workloads Started = 40` and `Workloads Completed = 40`.

Workloads Started - Rate: Actual rate achieved for the run. Compare it with the Target Rate to determine the performance of the application.

Percent Target: Comparison of the Target Rate with the Actual Rate (Workloads Started - Rate) to indicate by percentage how close the workload is to reach the specified target rate.

Percent Late - Percent: The percent of workloads in a user group or Rate Runner group that did not begin to run at the specified time.

Health

Use this page to view the number of workloads that did not meet the target rate.

Throughput

Use this page to view the number of workloads that passed successfully.





Related information

[Creating rate generators in user groups](#)

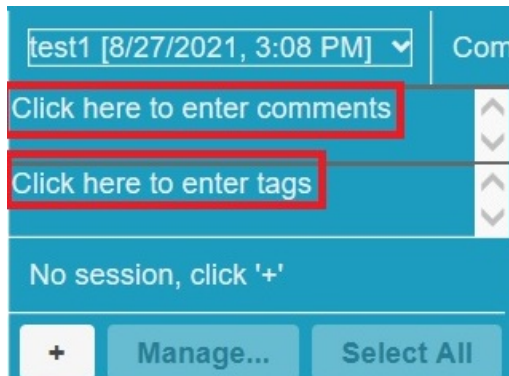
HTTP performance test reports


When you test an HTTP system, reports are produced during a run and saved after a run. You can then analyze the reports to know the performance of the system under test.

In a performance report, you can sort the order of HTTP pages that are captured in a test or schedule either by alphabetical order or order of execution of the HTTP pages. The default sorting of the HTTP pages is by order of execution.

You can either click the **Execution order** icon  or the **Alphabetical order** icon  to toggle between the sorting options. You can also click the Up Arrow icon  to sort the HTTP pages in the correct order of execution or the correct alphabetical order. Similarly, you can click the Down Arrow icon  to sort the HTTP pages in the reverse order of execution or the reverse alphabetical order.

If you want to identify specific test results, you can enter a tag or comment in the corresponding fields of the performance report to associate it with the test result as shown in the following image:



After the test run is complete, you can open the test results. You can click the **Copy** icon  to copy the contents of the table in the HTML format from the report. You can then paste the copied content into Microsoft Excel or any other application, such as browsers or scripts where you want to use the report information.

You can view the following pages in the Performance report:

- [Overall page on page 482](#)
- [Summary page on page 484](#)
- [Page Performance page on page 487](#)
- [Page Details on page 488](#)
- [Page Element Details on page 490](#)
- [Response vs. Time Summary page on page 491](#)
- [Response vs. Time Detail page on page 492](#)
- [Page Throughput page on page 493](#)
- [Server Throughput page on page 494](#)
- [Server Health Summary page on page 494](#)
- [Server Health Summary page on page 494](#)
- [Caching Details page on page 495](#)
- [Resources page on page 495](#)
- [Page Element Responses on page 498](#)
- [Page Response Time Contributions on page 498](#)
- [Page Size on page 498](#)
- [Errors on page 498](#)
- [Page Health on page 499](#)

Performance report

The performance report summarizes important information about the validity of the run and the data that is most significant during the test run. You can view a collection of performance reports. The report also shows the response trend of the pages in the test run and the graph of the response trend of each page for a specified interval.

List of the terms

The following table provides definitions of the list of the terms used in the reports

Term	Description
Attempt	The request sent from the client application to the server until the requested page is displayed successfully.
Hit	The number of requests sent by the virtual users to the server.
Response time	The time taken by the server to respond to the request sent from client application. It is the time from when the request is sent till the page is successfully downloaded.
Standard deviation	The standard deviation indicates whether the page was loaded consistently across all virtual users in the test. A smaller standard deviation indicates that the page loaded consistently while a larger standard deviation indicates the page loaded with different times with a larger degree of variance.
Throughput	The number of requests responded successfully per unit of time.
Page health	The number of pages that loaded successfully from the attempts made, the page is considered healthy.

Overall page

An overview of the overall performance of the pages. The Overall page provides the following information:

- A progress indicator that displays the status during the playback.
- A pie chart displays the information about the overall verification point passed and failed in the playback, if they are set. For the schedule run, it displays the overall requirements that passed and failed, including success rate in percentage for:
 - Page Verification Points (VPs) Status
 - Page Element VPs
 - Page Status Code

- Page Element Status Code
- Page Health

Option	Description	Verdict
Page VPs	Displays the verdict of the page title verification points if the verification points are set.	<ul style="list-style-type: none"> • Passed • Failed • Inconclusive • Error
Page Element VPs	Displays the verdict of the response code or response size verification points if the verification points are set.	<ul style="list-style-type: none"> • Passed • Failed • Inconclusive • Error
Page Primary Request Status Codes	<p>Displays the success and failure status in the playback.</p> <p>If a primary request includes verification points, the Page Status Code Successes value indicates that the verification point for the response code is passed.</p> <p>If a primary request has no verification points, the Page Status Code Successes value indicates that the primary request is received, and a response is returned with a status code.</p>	<ul style="list-style-type: none"> • Passed • Failed
Page Element Status Codes	<p>Displays the success and failure rate in the playback.</p> <p>If a primary request includes verification points, the Page Element Successes value indicates that the response code verification point passed for that request.</p> <p>If there are no verification points included in a request, the Page Element Successes indicates that the request is received and returned a response with a status code.</p>	<ul style="list-style-type: none"> • Passed • Failed
Page Health	Displays the total health of the pages, transactions, and loops in the playback or schedule run.	<ul style="list-style-type: none"> • Healthy • Unhealthy

- If you click any individual chart, you can go to that specific report to analyze the status in detail.
- If you click any legend for example, Passed, the chart is updated to show only the other verdicts of a test or a schedule run. For example, the Page Status Codes has a legend as Passed and Failed. If you click Passed, the chart is updated to show only errors in the test run or a schedule run.
- Similarly, if you double-click any legend, the chart is updated to display only the selected verdict by removing all other verdicts from the chart. Thus, you can focus on only one counter which you want to investigate in detail.

Summary page

The **Summary** page displays a summary of the page related information that were played back during the test. You can view the details of the test run and can analyze the final or intermediate results of a test quickly.

The **Summary** page displays the following `Run Summary` information:

Run Summary	Description of the value that is displayed
Parameter	
Active	The number of virtual users that are active
Completed	The number of users completed the test run.
Executed Test	The name of the test that were played back during the test.
Elapsed Time	The total time taken to complete a test run. The elapsed time is displayed in hours, minutes, and seconds.
Run Status	The status of the test that were played back. The Run status can be <code>Initializing Computers, Adding Users, Running, Transferring data to test log, Stopped, OR Complete.</code>
Requirement - Status	The status of the requirement. The requirement status indicates Passed or Failed status in the playback.
Requirement- Percent Passed	The percentage passed data for requirements in the playback.
Total Users	The total number of users in the test run.
Total Error Behaviors	The number of error behaviors that occurred in the test run.
Total Error Conditions	The number of error conditions that occurred in the test run.

The **Summary** page displays the following `Page Summary` information:

Page Summary	
Parameter	Description
Percent Page VPs Passed	The percentage of verification points that passed for a page in the playback.
Response Time For All Pages - Average	The average time that was taken for the pages to load successfully in the playback.
Response Time For All Pages - Standard Deviation	<p>The standard deviation of the time taken for all the pages to load successfully in the playback.</p> <p>The standard deviation indicates whether the page was loaded consistently across all virtual users in the test. A smaller standard deviation indicates that the page loaded consistently while a larger standard deviation indicates the page loaded at different times with a larger degree of variance.</p>
Response Time For All Pages - Max	The maximum time that was taken for all pages to load successfully in the playback.
Response Time For All Pages - Min	The minimum time that was taken for all pages to load successfully in the playback.
Total Page VPs Passed	The number of verification points that passed for the page in the playback.
Total Page VPs Failed	The number of verification points that failed for the page in the playback.
Total Page VPs Inconclusive	The number of verification points that was inconclusive for the page in the playback.
Total Page VPs Error	The number of page verification that failed and indicates an error in the playback.
Total Page Attempts	The number of requests sent from the client application to the server until the requested page is displayed successfully.
Total Page Hits	The number of pages that were loaded successfully.
Total Percent Healthy For All Pages	The number of pages that loaded successfully from the number of attempts made is expressed in percentage.
Total Count Healthy For All Pages	The count of all pages that loaded successfully from the number of attempts made.

Page Summary	
Parameter	Description
Total Count Unhealthy For All Pages	The count of all pages that were not loaded successfully from the number of attempts made.

The **Summary** page displays the following `Page Element Summary` information:

Page Element Summary	
Parameter	Description
Response Time For All Page Elements - Average	The average time that was taken to load the elements on all the page in the playback.
Response Time For All Page Elements - Standard Deviation	The standard deviation of the response time for all page elements. The standard deviation informs whether the request sent is consistent. If the standard deviation is smaller, then the response time is more consistent, and if the standard deviation is greater, the response time is more varied.
Total Page Element VPs Passed	The percentage of verification points that passed for the element on the page in the playback.
Total Page Element VPs Failed	The number of verification points that failed for the elements on the page in the playback.
Percent Page Element VPs Passed	The number of verification points that passed for the elements on the page in the playback.
Total Page Element VPs Inconclusive	The number of verification points was inconclusive for the elements on the page in the playback.
Total Page Element VPs Error	The number of verification points that failed indicates an error for the elements on the page in the playback.
Total Page Element Attempts	The number of requests sent from the client application to the server until the requested page element is displayed successfully.
Total Page Element Hits	The number of elements on the page that were loaded successfully.
Total Page Elements Fresh In Cache, Not Sent to Server	The number of elements on the page where no request was sent as the page elements were fresh in the local cache.

If you have set transactions in your test, the **Summary** page displays the following `Transaction` information:

Transaction	
Parameter	Description of the value that is displayed
Elapsed Time For All Transactions - Average	The average elapsed time that was taken for all transaction in the playback.
Elapsed Time For All Transactions - Max	The maximum elapsed time that was taken for all transaction in the playback.
Elapsed Time For All Transactions - Min	The minimum elapsed time that was taken for all transaction in the playback.
Elapsed Time For All Transactions - Standard Deviation	The standard deviation of the elapsed time for all transactions in the playback.
Total Transactions Completed [for Run]	The number of transactions completed in the playback.
Total Transactions Started [for Run]	The number of transactions started in the playback.

Page Performance page

The **Page Performance** page displays the average response of the slowest primary pages in the test as the test progresses. The page performance displays response for each page and name of each page with the corresponding response time. You can evaluate the response for each page request during and after the test.

The bar chart displays the average response time of the slowest pages. Each bar represents a page that you attempted during playback. When you play back, the bar chart changes, the slowest pages are updated dynamically during the playback. For example, the login page might be one of the slowest pages at the start of the run, but then, as the test progresses, the Shopping Cart page might replace it as one of the slowest. After the run, the page displays the slowest pages for the entire run.

The table that follows after the bar chart provides the following additional information:


Parameter	Description of the value that is displayed
Attempts - Rate	The number of attempts out of the total number of attempts to get a successful page displayed is expressed in percentage. For example, if the requested page was displayed for 5 of the 10 virtual users, the rate of page attempts completed is displayed as 50%.
Attempts	The number of requests sent from the client application to the server until the requested page is displayed successfully.

Parameter	Description of the value that is displayed
	For example, if the requested page was displayed after the second request, the number of attempts displayed is 2.
Page	The name of the page in the recording that was attempted in the playback.
Page Response Time - Average	The average time that was taken for each page to load successfully in the playback.
Percent Healthy Page	The number of pages that loaded successfully from the number of attempts made is expressed in percent.
Page Response Time - Min	The minimum time that was taken for each page to load successfully in the playback.
Page Response Time - Max	The maximum time that was taken for each page to load successfully in the playback.
Page Response Time - Standard Deviation	<p>The standard deviation of the time taken for each page to load successfully in the playback.</p> <p>The standard deviation indicates whether the page was loaded consistently across all virtual users in the test. A smaller standard deviation indicates that the page loaded consistently while a larger standard deviation indicates the page loaded at different times with a larger degree of variance.</p>

Page Details

The **Page Details** report displays a summary of the statistics of the pages that were played back during the test. You can view the details of the performance of each page.

You can copy the data of the table in the HTML format from the report and use the data in other reports or other

applications by using the Copy icon .


The Page Details table displays the following information:

Parameter	Description of the value that is displayed
Attempts	<p>The number of requests sent from the client application to the server until the requested page is displayed successfully.</p> <p>For example, if the requested page was displayed after the second request, the number of attempts displayed is 2.</p>
Attempts Completed - Rate	<p>The number of attempts out of the total number of attempts to get a successful page displayed is expressed in percentage.</p> <p>For example, if the requested page was displayed for 5 of the 10 virtual users, the rate of page attempts completed is displayed as 50%.</p>
Connection Time - Average	<p>The average time that was taken to successfully display the page when requests were sent by all virtual users. The connection time indicates whether the requests are responded, and it is generally used to identify the delay.</p> <p>For example, if the time to successfully display the page by the 5 virtual users were 100 ms, 100 ms, 150 ms, 150 ms, and 200 ms, the average connection is displayed as 140 ms.</p>
Client Delay Time - Average	<p>The average time taken by the client application during the test to process the data before sending out the next request.</p>
Fail	<p>The number of verification points that failed.</p>
Hits	<p>The number of pages that were loaded successfully.</p>
Page	<p>The name of the page in the recording that was attempted in the playback.</p>
Page Response Time Average	<p>The average time that was taken for each page to load successfully in the playback.</p>
Page Element Response Time -Average	<p>The average time that was taken for all elements to be loaded successfully on the page in the playback.</p>
Page Response Time - Min	<p>The minimum time that was taken for each page to load successfully in the playback.</p>
Page Response Time - Max	<p>The maximum time that was taken for each page to load successfully in the playback.</p>
Page Response Time - Standard Deviation	<p>The standard deviation of the time taken for each page to load successfully in the playback.</p> <p>The standard deviation indicates whether the page was loaded consistently across all virtual users in the test. A smaller standard deviation indicates that</p>

Parameter	Description of the value that is displayed
	the page loaded consistently while a larger standard deviation indicates the page loaded at different times with a larger degree of variance.
Percent Healthy Page	The number of pages that loaded successfully from the number of attempts made is expressed in percent.
Page Response Time - Percentile/90	The time taken to load the page successfully in the 90 th percentile.
Page Response Time - Percentile/95	The time taken to load the page successfully in the 95 th percentile.
Page Response Time - Percentile/99	The time taken to load the page successfully in the 99 th percentile.

Page Element Details

The **Page Element Details** report displays a summary of the statistics of page elements that were played back during the test. You can view the details of the performance of each page elements.

You can copy the data of the table in the HTML format from the report and use the data in other reports or in other applications by using the Copy icon  .

The Page Element Details table displays the following information:

Parameters	Description of the value that is displayed
Attempts	The number of requests sent from the client application to the server until the requested page element is displayed successfully. For example, if the requested page element was displayed after the second request, the number of attempts displayed is 2.
Fail	The number of verification points that failed to load successfully.
Hits	The number of elements on the page that were loaded successfully.
Page	The name of the element on the page in the recording that was attempted in the playback.
Page Element	The name of the element on the page that is attempted to be loaded in the playback.
Page Element Response Time - Average	The average time that was taken to load the element on the page during the playback.

Parameters	Description of the value that is displayed
Page Element Response Time - Min	The minimum time that was taken to load the element on the page during the playback.
Page Element Response Time - Max	The maximum time that was taken to load the element on the page during the playback.
Page Element Response Time - Standard Deviation	<p>The standard deviation of the time taken for each page element to load successfully in the playback.</p> <p>The standard deviation indicates whether the element on the page was loaded consistently across all virtual users in the test. A smaller standard deviation indicates that the element on the page loaded consistently while a larger standard deviation indicates the page element loaded with different times with a larger degree of variance.</p>
Page Element Response Size - Average	The average size of the element content that is sent by the server to the client application. The unit of the size is displayed in bytes.
Page Element Response Time - Percentile/90	The time taken to load the element on the page successfully in the 90 th percentile.
Page Element Response Time - Percentile/95	The time taken to load the element on the page successfully in the 95 th percentile.
Page Element Response Time - Percentile/99	The time taken to load the element on the page successfully in the 99 th percentile.

Response vs. Time Summary page

The **Response vs. Time Summary** page displays the average response trend for a specified interval displayed in the form of graph. The page displays two-line graphs with corresponding summary tables.

Graph	Description
The Page Response vs. Time	The graph displays the average time that was taken for the pages to load successfully in the playback. Each point on the graph is an average of what has occurred during that interval.
Page Element Response vs. Time	The graph displays the average time that was taken for the page element to load successfully in the playback. Each point on the graph is an average of what has occurred during that interval.

The table after the graph provides the following additional information:

<p>Graph</p> <p>Summary</p> <p>Performance Summary</p>	<p>Description</p> <p>Description</p> <p>The summary of average response time and the standard deviation for all pages in the playback.</p> <p>The summary of average response time for all page elements and the standard deviation in the playback. The table also lists the total number of page elements where no request was sent to the server because the client determined that the page elements were fresh in the local cache.</p>
--	--

Response vs. Time Detail page

The **Response vs. Time Detail** page displays the response trend in the form of graph for a specific interval.

<p>Graph</p> <p>Average Page Response Time [for Interval]</p>	<p>Description</p> <p>The graph displays the average time that was taken for each page for a sample interval to load successfully in the playback. When a schedule includes staged loads, the colored time-range markers, in the graph displays the stages.</p>
<p>The table after the graph provides the following additional information:</p>	
<p>Parameter</p> <p>Attempts</p> <p>Attempts - Rate</p> <p>Page Response Time - Average</p> <p>Page Response Time - Min</p> <p>Page Response Time - Max</p>	<p>Description of the value that is displayed</p> <p>The number of requests per interval from the client application to the server until the requested page is displayed successfully.</p> <p>The number of attempts out of the total number of attempts to get a successful page displayed.</p> <p>The average time that was taken for each page to load successfully in the playback.</p> <p>The minimum time that was taken for each page to load successfully in the playback.</p> <p>The maximum time that was taken for each page to load successfully in the playback.</p>

Graph	Description
Page Response Time - Standard Deviation	<p>The standard deviation of the time taken for each page to load successfully in the playback.</p> <p>The standard deviation indicates whether the page was loaded consistently across all virtual users in the test. A smaller standard deviation indicates that the page loaded consistently while a larger standard deviation indicates the page loaded at different times with a larger degree of variance.</p>

Page Throughput page

The **Page Throughput** page provides displays the frequency of requests transferred per sample interval. The line graph displays the Page Hit Rate, and the User Load.

Graph	Description
Page Hit Rate	The graph displays the number of attempts were made per interval for all pages that were loaded successfully.
User Load	The graph displays active users and completed users in the playback.
The table after the graph provides the following additional information:	
Summary	Description
Performance Summary	<p>The Page Rate Hit summary table lists the total hit rates and counts for each page in the run.</p> <p>The User load summary table lists the active, completed, and total users in the playback. As the run nears completion, the number of active users decreases and the number of completed users increases.</p>

If the number of requests and hits are not close, the server might be having trouble keeping up with the workload.

If you add virtual users during a run and view the two graphs, you can monitor the ability of your computer to manage the workload. As the page hit rate stabilizes, even though the active user count continues to be higher and the computer is well-tuned, the average response time naturally slows down. This response time reduction happens because the computer is running at its maximum effective throughput level and is effectively slows the rate of page hits.

Server Throughput page

The **Server Throughput** page lists the rate and number of bytes that are transferred per interval and for the entire run. The page also lists the status of the virtual users for each interval and for the entire run.

<p>Graph</p> <p>Byte Transfer Rates</p> <p>User Load</p>	<p>Description</p> <p>The graph displays the rate of bytes sent and received per interval for all intervals in the playback.</p> <p>The graph displays active users and users completed testing, per sample interval, over the course of a run. You set the Statistics sample interval value in the schedule, as a schedule property. As the run nears completion, the number of active users decreases and the number of completed users increases.</p>
<p>The table after the graph provides the following additional information:</p>	
<p>Summary</p> <p>Performance Summary</p>	<p>Description</p> <p>The Byte Transfer Rates summary table lists the number of bytes were sent and received in the playback.</p> <p>The User Load summary table lists the active, completed and total users in the playback.</p>

The bytes sent and bytes received throughput rate, which is computed from the client perspective, displays how much data Rational® Performance Tester is pushing through your server. Typically, you analyze this data with other metrics, such as the page throughput and resource monitoring data, to understand how network throughput demand affects server performance.

Server Health Summary page

The **Server Health Summary** page gives an overall indication of how well the server is responding to the load.

<p>Graph</p> <p>Page Health</p> <p>Page Element Health</p>	<p>Description</p> <p>The graph displays the number of pages that loaded successfully from the number of attempts made.</p> <p>The graph displays the number of page element that loaded successfully from the number of attempts made.</p>
<p>The table after the graph provides the following additional information:</p>	
<p>Summary</p> <p>Performance Summary</p>	<p>Description</p> <p>The Performance Summary displays the summary of of the pages and elements on the page that were loaded successfully.</p>

Server Health Detail page

The **Server Health Detail** page provides specific details of each page percent status code success for the test run. A successful status code refers to the HTTP response code passing if a verification point is set for that page.

Caching Details page

The **Caching Details** page provides specific details on caching behavior during a test run.


Graph	Description
Caching Activity	The graph displays the total number of page element cache attempts, page element cache hits, and page element cache misses for the run. The values correspond to responses, that indicates whether the content is modified. Additionally, the bar chart displays the total number of page elements in the cache that are skipped for the run. The-I value indicates the cache hits that are still in the local cache, where communication is not necessary.
Page Element Cache Hit Ratios	The graph displays the percentage of cache attempts that indicate confirmed success for the run. Server-confirmed cache hits occur when the server returns a response code. Client-confirmed cache hits occur when the content is still in the local cache and no communication is required.

Resources page

The **Resources** page displays information about all the resource counters that are monitored during the *schedule* run.

You can view the following information as mentioned in the table from the **Resources** page:


If...	Then the Resources page displays...
If you did not add any Resource Monitoring source to a performance schedule	A message that states that you must set up the Resource Monitoring sources to view the resource details.
If you added Resource Monitoring sources to a performance schedule	<ul style="list-style-type: none"> • The Resource Monitoring sources that are monitored during the schedule run. • All resource counters for those Resource Monitoring sources that are monitored during the schedule run. • The Unavailable sources section lists the Resource Monitoring sources that are unavailable or unreachable during the schedule run.

If...	Then the Resources page displays...
	 Note: The Unavailable sources section is displayed only if any of the Resource Monitoring sources are unreachable or unavailable during the schedule run.
<p>If you added Resource Monitoring sources by using labels to a performance schedule</p>	<ul style="list-style-type: none"> • The following information in the Server sources matching the labels set in the schedule (*Source defined in team space) section: <ul style="list-style-type: none"> ◦ Labels and the Resource Monitoring sources associated with those labels that are monitored during the schedule run. ◦ Resource Monitoring sources that are unavailable or unreachable during the schedule run. ◦ An empty array ([]) when you used labels that are not tagged to any Resource Monitoring source in Rational® Test Automation Server. ◦ The asterisk (*) symbol is displayed after the name of the Resource Monitoring source if you add the Resource Monitoring source at the team space level in Rational® Test Automation Server. • All resource counters for the Resource Monitoring sources that are monitored during the schedule run.
<p>If you ran a performance schedule by using the <code>overrideмлabels</code> command from the Rational® Performance Tester command line</p>	<ul style="list-style-type: none"> • The following information in the Server sources matching the labels set with the command-line flag <code>-overrideмлabels</code> (*Source defined in team space) section: <ul style="list-style-type: none"> ◦ Labels that you use to add the Resource Monitoring sources to the schedule for the schedule run. ◦ Resource Monitoring sources associated with those labels that are monitored during the schedule run.

If...	Then the Resources page displays...
	<ul style="list-style-type: none"> ◦ Resource Monitoring sources that are unavailable or unreachable during the schedule run. ◦ An empty array ([]) when you use labels that are not tagged to any Resource Monitoring source in Rational® Test Automation Server. ◦ The asterisk (*) symbol is displayed after the name of the Resource Monitoring source if you add the Resource Monitoring source at the team space level in Rational® Test Automation Server. <ul style="list-style-type: none"> • All resource counters for the Resource Monitoring sources that are monitored during the schedule run.

The Legend displays the Resource Monitoring type and its resource counters. When there are multiple Resource Monitoring sources, the resource counters for the respective sources are displayed in front of their Resource Monitoring source name. You can customize the resource counter information displayed in a graph by clicking any individual resource counter or type of source. You can click or double-click any individual resource counter for the following results:

- A single click on the resource counter hides the data displayed on the graph. Click the resource counter again to display the data in the graph.
- A double-click on the resource counter removes information about all other resource counters from the graph and displays only the information about the selected resource counter.

 **Tip:** You can click **Select All** option to restore all the resource counter information on the graph.

When you click on any of the sources, the graph removes all the resource counters of other sources and displays only the resource counters of the selected source.

For example, you have an Apache httpd server and a Windows Performance Host as a Resource Monitoring source. When the *schedule* completes, the **Resources** page displays the resource counter information of both the sources. If you want to analyze the resource counters for any one of the sources, you can click the Apache httpd server or the Windows Performance Host. Based on your selection, the graph is updated to display the selected source resource counters information.

The Performance Summary table that follows the graph, lists the most recent values of the resource counters that are monitored during the schedule run. The first two columns display the **Type** of the source and **Name** of the resource counter. This table also lists the minimum, maximum, and average values of the resource counters that are monitored during the schedule run.

Page Element Responses

The **Page Element** page displays the slowest page element responses for the selected page.

Page Response Time Contributions

The **Page Response Time Contributions** page displays how much time each page element contributes to the overall page response time and the request delay time and connection time.

Page Size

This page lists the size of each page of your application in the test. The size of the page contributes to the response time. If part of a page or an entire page is cached, then those requests coming from the cache did not contribute to the total page size.

Column	Description
Average Page Size For Run	The bar in the graph represents a page. To view the Page Elements Size report, click a bar and select Page Element Sizes . All the elements that are on the page displays with sizes. The size of a page is mostly determined by the size of its elements.
Page Size Summary	The summary table lists the average response size of the page element, the maximum and minimum response size of the page element, and the number of attempts.

Errors

The **Errors** page lists the number of errors and the corresponding actions that occurred in the test or schedule. You can view the following graphs on the **Errors** page:

- **Error Conditions:** This graph displays the number of errors that the conditions met.
- **Error Behaviors:** This graph displays how each error condition is managed.
- **Error Conditions over Time:** This graph displays errors against the time that occurred during the playback of the test or schedule.



Note: You must define how to manage errors in the **Advanced** tab of the **Test Details**, **VU Schedule Details**, or **Compound Test Details** pane to log errors when a specific condition occurs.

Page Health

Use this page or report to determine if the pages of your application contain errors. If a page contains any error, the report displays that the page is not 100% healthy. If there are pages that are not 100% healthy, the report displays another section that lists such pages and the reported errors.

Page Element report

This report summarizes the most important page element data for the run.

The graphs in this report show time intervals, attempts, hits, and successes.

- The *interval* depends on the **Statistics sample interval** value that you set for the schedule.
- An *attempt* means that a request was sent.
- A *hit* means that the server received the request and returned any response.
- A *success* means that the response code verification point passed for that request. If the request has no verification point, a success means that the server received a request and returned a response where the status code was in the 200 or 300 category, or returned an expected response in the 400 or 500 category.
- The *response time* is the time between the first request character sent and the last response character received. The response time does not include HTTP requests that time out or requests that return a status code in the range of 4XX (client errors) - 5XX (server errors) or requests that timed out. Response times for HTTP requests that time out or that return an unexpected status code (the recorded and played back codes do not match) in the range of 4XX (client errors) to 5XX (server errors) are discarded from the reported values.

Overall page

The Overall line graph shows the average response time for all page elements during a specified interval. When you have set staged loads in the schedule, this graph delineates the stages with time range markers, in various colors, at the top. The table after the graph provides the following information:

- The average response time for all page elements in the entire run
- The standard deviation of the average response time. The standard deviation tells you how tightly the data is grouped about the mean. For example, *System A* and *System B* both have an average response time of 12 ms. However, this does not mean that the response times are similar. *System A* might have response times of 11, 12, 13, and 12 ms. *System B* might have response times of 1, 20, 25, and 2. Although the mean time is the same, the standard deviation of *System B* is greater and the response time is more varied.
- The average number of page elements attempted for the entire run
- The total page element attempts for the entire run

Response vs. Time Summary page

The Response vs. Time Summary line graph shows the response time for the 10 slowest page elements in the run. When you have set staged loads in the schedule, this graph delineates the stages with time range markers, in various colors, at the top. The table after the graph lists the parent page and the page element, and provides the following information:

- The average response time for that page element during the entire run
- The standard deviation of the average response time. The standard deviation tells you how tightly the data is grouped about the mean. For example, *System A* and *System B* both have an average response time of 12 ms. However, this does not mean that the response times are similar. *System A* might have response times of 11, 12, 13, and 12 ms. *System B* might have response times of 1, 20, 25, and 2. Although the mean time is the same, the standard deviation of *System B* is greater and the response time is more varied.
- The attempts per second during the most recent sample interval
- The number of attempts during the most recent sample interval

Response vs. Time Detail page

The Response vs. Time Detail page shows the response time for each page element in the run. The table lists the parent page and page element, and provides the following information for each page element in the entire run:

- The average response time
- The standard deviation of the average response time. The standard deviation tells you how tightly the data is grouped about the mean. For example, *System A* and *System B* both have an average response time of 12 ms. However, this does not mean that the response times are similar. *System A* might have response times of 11, 12, 13, and 12 ms. *System B* might have response times of 1, 20, 25, and 2. Although the mean time is the same, the standard deviation of *System B* is greater and the response time is more varied.
- The number of attempts
- The number of attempts per second

Page Element Throughput page

The Page Element Throughput page shows the average response trend during a specified interval. It contains two line graphs with corresponding summary tables:

- The Page Element Hit Rate graph shows the combined attempt rate and hit rate for all page elements during the last recorded interval. The table after the graph lists one number: the average hit rate for all pages in the run. When a schedule includes staged loads, colored time-range markers at the top of the graph delineate the stages.
- The User Load graph shows active users compared to users that have completed testing. The table after the graph lists the number of active users, the number of users that have completed testing, and the total user count for the entire run.

Server Health Detail page

The Server Health Detail bar chart shows the percentage of successes for the 10 slowest page elements in the run. The table under the chart lists the parent page and page element, and provides the following information for the entire run:

- The number of attempts
- The number of hits
- The number of successes
- The percent of successes (matches the information in the bar chart)
- The number of attempts per second

Caching Details page

The Caching Details page provides specific details on caching behavior during a test run.

- The Caching Activity graph shows the total number of page element cache attempts, page element cache hits, page element cache misses for the run. These values correspond to responses from the server indicating whether the content has been modified. Additionally, the bar chart shows the total number of page elements in cache skipped for the run. That value indicates the cache hits that were still fresh in the local cache, where communication with the server was not necessary.
- The Page Element Cache Hit Ratios graph shows the percentage of cache attempts that indicate server-confirmed success and client-confirmed success for the run. Server-confirmed cache hits occur when the server returns a 304 response code. Client-confirmed cache hits occur when the content is still fresh in the local cache and no communication with the server is required.
- The summary table under the charts lists the the total number of page elements found fresh in the cache for the run. This value indicates the cache hits that were still fresh in the local cache, where communication with the server was not necessary. The table also lists the number of attempts, the total number of page element cache attempts sent to the server, and the total number of page element cache hits confirmed by the server for the run. Additionally, the table lists the percentage of cache attempts that indicate server-confirmed success and client-confirmed success for the run.

Why response time of a page does not equal the sum of its requests

The response time for a page typically differs from the sum of its requests. This does not mean that your data is incorrect. The difference can be caused by concurrent requests, page connection times, inter-request delays, and custom code within a page.

The most common reason for the sum of the individual request times within a page to exceed the total page response time is that requests are often sent concurrently (in parallel) to a server. Thus some of the individual request response times overlap so the sum of the request response times would exceed the page response time.

Additionally, the page response time can exceed the sum of the individual request response times within the page for the following reasons:

- The individual request response times do not include time to establish connections but the page response time does include the connection request time.
- Inter-request delays are not reflected in the individual request response time but are reflected in the page response time.
- Custom code placed within a page is executed serially (after waiting for all previous individual requests to complete) and thus contributes to the page response time. It does not affect individual request response times. However, we recommend that you place custom code outside of a page, where it will not affect page response time. For more information, see [Reducing the performance impact of custom code on page 407](#).

Page Percentile report

This report shows the 85th, 90th, and 95th percentile response times for all users and all pages in a run, as well as for the 10 slowest pages in a run.

The default percentiles in this report, 85, 90, and 95, are sufficient for most purposes. However, if you are required to report on a different percentile set, click **Window > Preferences > Test > Percentile Analysis Targets** to change the percentiles in this report and in the Transaction Percentile report.

The Summary page shows a graph with three bars, which represent the 85th percentile, 90th percentile, and 95th percentile response times for all users and for all pages in the run. For the 85th percentile bar, 85% of all users achieved the indicated response time or better. For the 90th percentile bar, 90% of all users achieved the indicated response time or better. And for the 95% percentile bar, 95% of all users achieved the indicated response time or better.

The 85%, 90%, and 95% pages show the response-time percentiles of the 10 slowest pages in the run. For example, if you click the tab for the 85th percentile, and the total for a page is 110 (the total is beneath each bar), you know that 85 percent of the response times for that page are less than or equal to 110 milliseconds (ms).

This graph provides an overall idea of the response times for each page. For example, the Page Performance report might indicate that a Login page is one of the 10 slowest pages. However, it is possible that one page attempt was extremely slow, but the other attempts were within range. The Page Percentile report shows which pages have slow responses because they were slow in general, not because a few responses (out of many) were extremely slow.

The table beneath the graph provides more detailed information for each page:

- The minimum response time for the run.
- The average response time for the run.
- The standard deviation of the average response time. The standard deviation tells you how tightly the data is grouped about the mean. For example, *System A* and *System B* both have an average response time of 12 ms. However, this does not mean that the response times are similar. *System A* might have response times of 11, 12, 13, and 12 ms. *System B* might have response times of 1, 20, 25, and 2. Although the mean time is the same, the standard deviation of *System B* is greater and the response time is more varied.
- The maximum response time for the run.
- The 85th percentile for the run. That is, for this page, 85% of the response times were equal to or faster than this time.

- The 90th percentile for the run. That is, for this page, 90% of the response time is equal to or faster than this time.
- The 95th percentile for the run. That is, for this page, 95% of the response time is equal to or faster than this time.
- The number of attempts in the run.

Verification Points report

This report shows the status of the verification points in your tests.

This report is displayed if your tests verify page titles, the return code for a page element, or the response size of a page element. To have your tests verify these items, complete the following steps:

1. Before recording, set the verification points. Click **Window > Preferences > Test > Test Generation > HTTP Test Generation** and under **Automatically include verification point of** select one of the verification point behavior that you want to test. Or, during test editing, enable page verification points individually by right-clicking the test item.
2. In the schedule, set the level for logging statistics to **Pages** or **All**.

The VP Pass Rate vs. Time graph on the Summary page lists the percentage of verification points that passed per sample interval during the last recorded interval. You set the **Statistics sample interval** value in the schedule, as a schedule property. When a schedule includes staged loads, colored time-range markers at the top of the graph delineate the stages.

The Page Verification Points page lists the following information for each page that has verification points:

- The number of verification points that passed in the run
- The number of verification points that failed in the run
- The percentage of verification points that passed

The Page Element Verification Points page lists the following information for each page element that has verification points:

- The name of the page and the element
- The number of verification points that passed in the run
- The number of verification points that failed in the run
- The percentage of verification points that passed

SAP performance test reports

When you test an SAP application, these performance test reports are produced during a run and saved after a run.

SAP Performance report

The SAP Performance report summarizes the health of the run, displays the data most significant to the run, shows the response trend of the slowest 10 transactions in the test, and graphs the response trend of each transaction for a specified interval.

Summary page

The *Summary* page summarizes the most important data about the test run, so that you can analyze the final or intermediate results of a test at a glance.

The *SAP Summary* section displays the following information:

- A progress indicator that shows the state of the run.
- The number of virtual users that are active and the number of virtual users that have completed testing. This number is updated during the run.
- The elapsed time. This figure is the run duration, which is displayed in hours, minutes, and seconds.
- The location and name of the test suite or schedule.
- Results for computer, All Hosts. To see summary results for individual computers, click the computer name in the Performance Test Runs view.
- The status of the run. This can be Initializing Computers, Adding Users, Running, Performing Execution History Data Transfer, Stopped, or Complete.
- The total number of virtual users that are simulated during the test.

The *Transaction Summary* section displays the following information:

- The minimum, maximum, average, and standard deviation execution time for all transactions. *Execution time* is the sum of response times for all screens of a transaction (including the connect time and inter-request delays).
- The total number of transactions that were completed and started.

The *Screen Summary* section displays the following information:

- The minimum, maximum, average, and standard deviation screen request response times for all SAP screens. *Response time* is the time between the first request character that is sent and the last response character that is received.
- The total number of SAP screens that were completed and started.
- Percent of verification points that passed.
- Total verification points that failed.
- Total verification point errors.
- Total verification points that passed.
- Total verification points that are inconclusive.

The *Element Summary* section displays the following information:

- The total number of SAP set, SAP get, and SAP call elements that were attempted.
- The total number of SAP set, SAP get, and SAP call elements that were completed.
- The percentage of completion of all SAP set, SAP get, and SAP call elements.

Screen Performance page

The *Screen Performance* page shows the average response time of the longest SAP screen requests in the test as the test progresses. With this information, you can evaluate system response during and after the test.

If you select **Do not measure performance on this screen** on a SAP screen in the test editor, then the report does not include the response time results for that screen.

The bar chart shows the average response time of the 10 slowest transactions. Each bar represents a page that you visited during recording. As you run the test, the bar chart changes, because the 10 slowest screens are updated dynamically during the run. For example, the SAP Easy Access screen might be one of the 10 slowest pages at the start of the run, but then, as the test progresses, Display Material (Initial Screen) might replace it as one of the 10 slowest. After the run, the page shows the 10 slowest transactions for the entire run.

The *Performance Summary* table that follows the bar chart provides the following additional information for each SAP screen:

- The minimum response time for each SAP screen in the run. *Response time* is the time between the first request character that is sent and the last response character that is received.
- The average response time for each SAP screen in the run. This matches the information in the chart.
- The maximum response time for each SAP screen in the run.
- The standard deviation response time for each SAP screen in the run.
- The rate per second at which each SAP screens is started.
- The total number of times each SAP screen is started.

Response vs. Time Summary page

The *Response vs. Time Summary* page shows the average response trend as graphed for a specified interval. The line graph shows the average response time for all SAP screens during the run. Each point on the graph is an average of what has occurred during that interval. You set the **Statistics sample interval** in the schedule, as a schedule property.

The table that follows the graph displays the average and standard deviation response time for all SAP screens during the run.

Response vs. Time Detail page

The *Response vs. Time Detail* page shows the response trend by screen as graphed for a specified interval. Each separate line represents a SAP screen. You set the **Statistics sample interval** in the schedule, as a schedule property.

The line graph shows the average response of each page during a specified interval. The table that follows the graph provides the following additional information for each SAP screen:

- The minimum SAP screen request response time for the run. This is the time between the moment the input is validated in the SAP GUI and the time the resulting SAP screen is displayed.
- The average SAP screen request response time for the run. This is similar to the graph, but the information in the table is for the duration of the entire run.
- The maximum SAP screen request response time for the run.
- The standard deviation for SAP screen request response time for the run.
- The rate per second at which each SAP screen is started for the most recent interval.
- The total number of times each SAP screen is started for the most recent interval.

Screen Throughput page

The *Screen Throughput* page provides an overview of the frequency of requests that are being transferred per interval.

- The line graph on the left shows two lines that represent the rate for intervals at which SAP screens are started and the rate at which SAP screens are completed. The summary table under the graph lists the start rates and the completed rates and counts for each SAP screen in the run.
- The line graph on the right shows active users and users that have completed testing, over the course of a run. The summary table under the graph lists the results for the most recent sample interval. You set the **Statistics sample interval** in the schedule, as a schedule property.

Server Health page

The *Server Health Summary* page shows an overall indication of how well the server is responding to the load.

The bar chart on the left represents the total number of SAP screens started. The bar chart on the right represents the percentage of SAP screens completed compared to the SAP screens started on each interval. The percentage can be over 100% if more screens were completed than started on a given interval.

The table that follows the charts lists the same information for each SAP screen.

Batch Input Transaction Rate page

The *Batch Input Transaction Rate* page displays the number of batch input tests that were run during the test run. The **Batch Input Transaction** graph represents the volume of batch input transactions that were processed during the test run.

SAP Verification Points report

The SAP Verification Points report shows the status of the verification points in your tests.

Summary page

The Summary page displays a line graph representing the percentage of verification points that passed per interval. You set the **Statistics sample interval** value in the schedule, as a schedule property.

Below the graph, the Verification Point Summary table lists the following information:

- The percentage of verification points that passed during the run.
- The number of verification points that were tested.
- The number of verification points that passed.
- The number of verification points that failed.
- The number of verification points that produced an error.
- The number of verification points that were inconclusive.

Screen Verification Points page

The Screen Verification Points page lists the following information for each page element that has verification points:

- The number of verification points that passed during the run
- The number of verification points that failed during the run
- The percentage of verification points that passed during the run

Verification points set on the screen title are counted with verification points set on get events in the screen.

Citrix performance test reports

When you test a Citrix XenApp application, these reports are produced during a run and saved after a run.

Citrix Performance report

The Citrix performance report summarizes the validity of the run, graphs show response times, average response time, and the server health depending on requests.

Citrix Overall page

The Overall page provides the following information:

- A progress indicator that shows the state of the run.
- The bar chart indicates the overall success of the run with the percentage of window and image synchronization successes and the percentage of verification point successes. *Synchronization success* indicates that the expected window and image events in the test match the actual window and image events in the test run.

Performance Summary page

The Summary page summarizes general data about the test run, so that you can analyze the final or intermediate results of a test at a glance.

The Run Summary table displays the following information:

- The number of virtual users that are active and the number of virtual users that have completed testing. This number is updated during the run.
- The elapsed time. This is the total duration of the run, which is displayed in hours, minutes, and seconds.
- The location and name of the test.

- The status of the run. This can be Initializing Computer(s), Adding Users, Running, Performing Execution History Data Transfer, Stopped, or Complete.
- The total number of virtual users emulated during the test.

The Citrix Summary section displays the following information:

- The statistics values (average, standard deviation, maximum, minimum) of the average response time for all response time measurements. *Response times* are determined by measurements that are located in the tests. Response time measurements can be automatically generated between the last input action before a window create event and the window create event. The table does not display values that equal zero.
- Total user actions for run. This indicates the total number of user input actions that were emulated during the run.
- The total number of window synchronization attempts.
- The total number of window synchronization successes.
- The total number of window synchronization timeouts. A timeout occurs when the synchronization fails.
- The total number of image synchronization attempts.
- The total number of image synchronization successes.
- The total number of image synchronization timeouts. A timeout occurs when the synchronization fails.

Server Performance Summary page

The server Performance Summary page shows the average time virtual users are connected, active, and disconnected.

The bar chart shows the average response time of each test phase. Each bar represents a particular phase in the test: connecting to the Web interface, connecting to the server, identifying, activity phase and disconnecting. As you run the test, the bar chart changes, because the time measurements are updated dynamically during the run.

The table under the bar chart provides the following additional information for each window:

- The minimum response time during the run.
- The average response time during the run. This matches the information in the chart.
- The maximum response time during the run.
- The standard deviation response time during the run.

Response vs. Time Summary page

The Response vs. Time Summary page shows the average response trend as graphed for a specified interval. You set the **Statistics sample interval** value in the schedule, as a schedule property. *Response times* are determined by measurements that are located in the tests. Response time measurements can be automatically generated between the last input action before a window create event and the window create event.

The line graph shows the average response time for all measurements during the run. Each point on the graph is an average of what has occurred during that interval. The table under the graph lists the statistics values of the total average response time for all measurements in the run.

Response vs. Time Details page

The Response vs. Time Details page shows the response trend as graphed for a specified interval. You set the **Statistics sample interval** value in the schedule, as a schedule property. *Response times* are determined by measurements that are located in the tests. Response time measurements can be automatically generated between the last input action before a window create event and the window create event.

The line graph shows the average response time of each measurement for a specified interval. Each measurement is represented by a separate line.

The table under the graph provides the following additional information for each response time measurement:

- The minimum response time during the run.
- The average window response time during the run. This is similar to the graph, but the information in the table includes the entire run.
- The maximum window response time during the run.
- The standard deviation window response time during the run.

User Action Throughput page

The User Action Throughput page provides an overview of the frequency of requests being transferred per interval. You set the **Statistics sample interval** value in the schedule, as a schedule property.

- The line graph on the left shows the user action rate per interval for all windows. This represents the activity of virtual user input actions per second for each interval. The table under the graph lists the user action rate per second for the entire run, and the total number of user actions for the run.
- The line graph on the right shows active users and users that have completed testing, over the course of a run. The summary table under the graph lists the results for the most recent sample interval.

Server Health Summary page

The Server Health Summary page provides an overall indication of how well the server has performed. The graph does not display values that equal zero.

The bar chart shows the following information:

- The total number of window synchronization attempts.
- The total number of window synchronization successes.
- The total number of window synchronization timeouts.
- The total number of image synchronization attempts.
- The total number of image synchronization successes.
- The total number of image synchronization timeouts.
- The total number of Citrix server errors or errors encountered during test execution.

Server Timeout page

The Server Timeout page shows when the synchronization timeouts and server errors occurred during the run. The graph does not display values that equal zero.

The line graph shows the following information:

- Citrix window synchronization timeouts.
- Citrix image synchronization timeouts.
- Citrix server errors or errors encountered during test execution.

Resources page

The Resources page shows all resource counters monitored during the schedule run.

- The line chart shows the values of the resources counters monitored during the schedule run.
The chart scales automatically to accommodate the highest resource counter value.
- The summary table under the chart lists the average values of the resource counters monitored during the schedule run. This table is organized by resource monitoring hosts.

Citrix Verification Points report

The Citrix Verification Points report shows the status of the verification points in your tests.

Summary page

The Summary page displays a line graph representing the percentage of verification points that passed per interval. You set the **Statistics sample interval** value in the schedule, as a schedule property.

Below the graph, the Window Verification Point Summary table lists the following information:

- The percentage of window verification points that passed during the run
- The number of window verification points that were attempted
- The number of window verification points that passed
- The number of window verification points that failed

The Image Synchronization Verification Point Summary table lists the following information:

- The percentage of image synchronization verification points that passed during the run
- The number of image synchronization verification points that were tested
- The number of image synchronization verification points that passed
- The number of image synchronization verification points that failed

Citrix Verification Points page

The Citrix Verification Points page contains tables with verification point details.

The Window Verification Points table lists the following information:

- The number of window verification points that passed during the run
- The number of window verification points that failed during the run
- The number of window verification points that caused an error during the run
- The number of window verification points that were inconclusive during the run
- The percentage of window verification points that passed during the run

The Image Synchronization Verification Points table lists the following information:

- The number of image synchronization verification points that passed during the run
- The number of image synchronization verification points that failed during the run
- The number of image synchronization verification points that caused an error during the run
- The number of image synchronization verification points that were inconclusive during the run
- The percentage of image synchronization verification points that passed during the run

Citrix response time percentile report

This report shows the 85th, 90th, and 95th percentile response times for all users in a run.

The default percentiles in this report, 85, 90, and 95, are sufficient for most purposes. However, if you are required to report on a different percentile set, click **Window > Preferences > Test > Percentile Analysis Targets** to change the percentiles in this report.

The Summary page shows a graph with three bars, which represent the 85th percentile, 90th percentile, and 95th percentile response times for all users in the run. For the 85th percentile bar, 85% of all users achieved the indicated response time or better. For the 90th percentile bar, 90% of all users achieved the indicated response time or better. And for the 95th percentile bar, 95% of all users achieved the indicated response time or better.

The 85%, 90%, and 95% pages show the response-time percentiles of the 10 slowest window events in the run. For example, if you click the tab for the 85th percentile, and the total for a page is 110 (the total is beneath each bar), you know that 85 percent of the response times for that window event are less than or equal to 110 milliseconds (ms).

This graph provides an overall idea of the response times for each window event. For example, the Response Time Measurements page of the Citrix performance report might indicate that a Login screen is one of the 10 slowest measurements. However, it is possible that one attempt was extremely slow, but the other attempts were within range. The Citrix response time percentile report shows which measurements have slow responses because they were slow in general, not because a few responses (out of many) were extremely slow.

The table beneath the graph provides more detailed information for each window event:

- The minimum response time for the run.
- The average response time for the run.
- The standard deviation of the average response time. The standard deviation tells you how tightly the data is grouped about the mean. For example, *System A* and *System B* both have an average response time of 12

ms. However, this does not mean that the response times are similar. *System A* might have response times of 11, 12, 13, and 12 ms. *System B* might have response times of 1, 20, 25, and 2. Although the mean time is the same, the standard deviation of *System B* is greater and the response time is more varied.

- The maximum response time for the run.
- The 85th percentile for the run. That is, for this particular measurement, 85% of the response times were equal to or faster than this time.
- The 90th percentile for the run. That is, for this particular measurement, 90% of the response times were equal to or faster than this time.
- The 95th percentile for the run. That is, for this particular measurement, 95% of the response times were equal to or faster than this time.
- The number of attempts in the run.

Related reference

Percentile analysis preferences

Web service reports

When you test a web service, these reports are produced during a run and saved after a run.

Service Performance report

The Service Performance report summarizes the validity of the run, summarizes the data most significant to the run, shows the response trend of the slowest 10 service calls in the test, the server health depending on requests, and graphs the response trend of each service calls for a specified interval.

Overall page

The Overall page provides the following information:

- A progress indicator that shows the state of the run.
- The bar graph on the left indicates the percentage of successful service calls during the run.
- The bar graph on the right indicates the percentage of verification points with a Pass status for the run.

Summary page

The Summary page summarizes the most important data about the test run, so that you can analyze the final or intermediate results of a test at a glance.

The Run Summary table displays the following information:

- The number of virtual users that are active and the number of virtual users that have completed testing. These numbers are updated during the run.
- The elapsed time. This is the total duration of the run, which is displayed in hours, minutes, and seconds.
- The location and name of the test.

- The results for the computer and for all computers. To see summary results for individual computers, click the computer name in the **Performance Test Runs** view.
- The status of the run. This can be Initializing Computers, Adding Users, Running, Performing Execution History Data Transfer, Stopped, or Complete.
- The total number of virtual users emulated during the test.

The Call Summary section displays the following information:

- The percentage of verification points with a Pass status.
- The total number of verification points with a Fail status.
- The total number of verification points with an Error status.
- The total number of attempted service calls.
- The total number of successful service calls.
- The total number of service calls that produced a timeout.

The Bytes Summary section displays the following information:

- The minimum, maximum, and average number of bytes sent and received for each call in the run.
- The byte rate per second for the run.
- The total number of bytes sent and received for the run.

Response Time Results page

The Response Time Results page shows the average response of the service calls in the test as the test progresses. With this information, you can evaluate system response during and after the test. The delay between the moment a service call is invoked and the moment the corresponding message return is received, determines the *Response times*.

The bar chart shows the average response time of each service call. Each bar represents a service call that was invoked during the test. As you run the test, the bar chart changes, because the response times are updated dynamically during the run.

The table that follows the bar chart provides the following additional information for each service call:

- The minimum response time during the run.
- The average response time during the run. This matches the information in the chart.
- The maximum response time during the run.
- The standard deviation response time during the run.

Response Time vs. Time Summary page

The Response vs. Time Summary page shows the average response trend as graphed for a specified interval. You set the **Statistics sample interval** value in the schedule, as a schedule property. Measurements that are located in the tests determine the *Response times*. Response time measurements can be automatically generated between the last input action before a service call and the corresponding message return event.

The line graph shows the average response time for all measurements during the run. Each point on the graph is an average of what has occurred during that interval. The table that follows the graph lists one number: the total average response time for all measurements in the run.

Response Time vs. Time Details page

The Response vs. Time Details page shows the response trend as graphed for a specified interval. You set the **Statistics sample interval** value in the schedule, as a schedule property. The delay between the moment a service call is invoked and the moment the corresponding message return is received determines the *Response times*.

The line graph shows the average response time of each measurement for a specified interval. A separate line represents each measurement.

The table under the graph provides the following additional information for each response time measurement:

- The minimum response time during the run.
- The average service call response time during the run. This is similar to the graph, but the information in the table includes the entire run.
- The maximum service call response time during the run.
- The standard deviation service call response time during the run.

Data Volume page

The Data Volume page provides details about the volume of data that is sent to and received from the service. You set the **Statistics sample interval** value in the schedule, as a schedule property.

- The **Sent and Received** line graph shows the total bytes sent and received per interval.
- The **Received Summary** table lists, for each call, the received volume rate (bytes per second) for the entire run, the minimum and maximum received bytes per interval, and the average number of bytes received for each call.
- The **Sent Summary** table lists, for each call, the sent volume rate (bytes per second) for the entire run, the minimum and maximum sent bytes per interval, and the average number of bytes sent for each call.


Call Throughput page

The Call Throughput page provides an overview of the frequency of service calls that are being transferred per interval. You set the **Statistics sample interval** value in the schedule, as a schedule property.

- The line graph shows the calls that are started and ended per interval. Ended calls can be: *success*, *fail*, or *timeout*.
- The **Performance Summary** table lists the details of the number of call starts, successes, failures or timeouts for each call and for the run.

Resources page

The **Resources** page displays information about all the resource counters that are monitored during the *schedule* run. You can view the following information as mentioned in the table from the **Resources** page:


If...	Then the Resources page displays...
If you did not add any Resource Monitoring source to a performance schedule	A message that states that you must set up the Resource Monitoring sources to view the resource details.
If you added Resource Monitoring sources to a performance schedule	<ul style="list-style-type: none"> • The Resource Monitoring sources that are monitored during the schedule run. • All resource counters for those Resource Monitoring sources that are monitored during the schedule run. • The Unavailable sources section lists the Resource Monitoring sources that are unavailable or unreachable during the schedule run. <p> Note: The Unavailable sources section is displayed only if any of the Resource Monitoring sources are unreachable or unavailable during the schedule run.</p>
If you added Resource Monitoring sources by using labels to a performance schedule	<ul style="list-style-type: none"> • The following information in the Server sources matching the labels set in the schedule (*Source defined in team space) section: <ul style="list-style-type: none"> ◦ Labels and the Resource Monitoring sources associated with those labels that are monitored during the schedule run. ◦ Resource Monitoring sources that are unavailable or unreachable during the schedule run. ◦ An empty array (<code>[]</code>) when you used labels that are not tagged to any Resource Monitoring source in Rational® Test Automation Server. ◦ The asterisk (*) symbol is displayed after the name of the Resource Monitoring

If...	Then the Resources page displays...
	<p>source if you add the Resource Monitoring source at the team space level in Rational® Test Automation Server.</p> <ul style="list-style-type: none"> • All resource counters for the Resource Monitoring sources that are monitored during the schedule run.
<p>If you ran a performance schedule by using the <code>overrideLabels</code> command from the Rational® Performance Tester command line</p>	<ul style="list-style-type: none"> • The following information in the Server sources matching the labels set with the command-line flag <code>-overrideLabels</code> (*Source defined in team space) section: <ul style="list-style-type: none"> ◦ Labels that you use to add the Resource Monitoring sources to the schedule for the schedule run. ◦ Resource Monitoring sources associated with those labels that are monitored during the schedule run. ◦ Resource Monitoring sources that are unavailable or unreachable during the schedule run. ◦ An empty array (<code>[]</code>) when you use labels that are not tagged to any Resource Monitoring source in Rational® Test Automation Server. ◦ The asterisk (*) symbol is displayed after the name of the Resource Monitoring source if you add the Resource Monitoring source at the team space level in Rational® Test Automation Server. • All resource counters for the Resource Monitoring sources that are monitored during the schedule run.

The Legend displays the Resource Monitoring type and its resource counters. When there are multiple Resource Monitoring sources, the resource counters for the respective sources are displayed in front of their Resource Monitoring source name. You can customize the resource counter information displayed in a graph by clicking any

individual resource counter or type of source. You can click or double-click any individual resource counter for the following results:

- A single click on the resource counter hides the data displayed on the graph. Click the resource counter again to display the data in the graph.
- A double-click on the resource counter removes information about all other resource counters from the graph and displays only the information about the selected resource counter.

 **Tip:** You can click **Select All** option to restore all the resource counter information on the graph.

When you click on any of the sources, the graph removes all the resource counters of other sources and displays only the resource counters of the selected source.

For example, you have an Apache httpd server and a Windows Performance Host as a Resource Monitoring source. When the *schedule* completes, the **Resources** page displays the resource counter information of both the sources. If you want to analyze the resource counters for any one of the sources, you can click the Apache httpd server or the Windows Performance Host. Based on your selection, the graph is updated to display the selected source resource counters information.

The Performance Summary table that follows the graph, lists the most recent values of the resource counters that are monitored during the schedule run. The first two columns display the **Type** of the source and **Name** of the resource counter. This table also lists the minimum, maximum, and average values of the resource counters that are monitored during the schedule run.

Web Service Verification Points report

The web service verification points report shows the status of the verification points in your tests.

Summary page

The **Summary** page displays a bar graph representing the percentage of successful web service calls for the test run. You can set the **Statistics sample interval** value in the schedule, as a schedule property.

The *Verification Point Summary Tab* table lists the following information:

- The number of verification points that were attempted, passed, and failed during the test run.
- The percentage of verification points that passed per interval during the test run.

Verification Points Detail page

The **Verification Points Detail** page displays the details for all types of verification points that were checked during the test run.

The *Verification Points Detail* table lists the following information:

- The number of verification points that passed, failed, caused an error, were inconclusive during the test run.
- The percentage of verification points that passed during the test run.

Response Contain Verification Points page

The **Response Contain Verification Points** page displays the details of contain verification points that were checked during the test run.

The *Response Contain Verification Points* table lists the following information:

- The number of verification points that passed, failed, caused an error, were inconclusive during the test run.
- The percentage of verification points that passed during the test run.

Response Equal Verification Points page

The **Response Equal Verification Points** page displays the details of equal verification points that were checked during the test run.

The *Response Equal Verification Points* table lists the following information:

- The number of verification points that passed, failed, caused an error, were inconclusive during the test run.
- The percentage of verification points that passed during the test run.

Response Properties Verification Points page

The **Response Properties Verification Points** page displays the details of the verification point for the properties that were checked during the test run.

The *Response Properties Verification Points* table lists the following information:

- The number of verification points that passed, failed, caused an error, and were inconclusive during the test run.
- The percentage of verification points that passed during the test run.

Response Query Verification Points page

The **Response Query Verification Points** page displays the details of query verification points that were checked during the test run.

The *Response Query Verification Points* table lists the following information:

- The number of verification points that passed, failed, caused an error, were inconclusive during the test run.
- The percentage of verification points that passed during the test run.

Response XSD Verification Points

The **Response XSD Verification Points** page displays the details of verification points that were checked during the test run.

The *Response XSD Verification Points* table lists the following information:

- The number of verification points passed, failed, caused an error, were inconclusive during the test run.
- The percentage of verification points that passed during the test run.

Response Attachment Verification Points page

The **Response Attachment Verification Points** page displays the details of attachment verification points that were checked during the test run.

The *Response Attachment Verification Points* table lists the following information:

- The number of verification points that passed, failed, caused an error, were inconclusive during the test run.
- The percentage of verification points that passed during the test run.

Response Text Verification Points

The **Response Text Verification Points** page displays the details of verification points that were checked during the test run.

The *Response Text Verification Points* table lists the following information:

- The number of verification points that passed, failed, caused an error during the test run.
- The percentage of verification points that passed during the test run.

Callback Verification Points

The **Callback Verification Points** page displays the details of verification points that were checked during the test run.

The *Callback Verification Points* table lists the following information:

- The number of verification points that passed, failed, caused an error, were inconclusive during the test run.
- The percentage of verification points that passed during the test run.

WSDL Coverage report

The Web Services Description Language (WSDL) Coverage report displays all the bindings, methods, and ports of a web service that were used in the test, compound test, or schedule. In this report, you can see the methods or ports that are not called by the test so that they require more tests for better coverage.

To generate a WSDL Coverage report, in the Test Navigator view, right-click a service test result, and click **WSDL Coverage Report**.

Global Summary

This section displays the number of tests, verification points, and defects available in the report.













WSDL Coverage Summary

This section includes sub-sections for the name of all the compound tests, schedules, and test runs for each WSDL service. For example, if you run a compound test that contains four tests and each test runs a different WSDL service, this section displays a table with four rows for the same compound test but different WSDL services used. Another table displays each test run with the corresponding WSDL service. The tables would display a column for the overall coverage percentage of each WSDL service.

WSDL Coverage Details

This section also includes sub-sections for the names of all the compound tests, schedules, and test runs for each WSDL service. However, this section provides detail coverage information at the bindings, methods, and ports level for the WSDL service. It reports the verdict and the number of requests sent and responses received. The Coverage column of the table indicates whether the methods or ports are called by the test. If the method is called, it shows as 100%. If the method is not called, it shows as 0%. The percentages are then rolled up to the bindings and WSDL service level.

For example, in the following screenshot, there is one binding `BasicHttpBinding_IStockQuoteService` with two methods. The method `GetStockQuote` is called by the test. But, the `GetWorldMajorIndices` method was not called. Therefore, at the binding level, you see 50% coverage. As a user of the web service, if the `GetWorldMajorIndices` method is required, you can check why it was not used by the test.

WSDL	Coverage
▲  StockQuoteService	50 % 
▲  BasicHttpBinding_IStockQuoteService	50 % 
▲  GetStockQuote	100 % 
 BasicHttpBinding_IStockQuoteService	100 % 
▲  GetWorldMajorIndices	0 % 
 BasicHttpBinding_IStockQuoteService	0 % 

Socket performance test reports

When you test a socket application, these performance test reports are produced during a run and saved after a run.

Socket Performance report

The Socket Performance report summarizes the health of the run, displays the data most significant to the run, shows the response trend of the slowest 10 transactions in the test, and graphs the response trend of each transaction for a specified interval, for socket API performance tests.

Overall page

The Overall page summarizes the most important data about the test run, so that you can analyze the final or intermediate results of a test at a glance.

The top of the page displays a progress indicator bar that shows the state of the run.

The Virtual Users Activity bar chart displays the number of virtual users that are active and the number of virtual users that have completed testing. This number is updated during the run.

The Run Summary section displays the following information:

- The name of the test or schedule.
- The number of virtual users that are active and the number of virtual users that have completed testing. This number is updated during the run.
- The total number of virtual users that are simulated during the test.
- The elapsed time. This figure is the run duration, which is displayed in hours, minutes, and seconds.
- The status of the run. This can be Initializing Computers, Adding Users, Running, Performing Execution History Data Transfer, Stopped, or Complete.
- Results for the computer, All Hosts. To see summary results for individual computers, click the computer name in the Performance Test Runs view.

Connect Performance page

The Connect Performance page shows the performance of socket connection actions. With this information, you can evaluate system response during and after the test.

The Connect Actions vs. Time bar chart shows the attempted and successful connection actions during the test run.

The Connect Times vs. Time bar chart shows average, maximum, minimum, and standard deviation of connection times during the test run.

The Connect Actions Summary displays this information:

- The number of connections that were attempted
- The number of connections that were successful

The Connect Times Summary displays the average, maximum, minimum, and standard deviation of connection times.

Send/Receive Performance page

The Send/Receive Performance page shows the data throughput trend as graphed for a specified interval. The line graph shows the average response time for all socket send and receive actions during the run. Each point on the graph is an average of what has occurred during that interval. You set the **Statistics sample interval** in the schedule as a schedule property.

The Send/Receive Actions vs. Time bar chart shows the attempted and successful send and receive actions during the test run.

The Response Times vs. Time bar chart shows average, maximum, minimum, and standard deviation of response times during the test run.

The Connect Actions Summary displays this information:

- The number of send and receive actions that were attempted
- The number of send and receive actions that were successful

The Connect Times Summary displays the average, maximum, minimum, and standard deviation of response times.

Byte Performance page

The Byte Performance page provides an overview of the frequency of requests that are being transferred per interval:


- The Exchanged Bytes vs. Time line graph shows the quantity of bytes sent and received during the test run.
- The Exchanged Bytes summary table after the graph lists the total number of bytes sent and received.

HTTP counters

HTTP counters, which are displayed in the Performance Test Runs view, enable you to customize your reports with dynamic information that is updated for each run.

Byte counters

These counters provide throughput information regarding the rate and the number of bytes sent and received during a sample interval and during a run.

The counters in the following tables provide an aggregate value for all tests. Folders that contain aggregate counters have a clock superimposed on them: .

Counter name	Description
Bytes Received [for Run]	The total number of bytes received for all tests for the entire run
Bytes Sent [for Run]	The total number of bytes sent for all tests for the entire run

Counter name	Description
Count [for Interval]	The total number of bytes received for all tests within the most recent sample interval
Rate [per second] [for Interval]	The bytes per second received for all tests within the most recent sample interval


Counter name	Description
Rate [per second] [for Run]	The bytes per second received for all tests for the entire run

Counter name	Description
Count [for Interval]	The total number of bytes sent for all tests within the most recent sample interval
Rate [per second] [for Interval]	The bytes per second sent for all tests within the most recent sample interval
Rate [per second] [for Run]	The bytes per second sent for all tests for the entire run


Page counters

These counters provide information about page attempts, page hits, response time, response success, and verification points. Some counters produce an aggregate value, where the values for several pages are rolled up into one value; others produce values for each page.

Aggregate counters use the values for all the pages in a test to produce a single value for a report. This value is rolled up from all values that satisfy the counter. When you drag an aggregate counter onto a report, one value is displayed.

Folders that contain aggregate counters have a clock superimposed on the folder icon: . The tables below that list aggregate counters have this icon in their title.

Individual counters produce values for each item that satisfies the counter, rather than a single rolled-up value.

Folders that contain individual counters have an asterisk superimposed on the folder icon: . The tables below that list individual counters have this icon in their title.

Some counters pertain to *intervals* in the run. You set the **Statistics sample interval** value in the schedule, as a schedule property.

Adjustment counters

The counters in this section provide information about adjustments made to page response times.

Counter name	Description
Average [for Interval]	The average of all adjustments applied during the most recent sample interval
Average [for Run]	The average of all adjustments applied for the entire run
Maximum [for Interval]	The maximum adjustment applied during the most recent sample interval
Maximum [for Run]	The maximum adjustment applied for the entire run
Minimum [for Interval]	The minimum adjustment applied during the most recent sample interval
Minimum [for Run]	The minimum adjustment applied for the entire run

Counter name	Description
Standard Deviation [for Interval]	The standard deviation for adjustments applied within the most recent sample interval
Standard Deviation [for Run]	The standard deviation for adjustments applied for the entire run

The counters in the following table are available only after a run, and only after you have displayed the Page Percentile report.

Counter name	Description
85	85% of the users had a maximum adjustment of this amount applied
90	90% of the users had a maximum adjustment of this amount applied
95	95% of the users had a maximum adjustment of this amount applied

Attempt counters

The counters in this section provide information about *attempts*. When an attempt refers to a page, it means that the primary request was sent; it does not include requests within the page. When an attempt refers to a page element, it means that a request was sent.

Counter name	Description
Count [for Interval]	The number of attempts for a specific page within the most recent sample interval
Count [for Run]	The number of attempts for a specific page for the entire run
Rate [per second] [for Interval]	The rate at which attempts occurred for a specific page during the most recent sample interval
Rate [per second] [for Run]	The rate at which attempts occurred for a specific page for the entire run

Counter name	Description
Count [for Interval]	The total number of page attempts completed for a specific page during the most recent sample interval

Counter name	Description
Count [for Interval]	The total number of page attempts completed for a specific page element for the most recent sample interval

Counter name	Description
Count [for Interval]	The number of attempts for a specific page element within the most recent sample interval
Count [for Run]	The number of attempts for a specific page element for the entire run
Rate [per second] [for Interval]	The rate at which attempts occurred for a specific page element during the most recent sample interval
Rate [per second] [for Run]	The rate at which attempts occurred for a specific page element for the entire run

Counter name	Description
Page Attempt Rate [per second] [for Interval]	The rate at which page attempts occurred within the most recent sample interval
Page Attempt Rate [per second] [for Run]	The rate at which page attempts occurred for the entire run
Page Element Attempt Rate [per second] [for Interval]	The rate at which page element attempts occurred within the most recent sample interval
Page Element Attempt Rate [per second] [for Run]	The rate at which page element attempts occurred for the entire run
Total Page Attempts [for Interval]	The number of page attempts during the most recent sample interval
Total Page Attempts [for Run]	The number of page attempts for the entire run
Total Page Element Attempts [for Interval]	The number of page element attempts within the most recent sample interval
Total Page Element Attempts [for Run]	The number of page element attempts for the entire run

Example

Each counter in the following table provides a single value to the report. This aggregated value has been calculated from all pages.

Hit counters

Counter name	Description
Count [for Interval]	The number of hits for a specific page within the most recent sample interval
Count [for Run]	The number of hits for a specific page for the entire run

Counter name	Description
Rate [per second] [for Interval]	The rate at which hits occurred for a specific page during the most recent sample interval
Rate [per second] [for Run]	The rate at which hits occurred for a specific page for the entire run

Counter name	Description
Count [for Interval]	The number of hits for a specific page element within the most recent sample interval
Count [for Run]	The number of hits for a specific page element for the entire run
Rate [per second] [for Interval]	The rate at which hits occurred for a specific page element during the most recent sample interval
Rate [per second] [for Run]	The rate at which hits occurred for a specific page element for the entire run

Counter name	Description
Page Element Hit Rate [per second] [for Interval]	The rate of page element hits during the most recent sample interval
Page Element Hit Rate [per second] [for Run]	The rate of page element hits for the entire run
Page Hit Rate [per second] [for Interval]	The rate of page hits during the most recent sample interval
Page Hit Rate [per second] [for Run]	The rate of page hits for the entire run
Total Page Element Hits [for Interval]	The total number of page element hits within the most recent sample interval
Total Page Element Hits [for Run]	The total number of page element hits for the entire run
Total Page Hits [for Interval]	The total number of page hits during the most recent sample interval
Total Page Hits [for Run]	The total number of page hits for the entire run

Response Time counters

Response time counters omit page response times for pages that contain requests with status codes in the range of 4XX (client errors) to 5XX (server errors). The only exception is when the failure (for example, a 404) is recorded and returned, and the request is not the primary request for the page. Page response times that contain requests that time out are always discarded.

Counter name	Description
Average [for Interval]	The average response time for a specific page within the most recent sample interval
Average [for Run]	The average response time for a specific page for the entire run
Maximum [for Interval]	The maximum response time for a specific page within the most recent sample interval
Maximum [for Run]	The maximum response time for a specific page for the entire run
Minimum [for Interval]	The minimum response time for a specific page within the most recent sample interval
Minimum [for Run]	The minimum response time for a specific page for the entire run
Standard Deviation [for Interval]	The standard deviation for a specific page within the most recent sample interval
Standard Deviation [for Run]	The standard deviation for a specific page for the entire run

Counter name	Description
Average [for Interval]	The average response time for a specific page within the most recent sample interval
Average [for Run]	The average response time for a specific page for the entire run

The counters in the following table are available only after a run, and only after you have displayed the Page Percentile report.

Counter name	Description
85	85% of the users experienced this response time or better on this specific page
90	90% of the users experienced this response time or better on this specific page
95	95% of the users experienced this response time or better on this specific page

The counters in the following table are available only after a run, and only after you have displayed the Page Percentile report.

Counter name	Description
85	85% of the users experienced this response time or better
90	90% of the users experienced this response time or better
95	95% of the users experienced this response time or better

Response time counters omit page response times for pages that contain requests with status codes in the range of 4XX (client errors) to 5XX (server errors). The only exception is when the failure (for example, a 404) is recorded and returned, and the request is not the primary request for the page. Page response times that contain requests that time out are always discarded.

Counter name	Description
Average Response Time for All Page Elements [ms] [for Interval]	The average response time for all page elements within the most recent sample interval
Average Response Time for All Page Elements [ms] [for Run]	The average response time for all page elements for the entire run
Average Response Time for All Pages [ms] [for Interval]	The average response time for all pages within the most recent sample interval
Average Response Time for All Pages [ms] [for Run]	The average response time for all pages for the entire run
Maximum Response Time for All Pages [ms] [for Interval]	The maximum response time for all pages within the most recent sample interval
Maximum Response Time for All Pages [ms] [for Run]	The maximum response time for all pages for the entire run
Minimum Response Time for All Pages [ms] [for Interval]	The minimum response time for all pages within the most recent sample interval
Minimum Response Time for All Pages [ms] [for Run]	The minimum response time for all pages for the entire run
Response Time Standard Deviation for All Page Elements [for Interval]	The standard deviation for all page elements within the most recent sample interval
Response Time Standard Deviation for All Page Elements [for Run]	The standard deviation for all page elements for the entire run

Counter name	Description
Response Time Standard Deviation for All Pages [for Interval]	The standard deviation for all pages within the most recent sample interval
Response Time Standard Deviation for All Pages [for Run]	The standard deviation for all pages for the entire run

Status Code Success counters

A *status code success* means that the response code verification point passed for that request. If the request has no verification points, a success means that the server received a request and returned a response where the status code was in the 200 or 300 category, or returned an expected response in the 400 or 500 category.

Counter name	Description
Count [for Interval]	The number of status code successes for a specific page during the most recent sample interval
Count [for Run]	The number of status code successes for a specific page for the entire run
Percent Status Code Success [for Interval]	The percentage of status code successes for a specific page during the most recent sample interval
Percent Status Code Success [for Run]	The percentage of status code successes for a specific page for the entire run

Counter name	Description
Count [for Interval]	The number of status code successes for a specific page element during the most recent sample interval
Count [for Run]	The number of status code successes for a specific page element for the entire run
Percent Status Code Success [for Interval]	The percentage of status code successes for a specific page element during the most recent sample interval
Percent Status Code Success [for Run]	The percentage of status code successes for a specific page element for the entire run

Counter name	Description
Percent Page Element Status Code Success [for Interval]	The percentage of status code successes for all page elements during the most recent sample interval

Counter name	Description
Percent Page Element Status Code Success [for Run]	The percentage of status code successes for all page elements for the entire run
Percent Page Status Code Success [for Interval]	The percentage of status code successes for all pages during the most recent sample interval
Percent Page Status Code Success [for Run]	The percentage of status code successes for all pages for the entire run
Total Page Element Status Code Successes [for Interval]	The number of status code successes for all page elements during the most recent sample interval
Total Page Element Status Code Successes [for Run]	The number of status code successes for all page elements for the entire run
Total Page Status Code Successes [for Interval]	The number of status code successes for all pages within the most recent sample interval
Total Page Status Code Successes [for Run]	The number of status code successes for all pages for the entire run

Verification Point counters

The tables in this section provide information about verification points. The verdict for a verification point can be `PASS`, `Fail`, `Error`, or `Inconclusive`.

- **Pass** indicates that all verification points matched or received the expected response. For example, a response code verification point is set to `PASS` when the recorded response code is received during playback. If your test does not contain verification points, `PASS` means that all primary requests in the test were successful.
- **Fail** indicates that at least one verification point did not match the expected response or that the expected response was not received.
- **Error** indicates one of the following results: a primary request was not successfully sent to the server, no response was received from the server for a primary request, or the primary request response was incomplete or could not be parsed.
- The verdict is set to **Inconclusive** only if you provide custom code that defines a verdict of **Inconclusive**.

Counter name	Description
Percent Pass	The percentage of page title verification points that passed for a specific page for the entire run

Counter name	Description
Percent Pass	The percentage of response code or response size verification points that passed for a specific page element for the entire run

Counter name	Description
Count [for Interval]	The number of response code or response size verification points classified as <code>Error</code> for a specific page element during the most recent sample interval
Count [for Run]	The number of response code or response size verification points classified as <code>Error</code> for a specific page element for the entire run

Counter name	Description
Count [for Interval]	The number of response code or response size verification points that failed for a specific page element during the most recent sample interval
Count [for Run]	The number of response code or response size verification points that failed for a specific page element for the entire run

Counter name	Description
Count [for Interval]	The number of response code or response size verification points classified as <code>Inconclusive</code> for a specific page element during the most recent sample interval
Count [for Run]	The number of response code or response size verification points classified as <code>Inconclusive</code> for a specific page element for the entire run

Counter name	Description
Count [for Interval]	The number of response code or response size verification points that passed for a specific page element during the most recent sample interval
Count [for Run]	The number of response code or response size verification points that passed for a specific page element for the entire run

Counter name	Description
Count [for Interval]	The number of response code or response size verification points classified as <code>ERROR</code> for a specific page during the most recent sample interval
Count [for Run]	The number of response code or response size verification points classified as <code>ERROR</code> for a specific page for the entire run

Counter name	Description
Count [for Interval]	The number of response code or response size verification points that failed for a specific page during the most recent sample interval
Count [for Run]	The number of response code or response size verification points that failed for a specific page for the entire run

Counter name	Description
Count [for Interval]	The number of response code or response size verification points classified as <code>Inconclusive</code> for a specific page during the most recent sample interval
Count [for Run]	The number of response code or response size verification points classified as <code>Inconclusive</code> for a specific page for the entire run

Counter name	Description
Count [for Interval]	The number of response code or response size verification points that passed for a specific page during the most recent sample interval
Count [for Run]	The number of response code or response size verification points that passed for a specific page for the entire run

Counter name	Description
Percent Page Element VPs Passed [for Interval]	The percentage of response code or response size verification points that passed during the most recent sample interval

Counter name	Description
Percent Page Element VPs Passed [for Run]	The percentage of response code or response size verification points that passed for the entire run
Percent Page VPs Passed [for Interval]	The percentage of page title verification points that passed during the most recent sample interval
Percent Page VPs Passed [for Run]	The percentage of page title verification points that passed for the entire run
Total Page Element VPs Attempted [for Interval]	The number of response code or response size verification points executed during the most recent sample interval
Total Page Element VPs Attempted [for Run]	The number of response code or response size verification points executed for the entire run
Total Page Element VPs Error [for Interval]	The number of response code or response size verification points with a verdict of <code>ERROR</code> within the most recent sample interval
Total Page Element VPs Error [for Run]	The number of response code or response size verification points with a verdict of <code>ERROR</code> for the entire run
Total Page Element VPs Failed [for Interval]	The number of response code or response size verification points that failed during the most recent sample interval
Total Page Element VPs Failed [for Run]	The number of response code or response size verification points that failed for the entire run
Total Page Element VPs Inconclusive [for Interval]	The number of response code or response size verification points that were marked as <code>Inconclusive</code> within the most recent sample interval
Total Page Element VPs Inconclusive [for Run]	The number of response code or response size verification points that were marked as <code>Inconclusive</code> for the entire run
Total Page Element VPs Passed [for Interval]	The number of response code or response size verification points that passed during the most recent sample interval
Total Page Element VPs Passed [for Run]	The number of response code or response size verification points that passed for the entire run
Total Page VPs Attempted [for Interval]	The percentage of page title verification points that were executed during the most recent sample interval
Total Page VPs Attempted [for Run]	The percentage of page title verification points that were executed for the entire run
Total Page VPs Error [for Interval]	The percentage of page title verification points with a verdict of <code>ERROR</code> during the most recent sample interval
Total Page VPs Error [for Run]	For each page that contains verification points, the total with a verdict of <code>ERROR</code> for the entire run

Counter name	Description
Total Page VPs Failed [for Interval]	The number of page title verification points that failed during the most recent sample interval
Total Page VPs Failed [for Run]	The number of page title verification points that failed for the entire run
Total Page VPs Inconclusive [for Interval]	The number of page title verification points that were marked as <i>Inconclusive</i> during the most recent sample interval
Total Page VPs Inconclusive [for Run]	The number of page title verification points that were marked as <i>Inconclusive</i> for the entire run
Total Page VPs Passed [for Interval]	The number of page title verification points that passed during the most recent sample interval
Total Page VPs Passed [for Run]	The number of page title verification points that passed for the entire run

Run counters

These counters provide information about the active users in the run, the users that have completed the run, and the HTTP status codes that were received.

Some counters pertain to *intervals* in the run. You set the **Statistics sample interval** value in the schedule as a schedule property.

Run

Counter name	Description
Displaying Results for computer	The name of the host.
Run Status	The run status or errors.

> Active Users

Counter name	Description
Count [for Run]	The number of users that are currently active.

Counter Name	Description
Count [for Interval]	The length of time it took to collect all the statistics to send for the most recent sample interval. Typically a brief period. This value has limited use in a custom report.

Counter name	Description
Count [for Run]	The number of users that have completed the run.

Counter Name	Description
Count [for Inter- val]	The number of HTTP status codes between 100 and 199 that were received during the most recent sample interval.
Count [for Run]	The number of HTTP status codes between 100 and 199 that were received in the entire run.

Counter name	Description
Count [for Inter- val]	The number of HTTP status codes between 200 and 299 that were received during the most recent sample interval.
Count [for Run]	The number of HTTP status codes between 200 and 299 that were received in the entire run.

Counter name	Description
Count [for Inter- val]	The number of HTTP status codes between 300 and 399 that were received during the most recent sample interval.
Count [for Run]	The number of HTTP status codes between 300 and 399 that were received in the entire run.

Counter name	Description
Count [for Inter- val]	The number of HTTP status codes between 400 and 499 that were received during the most recent sample interval.
Count [for Run]	The number of HTTP status codes between 400 and 499 that were received in the entire run.

Counter name	Description
Count [for Inter- val]	The number of HTTP status codes between 500 and 599 that were received during the most recent sample interval.
Count [for Run]	The number of HTTP status codes between 500 and 599 that were received in the entire run.

Counter name	Description
Count [for Run]	The duration of the run, in milliseconds (ms).

Counter name	Description
Elapsed Time [H:M:S]	The duration of the run, in hours, minutes, and seconds.

Run> Sample Interval Length

Counter name	Description
Count [for Run]	The length of the sample interval.

Counter name	Description
Executed Test	The web address of the schedule or test that was run.


Counter name	Description
Count [for Run]	The clock value at the start of the run, in milliseconds (ms).

Counter Name	Description
Count [for Run]	The total number of users that were involved in the run.

Test counters


These counters provide information about the execution time of the tests in a run. Some counters produce an aggregate value, where the values for all tests are rolled up into one value; others produce individual values for each test.

Some counters pertain to *intervals* in the run. You set the **Statistics sample interval** value in the schedule, as a schedule property.

The counters in the following table provide an aggregate value for all tests. Folders that contain aggregate counters have a clock superimposed on them: .

Counter name	Description
Average [for Interval]	The average execution time for all tests within the most recent sample interval
Average [for Run]	The average execution time for all tests in the entire run

Counter name	Description
Maximum [for Interval]	The maximum execution time for all tests within the most recent sample interval
Maximum [for Run]	The maximum execution time for all tests in the entire run
Minimum [for Interval]	The minimum execution time for all tests within the most recent sample interval
Minimum [for Run]	The minimum execution time for all tests in the entire run
Standard Deviation [for Interval]	The standard deviation for all tests within the most recent sample interval
Standard Deviation [for Run]	The standard deviation for all tests in the entire run


The counters in the following table provide individual values for each test. Folders that contain individual counters have an asterisk superimposed on them: .

Counter name	Description
Average Execution Time for All Tests [ms] [for Interval]	The average execution time for each test within the most recent sample interval
Average Execution Time for All Tests [ms] [for Run]	The average execution time for each test in the entire run
Execution Time Standard Deviation for All Tests [for Interval]	The standard deviation for each test within the most recent sample interval
Execution Time Standard Deviation for All Tests [for Run]	The standard deviation for each test in the entire run
Maximum Execution Time for All Tests [ms] [for Interval]	The maximum execution time for each test within the most recent sample interval
Maximum Execution Time for All Tests [ms] [for Run]	The maximum execution time for each test in the entire run
Minimum Execution Time for All Tests [ms] [for Interval]	The minimum execution time for each test within the most recent sample interval
Minimum Execution Time for All Tests [ms] [for Run]	The minimum execution time for each test in the entire run


Transaction counters

These counters provide information about transactions that were attempted, transactions that were completed, and the elapsed time for the transactions. Some counters produce an aggregate value, where the values for all transactions are rolled up into one value; others produce individual values for each transaction.


Some counters pertain to *intervals* in the run. You set the **Statistics sample interval** value in the schedule, as a schedule property.

The counters in the following table provide an aggregate value for all transactions. Folders that contain aggregate counters have a clock superimposed on them: .


Counter name	Description
Count [for Interval]	The total number of transactions that were attempted within the last recorded interval
Count [for Run]	The total number of transactions that were attempted in the entire run
Rate [per second] [for Interval]	The number of transactions that were attempted per second in the last recorded interval
Rate [per second] [for Run]	The number of transactions that were attempted per second in the entire run

The counters in the following table provide individual values for each transaction. Folders that contain individual counters have an asterisk superimposed on them: .


Counter name	Description
Total Transactions Started [for Interval]	The number of transactions that were started within the last recorded interval
Transaction Start Rate [per second] [for Interval]	The rate that transactions that were started within the last recorded interval
Transaction Start Rate [per second] [for Run]	The rate that transactions that were started in the entire run

The counters in the following table provide an aggregate value for all transactions. Folders that contain aggregate counters have a clock superimposed on them: .


Counter name	Description
Count [for Interval]	The total number of transactions that were completed within the last recorded interval
Count [for Run]	The total number of transactions that were completed in the entire run
Rate [per second] [for Interval]	The number of transactions that were completed per second in the last recorded interval
Rate [per second] [for Run]	The number of transactions that were completed per second in the entire run

The counters in the following table provide individual values for each transaction. Folders that contain individual counters have an asterisk superimposed on them: .

Counter name	Description
Total Transactions Completed [for Run]	The total number of transactions that were completed
Transaction Completion Rate [per second] [for Interval]	The rate that transactions were completed in the last recorded interval
Transaction Completion Rate [per second] [for Run]	The rate that transactions were completed in the entire run

The counters in the following table provide an aggregate value for all transactions. Folders that contain aggregate counters have a clock superimposed on them: .

Counter name	Description
Average [for Interval]	The average elapsed time for all transactions within the most recent sample interval
Average [for Run]	The average elapsed time for all transactions in the entire run
Maximum [for Interval]	The maximum elapsed time for all transactions within the most recent sample interval
Maximum [for Run]	The maximum elapsed time for all transactions in the entire run
Minimum [for Interval]	The minimum elapsed time for all transactions within the most recent sample interval
Minimum [for Run]	The minimum elapsed time for all transactions in the entire run
Standard deviation [for Interval]	The standard deviation for all transactions within the most recent sample interval
Standard deviation [for Run]	The standard deviation for all transactions for the entire run

The counters in the following table provide individual values for each transaction. Folders that contain individual counters have an asterisk superimposed on them: .

Counter name	Description
Average Elapsed Time for All Transactions [ms] [for Interval]	The average elapsed time for each transaction within the most recent sample interval
Average Elapsed Time for All Transactions [ms] [for Run]	The average elapsed time for each transaction in the entire run

Counter name	Description
Elapsed Time Standard Deviation for All Transactions [for Interval]	The standard deviation for each transaction within the most recent sample interval
Elapsed Time Standard Deviation for All Transactions [for Run]	The standard deviation for each transaction in the entire run
Maximum Elapsed Time for All Transactions [ms] [for Interval]	The maximum elapsed time for each transaction within the most recent sample interval
Maximum Elapsed Time for All Transactions [ms] [for Run]	The maximum elapsed time for each transaction in the entire run
Minimum Elapsed Time for All Transactions [ms] [for Interval]	The minimum elapsed time for each transaction within the most recent sample interval
Minimum Elapsed Time for All Transactions [ms] [for Run]	The minimum elapsed time for each transaction in the entire run

Recording with Internet Protocol v6

To record Internet Protocol version 6 traffic, set the HTTP recording preferences to **Firefox** and select **Record using IPv6**.

1. Click **Windows > Preferences > Test > RPT HTTP recording**.
2. In **Application to record**, select **Firefox**.
You cannot use Internet Explorer to record IPv6.
3. Select **Record using IPv6**.

Result

You are now ready to record IPv6 traffic.

Changing HTTP recording preferences

You can change the behavior of the recorder by changing the preference settings. The default settings, however, are appropriate for recording under Windows® or Linux® systems.

1. Open the HTTP Recording page. Click **Window > Preferences > Test > Recording > Browsers Recording**.
2. Select the setting to change.

Enable the Rational® Performance Tester toolbar in browsers

Click to install the annotation toolbar. This enables you to add comments and transactions, and to change page names during recording.

Verify annotation toolbar is installed before recording

Click to verify that the annotation toolbar is installed in the web browser before recording.


3. After changing a setting, click **Apply**.


SAP counters

SAP generic counters, which are displayed in the Performance Test Runs view, enable you to customize your SAP reports with dynamic information that is updated for each run.

SAP screen counters

These counters provide information about screens started, screens completed, response times, and verification points. Some counters produce an aggregate value, where the values for several screens are rolled up into one value; others produce values for each screen.

Aggregate counters use the values for all the screens in a test to produce a single value for a report. This value is rolled up from all values that satisfy the counter. When you drag an aggregate counter onto a report, one value is displayed. Folders that contain aggregate counters have a clock superimposed on the folder icon: . The tables below that list aggregate counters have this icon in their title.

Individual counters produce values for each item that satisfies the counter, rather than a single rolled-up value. Folders that contain individual counters have an asterisk superimposed on the folder icon: . The tables below that list individual counters have this icon in their title.

Some counters pertain to *intervals* in the run. You set the **Statistics sample interval** value in the schedule, as a schedule property.

SAP Screen Request Response Time counters

The counters in this section provide information about the SAP application response time. This is the time that elapses from the point that the SAP GUI client sends a request and the moment the response is received and displayed by the SAP GUI client.

Counter name	Description
Average [for Interval]	The average response time for a specific SAP screen during the most recent sample interval
Average [for Run]	The average response time for a specific SAP screen for the entire run
Maximum [for Interval]	The maximum response time for a specific SAP screen during the most recent sample interval
Maximum [for Run]	The maximum response time for a specific SAP screen for the entire run
Minimum [for Interval]	The minimum response time for a specific SAP screen during the most recent sample interval
Minimum [for Run]	The minimum response time for a specific SAP screen for the entire run
Standard Deviation [for Interval]	The standard deviation of response times for a specific SAP screen during the most recent sample interval

Counter name	Description
Standard Deviation [for Run]	The standard deviation of response times for a specific SAP screen for the entire run

Counter name	Description
Average Screen Request Response Time for All SAP Screens [ms] [for Interval]	The average response time for all SAP screens within the most recent sample interval
Average Screen Request Response Time for All SAP Screens [ms] [for Run]	The average response time for all SAP screens for the entire run
Maximum Screen Request Response Time for All SAP Screens [ms] [for Interval]	The maximum response time for all SAP screens within the most recent sample interval
Maximum Screen Request Response Time for All SAP Screens [ms] [for Run]	The maximum response time for all SAP screens for the entire run
Minimum Screen Request Response Time for All SAP Screens [ms] [for Interval]	The minimum response time for all SAP screens within the most recent sample interval
Minimum Screen Request Response Time for All SAP Screens [ms] [for Run]	The minimum response time for all SAP screens for the entire run
Standard Deviation Screen Request Response Time for All SAP Screens [ms] [for Interval]	The standard deviation of response times for all SAP screens during the most recent sample interval
Standard Deviation Screen Request Response Time for All SAP Screens [ms] [for Run]	The standard deviation of response times for all SAP screens for the entire run

SAP Screen Request Interpretation Time counters

The counters in this section provide information about the SAP GUI interpretation time. This is the duration from the time that the SAP GUI client receives data from the server and the moment when this data is displayed. This measures the health of the SAP GUI client used for the test rather than the performance of the SAP server.

Counter name	Description
Average [for Interval]	The average interpretation time for a specific SAP screen during the most recent sample interval
Average [for Run]	The average interpretation time for a specific SAP screen for the entire run
Maximum [for Interval]	The maximum interpretation time for a specific SAP screen during the most recent sample interval
Maximum [for Run]	The maximum interpretation time for a specific SAP screen for the entire run

Counter name	Description
Minimum [for Interval]	The minimum interpretation time for a specific SAP screen during the most recent sample interval
Minimum [for Run]	The minimum interpretation time for a specific SAP screen for the entire run
Standard Deviation [for Interval]	The standard deviation of interpretation times for a specific SAP screen during the most recent sample interval
Standard Deviation [for Run]	The standard deviation of interpretation times for a specific SAP screen for the entire run

Counter name	Description
Average Screen Request Interpretation Time for All SAP Screens [ms] [for Interval]	The average interpretation time for all SAP screens during the most recent sample interval
Average Screen Request Interpretation Time for All SAP Screens [ms] [for Run]	The average interpretation time for all SAP screens for the entire run
Maximum Screen Request Interpretation Time for All SAP Screens [ms] [for Interval]	The maximum interpretation time for all SAP screens during the most recent sample interval
Maximum Screen Request Interpretation Time for All SAP Screens [ms] [for Run]	The maximum interpretation time for all SAP screens for the entire run
Minimum Screen Request Interpretation Time for All SAP Screens [ms] [for Interval]	The minimum interpretation time for all SAP screens during the most recent sample interval
Minimum Screen Request Interpretation Time for All SAP Screens [ms] [for Run]	The minimum interpretation time for all SAP screens for the entire run
Standard Deviation Screen Request Interpretation Time for All SAP Screens [ms] [for Interval]	The standard deviation of interpretation times for all SAP screens during the most recent sample interval
Standard Deviation Screen Request Interpretation Time for All SAP Screens [ms] [for Run]	The standard deviation of interpretation times for all SAP screens for the entire run

SAP Screens Started counters

The counters in this section provide information about screens that are started.

Counter name	Description
Count [for Interval]	The number of started screens for a specific SAP screen during the most recent sample interval

Counter name	Description
Count [for Run]	The number of started screens for a specific SAP screen for the entire run
Rate [per second] [for Interval]	The rate at which screens were started for a specific SAP screen during the most recent sample interval
Rate [per second] [for Run]	The rate at which screens were started for a specific SAP screen for the entire run

Counter name	Description
SAP Screens Started Rate [for Interval]	The rate at which SAP screens were started within the most recent sample interval
SAP Screens Started Rate [for Run]	The rate at which SAP screens were started for the entire run
Total SAP Screens Started [for Interval]	The number of started SAP screens for the entire run
Total SAP Screens Started [for Run]	The number of started SAP screens during the most recent sample interval

SAP Screens Completed counters

The counters in this section provide information about SAP screens that were completed.

Counter name	Description
Count [for Interval]	The number of completed screens for a specific SAP screen during the most recent sample interval
Count [for Run]	The number of completed screens for a specific SAP screen for the entire run
Rate [per second] [for Interval]	The rate at which screens were completed for a specific SAP screen during the most recent sample interval
Rate [per second] [for Run]	The rate at which screens were completed for a specific SAP screen for the entire run

Counter name	Description
SAP Screens Completed Percent [for Interval]	The percentage of completed SAP screens within the most recent sample interval
SAP Screens Completed Percent [for Run]	The percentage of completed SAP screens for the entire run

Counter name	Description
SAP Screens Completed Rate [for Interval]	The rate at which SAP screens were completed within the most recent sample interval
SAP Screens Completed Rate [for Run]	The rate at which SAP screens were completed for the entire run
Total SAP Screens Completed [for Interval]	The number of completed SAP screens within the most recent sample interval
Total SAP Screens Completed [for Run]	The number of completed SAP screens for the entire run

Verification Point counters

The tables in this section provide information about SAP screen verification points. The verdict for a verification point can be `Pass`, `Fail`, `Error`, or `Inconclusive`.

- `Pass` indicates that the verification point matched or received the expected response. For example, a response code verification point is set to `Pass` when the recorded response code is received during playback. If your test does not contain verification points, it means that the connection succeeded.
- `Fail` indicates that the verification point did not match the expected response or that the expected response was not received.
- `Error` indicates that the primary request was not successfully sent to the server, no response was received from the server, or the response was incomplete or could not be parsed.
- The verdict is set to `Inconclusive` only if you provide custom code that defines a verdict of `Inconclusive`.

Counter name	Description
Percent Pass	The percentage of verification points that passed for a specific SAP screen for the entire run

Counter name	Description
Count [for Interval]	The number of verification points classified as <code>Error</code> for a specific SAP screen during the most recent sample interval
Count [for Run]	The number of verification points classified as <code>Error</code> for a specific SAP screen for the entire run

Counter name	Description
Count [for Interval]	The number of verification points that failed for a specific SAP screen during the most recent sample interval
Count [for Run]	The number of verification points that failed for a specific SAP screen for the entire run

Counter name	Description
Count [for Interval]	The number of verification points classified as <i>Inconclusive</i> for a specific SAP screen during the most recent sample interval
Count [for Run]	The number of verification points classified as <i>Inconclusive</i> for a specific SAP screen for the entire run

Counter name	Description
Count [for Interval]	The number of verification points that passed for a specific SAP screen during the most recent sample interval
Count [for Run]	The number of verification points that passed for a specific SAP screen for the entire run

Counter name	Description
Percent SAP Screens VPs Passed [for Interval]	The percentage of verification points that passed within the most recent sample interval
Percent SAP Screens VPs Passed [for Run]	The percentage of verification points that passed for the entire run
Total SAP Screens VPs Attempted [for Interval]	The number of verification points executed within the most recent sample interval
Total SAP Screens VPs Attempted [for Run]	The number of verification points executed for the entire run
Total SAP Screens VPs Error [for Interval]	The number of verification points with a verdict of <i>Error</i> within the most recent sample interval
Total SAP Screens VPs Error [for Run]	The number of verification points with a verdict of <i>Error</i> for the entire run
Total SAP Screens VPs Failed [for Interval]	The number of verification points that failed within the most recent sample interval
Total SAP Screens VPs Failed [for Run]	The number of verification points that failed for the entire run
Total SAP Screens VPs Inconclusive [for Interval]	The number of verification points that were marked as <i>Inconclusive</i> within the most recent sample interval
Total SAP Screens VPs Inconclusive [for Run]	The number of verification points that were marked as <i>Inconclusive</i> for the entire run

Counter name	Description
Total SAP Screens VPs Passed [for Interval]	The number of verification points that passed within the most recent sample interval
Total SAP Screens VPs Passed [for Run]	The number of verification points that passed for the entire run

SAP Elements Attempted counters

The counters in this section provide information about SAP set elements, SAP get elements or SAP calls that were attempted.

Counter name	Description
Count [for Interval]	The number of attempted elements for a specific SAP screen during the most recent sample interval
Count [for Run]	The number of attempted elements for a specific SAP screen for the entire run
Rate [per second] [for Interval]	The rate at which elements were attempted for a specific SAP screen within the most recent sample interval
Rate [per second] [for Run]	The rate at which elements were attempted for a specific SAP screen for the entire run

Counter name	Description
SAP Elements Attempted Rate [for Interval]	The rate at which SAP elements were attempted within the most recent sample interval
SAP Elements Attempted Rate [for Run]	The rate at which SAP elements were attempted for the entire run
Total SAP Elements Attempted [for Interval]	The number of attempted SAP elements for the entire run
Total SAP Elements Attempted [for Run]	The number of attempted SAP elements within the most recent sample interval

SAP Elements Completed counters

The counters in this section provide information about SAP set elements, SAP get elements or SAP calls that terminated normally. If an element does not complete, it is counted in the SAP Elements Attempted, but not in SAP Elements Complete.

Counter name	Description
Count [for Interval]	The number of completed elements for a specific SAP screen during the most recent sample interval
Count [for Run]	The number of completed elements for a specific SAP screen for the entire run
Rate [per second] [for Interval]	The rate at which elements were completed for a specific SAP screen during the most recent sample interval
Rate [per second] [for Run]	The rate at which elements were completed for a specific SAP screen for the entire run

Counter name	Description
SAP Elements Completed Percent [for Interval]	The percentage of completed SAP elements within the most recent sample interval
SAP Elements Completed Percent [for Run]	The percentage of completed SAP elements for the entire run
SAP Elements Completed Rate [for Interval]	The rate at which SAP elements were completed within the most recent sample interval
SAP Elements Completed Rate [for Run]	The rate at which SAP elements were completed for the entire run
Total SAP Elements Completed [for Interval]	The number of completed SAP elements within the most recent sample interval
Total SAP Elements Completed [for Run]	The number of completed SAP elements for the entire run



Citrix counters

Citrix generic counters, which are displayed in the Performance Test Runs view, enable you to customize your Citrix reports with dynamic information that is updated for each run.


Citrix window synchronization counters

Citrix window synchronization counters provide information about window synchronizations that were attempted, completed, or produced a timeout within the specified timeout limit. Window synchronization is the mechanism used by the test to compare windows from the test with actual windows in the run.


Some counters produce an aggregate value, where the values for several web service message returns are combined into a single value; others produce values for each web service message return.

- *Aggregate* counters use the values for all the elements in a test to produce a single value for a report. This value is rolled up from all values that satisfy the counter. When you drag an aggregate counter onto a report, one value is displayed. Folders that contain aggregate counters have a clock superimposed on the folder icon: . In the tables that follow, the aggregate counters are listed after the titles that contain this icon.
- *Individual* counters produce values for each item that satisfies the counter, rather than an aggregate value. Folders that contain individual counters have an asterisk superimposed on the folder icon: . In the tables that follow, the individual counters are listed after the titles that contain this icon.


Some counters pertain to *intervals* in the run. You set the **Statistics sample interval** value in the schedule, as a schedule property.

The counters in the following table provide an aggregate value for all window synchronizations. Folders that contain aggregate counters have a clock superimposed on them: .

Counter name	Description
Count [for Interval]	The total number of window synchronizations that were attempted within the last recorded interval
Count [for Run]	The total number of window synchronizations that were attempted in the entire run

The counters in the following table provide an aggregate value for all window synchronizations. Folders that contain aggregate counters have a clock superimposed on them: .

Counter name	Description
Count [for Interval]	The total number of window synchronizations that succeeded within the last recorded interval
Count [for Run]	The total number of window synchronizations that succeeded in the entire run
Percent Citrix Synchronization Success for Interval	The percentage of window synchronizations that succeeded within the last recorded interval
Percent Citrix Synchronization Success for Run	The percentage of window synchronizations that succeeded in the entire run



The counters in the following table provide an aggregate value for all transactions. Folders that contain aggregate counters have a clock superimposed on them: .

Counter name	Description
Count [for Interval]	The total number of window synchronizations that produced a timeout within the last recorded interval
Count [for Run]	The total number of window synchronizations that produced a timeout in the entire run


Citrix image synchronization counters

Citrix image synchronization counters provide information about image synchronizations that were attempted, completed, or produced a timeout within the specified timeout limit. Image synchronization is the mechanism used by the test to compare image areas from the test with actual image area in the run.


Some counters produce an aggregate value, where the values for several web service message returns are combined into a single value; others produce values for each web service message return.

- *Aggregate* counters use the values for all the elements in a test to produce a single value for a report. This value is rolled up from all values that satisfy the counter. When you drag an aggregate counter onto a report, one value is displayed. Folders that contain aggregate counters have a clock superimposed on the folder icon: . In the tables that follow, the aggregate counters are listed after the titles that contain this icon.
- *Individual* counters produce values for each item that satisfies the counter, rather than an aggregate value. Folders that contain individual counters have an asterisk superimposed on the folder icon: . In the tables that follow, the individual counters are listed after the titles that contain this icon.


Some counters pertain to *intervals* in the run. You set the **Statistics sample interval** value in the schedule, as a schedule property.

The counters in the following table provide an aggregate value for all image synchronizations. Folders that contain aggregate counters have a clock superimposed on them: .

Counter name	Description
Count [for Interval]	The total number of image synchronizations that were attempted within the last recorded interval
Count [for Run]	The total number of image synchronizations that were attempted in the entire run

The counters in the following table provide an aggregate value for all image synchronizations. Folders that contain aggregate counters have a clock superimposed on them: .

Counter name	Description
Count [for Interval]	The total number of image synchronizations that succeeded within the last recorded interval
Count [for Run]	The total number of image synchronizations that succeeded in the entire run
Percent Citrix Synchronization Success for Interval	The percentage of image synchronizations that succeeded within the last recorded interval
Percent Citrix Synchronization Success for Run	The percentage of image synchronizations that succeeded in the entire run

The counters in the following table provide an aggregate value for all transactions. Folders that contain aggregate counters have a clock superimposed on them: .

Counter name	Description
Count [for Interval]	The total number of image synchronizations that produced a timeout within the last recorded interval
Count [for Run]	The total number of image synchronizations that produced a timeout in the entire run

Counter name	Description
Percent Pass	The percentage of image synchronization verification points that passed for a specific window event for the entire run

Counter name	Description
Count [for Interval]	The number of image synchronization verification points classified as <code>Error</code> for a specific window event during the most recent sample interval
Count [for Run]	The number of image synchronization verification points classified as <code>Error</code> for a specific window for the entire run

Counter name	Description
Count [for Interval]	The number of image synchronization verification points that failed for a specific window event during the most recent sample interval
Count [for Run]	The number of image synchronization verification points that failed for a specific window event for the entire run

Counter name	Description
Count [for Interval]	The number of image synchronization verification points classified as <i>Inconclusive</i> for a specific window event during the most recent sample interval
Count [for Run]	The number of image synchronization verification points classified as <i>Inconclusive</i> for a specific window event for the entire run

Counter name	Description
Count [for Interval]	The number of image synchronization verification points that passed for a specific window event during the most recent sample interval
Count [for Run]	The number of image synchronization verification points that passed for a specific window event for the entire run



Counter name	Description
Percent Image Synchronization VPs Passed For Interval	The percentage of image synchronization verification points that passed during the most recent sample interval
Percent Image Synchronization VPs Passed For Run	The percentage of image synchronization verification points that passed for the entire run
Total Image Synchronization VPs Attempted [for Interval]	The number of image synchronization verification points executed during the most recent sample interval
Total Image Synchronization VPs Attempted [for Run]	The number of image synchronization verification points executed for the entire run
Total Image Synchronization VPs Error [for Interval]	The number of image synchronization verification points with a verdict of <i>ERROR</i> during the most recent sample interval

Counter name	Description
Total Image Synchronization VPs Error [for Run]	The number of image synchronization verification points with a verdict of <code>ERROR</code> for the entire run
Total Image Synchronization VPs Failed [for Interval]	The number of image synchronization verification points that failed during the most recent sample interval
Total Image Synchronization VPs Failed [for Run]	The number of image synchronization verification points that failed for the entire run
Total Image Synchronization VPs Inconclusive [for Interval]	The number of image synchronization verification points that were marked as <code>Inconclusive</code> within the most recent sample interval
Total Image Synchronization VPs Inconclusive [for Run]	The number of image synchronization verification points that were marked as <code>Inconclusive</code> for the entire run
Total Image Synchronization VPs Passed [for Interval]	The number of image synchronization verification points that passed during the most recent sample interval
Total Image Synchronization VPs Passed [for Run]	The number of image synchronization verification points that passed for the entire run


Citrix timer counters

Citrix timer counters provide information about the response time of window events in a run. *Response times* are determined by measurements that are located in the tests.


Some counters produce an aggregate value, where the values for several web service message returns are combined into a single value; others produce values for each web service message return.

- *Aggregate* counters use the values for all the elements in a test to produce a single value for a report. This value is rolled up from all values that satisfy the counter. When you drag an aggregate counter onto a report, one value is displayed. Folders that contain aggregate counters have a clock superimposed on the folder icon: . In the tables that follow, the aggregate counters are listed after the titles that contain this icon.
- *Individual* counters produce values for each item that satisfies the counter, rather than an aggregate value. Folders that contain individual counters have an asterisk superimposed on the folder icon: . In the tables that follow, the individual counters are listed after the titles that contain this icon.

Some counters pertain to *intervals* in the run. You set the **Statistics sample interval** value in the schedule, as a schedule property.

The counters in the following table provide an aggregate value for all tests. Folders that contain aggregate counters have a clock superimposed on them: .

Counter name	Description
Average Response Time For All Timers For Interval [ms]	The average response time for all measurements within the most recent sample interval
Average Response Time For All Timers For Run [ms]	The average response time for all measurements in the entire run
Maximum Response Time For All Timers For Interval [ms]	The maximum response time for all measurements within the most recent sample interval
Maximum Response Time For All Timers For Run [ms]	The maximum response time for all measurements in the entire run
Minimum Response Time For All Timers For Interval [ms]	The minimum response time for all measurements within the most recent sample interval
Minimum Response Time For All Timers For Run [ms]	The minimum response time for all measurements in the entire run
Standard Deviation Response Time For All Timers For Interval [ms]	The standard deviation response time for all measurements during the most recent sample interval
Standard Deviation Response Time For All Timers For Run [ms]	The standard deviation response time for all measurements in the entire run

The counters in the following table provide individual values for each test. Folders that contain individual counters have an asterisk superimposed on them: .


Counter name	Description
Average [for Interval]	The average response time for all measurements within the most recent sample interval
Average [for Run]	The average response time for all measurements in the entire run
Maximum [for Interval]	The maximum response time for all measurements within the most recent sample interval
Maximum [for Run]	The maximum response time for all measurements in the entire run
Minimum [for Interval]	The minimum response time for all measurements within the most recent sample interval
Minimum [for Run]	The minimum response time for all measurements in the entire run
Standard Deviation [for Interval]	The standard deviation response time for all measurements during the most recent sample interval

Counter name	Description
Standard Deviation [for Run]	The standard deviation response time for all measurements in the entire run


Citrix user action counters

Citrix user action counters provide information about emulated user input actions in the run.

Some counters pertain to *intervals* in the run. You set the **Statistics sample interval** value in the schedule, as a schedule property.

The counters in the following table provide an aggregate value for all transactions. Folders that contain aggregate counters have a clock superimposed on them: .

Counter Name	Description
Count [for Interval]	The total number of user input actions within the most recent sample interval
Count [for Run]	The total number of user input actions in the entire run
Rate [per second] [for Interval]	The rate per second of user input actions within the most recent sample interval
Rate [per second] [for Run]	The rate per second of user input actions in the entire run

The counters in the following table provide individual values for each test. Folders that contain individual counters have an asterisk superimposed on them: .

Counter Name	Description
Total User Actions For Interval	The total number of user input actions within the most recent sample interval
Total User Actions For Run	The total number of user input actions in the entire run
User Action Rate For Interval	The rate per second of user input actions within the most recent sample interval
User Action Rate For Run	The rate per second of user input actions in the entire run



Citrix window counters

Citrix window counters provide information about verification points.

The verdict for a verification point can be `Pass`, `Fail`, `Error`, or `Inconclusive`.

- `Pass` indicates that the verification point matched or received the expected response. For example, a response code verification point is set to `Pass` when the recorded response code is received during the run. If your test does not contain verification points, it means that the connection succeeded.
- `Fail` indicates that the verification point did not match the expected response or that the expected response was not received.
- `Error` indicates that the primary request was not successfully sent to the server, no response was received from the server, or the response was incomplete or could not be parsed.
- The verdict is set to `Inconclusive` only if you provide custom code that defines a verdict of `Inconclusive`.

Some counters produce an aggregate value, where the values for several window events are rolled up into one value; others produce values for each window event.

- *Aggregate* counters use the values for all the elements in a test to produce a single value for a report. This value is rolled up from all values that satisfy the counter. When you drag an aggregate counter onto a report, one value is displayed. Folders that contain aggregate counters have a clock superimposed on the folder icon: . The tables below that list aggregate counters have this icon in their title.
- *Individual* counters produce values for each item that satisfies the counter, rather than a single rolled-up value. Folders that contain individual counters have an asterisk superimposed on the folder icon: . The tables below that list individual counters have this icon in their title.

Some counters pertain to *intervals* in the run. You set the **Statistics sample interval** value in the schedule, as a schedule property.

Counter name	Description
Percent Pass	The percentage of window title verification points that passed for a specific window event for the entire run

Counter name	Description
Count [for Interval]	The number of window title verification points classified as <code>Error</code> for a specific window event within the most recent sample interval
Count [for Run]	The number of window title verification points classified as <code>Error</code> for a specific window for the entire run

Counter name	Description
Count [for Interval]	The number of window title verification points that failed for a specific window event during the most recent sample interval
Count [for Run]	The number of window title verification points that failed for a specific window event for the entire run

Counter name	Description
Count [for Interval]	The number of window title verification points classified as <code>Inconclusive</code> for a specific window event within the most recent sample interval
Count [for Run]	The number of window title verification points classified as <code>Inconclusive</code> for a specific window event for the entire run

Counter name	Description
Count [for Interval]	The number of window title verification points that passed for a specific window event during the most recent sample interval
Count [for Run]	The number of window title verification points that passed for a specific window event for the entire run

Counter name	Description
Percent Window VPs Passed For Interval	The percentage of window title verification points that passed during the most recent sample interval
Percent Window VPs Passed For Run	The percentage of window title verification points that passed for the entire run
Total Window VPs Attempted [for Interval]	The number of window title verification points executed during the most recent sample interval
Total Window VPs Attempted [for Run]	The number of window title verification points executed for the entire run
Total Window VPs Error [for Interval]	The number of window title verification points with a verdict of <code>Error</code> during the most recent sample interval
Total Window VPs Error [for Run]	The number of window title verification points with a verdict of <code>Error</code> for the entire run

Counter name	Description
Total Window VPs Failed [for Interval]	The number of window title verification points that failed within the most recent sample interval
Total Window VPs Failed [for Run]	The number of window title verification points that failed for the entire run
Total Window VPs Inconclusive [for Interval]	The number of window title verification points that were marked as <i>Inconclusive</i> during the most recent sample interval
Total Window VPs Inconclusive [for Run]	The number of window title verification points that were marked as <i>Inconclusive</i> for the entire run
Total Window VPs Passed [for Interval]	The number of window title verification points that passed during the most recent sample interval
Total Window VPs Passed [for Run]	The number of window title verification points that passed for the entire run



Service counters

Service counters, which are displayed in the Test Runs view, enable you to customize your service test reports with dynamic information that is updated for each run.

Service call counters

Service call counters provide information about the calls that are invoked during the service test.

Some counters produce an aggregate value, where the values for several service calls are combined into a single value; others produce values for each service call.

- *Aggregate* counters use the values for all the elements in a test to produce a single value for a report. This value combines all values that satisfy the counter. When you drag an aggregate counter onto a report, one value is displayed. Folders that contain aggregate counters have a clock superimposed on the folder icon: . In the tables that follow, the aggregate counters are listed after the titles that contain this icon.
- *Individual* counters produce values for each item that satisfies the counter, rather than an aggregate value. Folders that contain individual counters have an asterisk superimposed on the folder icon: . In the tables that follow, the individual counters are listed after the titles that contain this icon.

Some counters pertain to *intervals* in the run. You set the **Statistics sample interval** value in the schedule, as a schedule property.

Counter name	Description
Average [for Interval]	The average time, in milliseconds, required to establish a connection to the web service for all web service calls within the most recent sample interval
Average [for Run]	The average time, in milliseconds, required to establish a connection to the web service for all web service calls for the entire run
Standard deviation [for Interval]	The standard deviation time, in milliseconds, required to establish a connection to the web service for all web service calls within the most recent sample interval
Standard deviation [for Run]	The standard deviation time, in milliseconds, required to establish a connection to the web service for all web service calls for the entire run
Maximum [for Interval]	The longest time, in milliseconds, required to establish a connection to the web service for all web service calls within the most recent sample interval
Maximum [for Run]	The longest time, in milliseconds, required to establish a connection to the web service for all web service calls for the entire run
Minimum [for Interval]	The shortest time, in milliseconds, required to establish a connection to the web service for all web service calls within the most recent sample interval
Minimum [for Run]	The shortest time, in milliseconds, required to establish a connection to the web service for all web service calls for the entire run

Counter name	Description
Average connection time [ms] [for Interval]	The average time, in milliseconds, required to establish a connection to the web service for a specific web service call within the most recent sample interval
Average connection time [ms] [for Run]	The average time, in milliseconds, required to establish a connection to the web service for a specific web service call for the entire run
Connection time standard deviation [ms] [for Interval]	The standard deviation time, in milliseconds, required to establish a connection to the web service for a specific web service call during the most recent sample interval
Connection time standard deviation [ms] [for Run]	The standard deviation time, in milliseconds, required to establish a connection to the web service for a specific web service call for the entire run
Maximum connection time [ms] [for Interval]	The longest time, in milliseconds, required to establish a connection to the web service for a specific web service call within the most recent sample interval
Maximum connection time [ms] [for Run]	The longest time, in milliseconds, required to establish a connection to the web service for a specific web service call for the entire run
Minimum connection time [ms] [for Interval]	The shortest time, in milliseconds, required to establish a connection to the web service for a specific web service call within the most recent sample interval

Counter name	Description
Minimum connection time [ms] [for Run]	The shortest time, in milliseconds, required to establish a connection to the web service for a specific web service call for the entire run

Counter name	Description
Average [for Interval]	The average data volume, in bytes, received as a response from the web service for all web service calls during the most recent sample interval
Average [for Run]	The average data volume, in bytes, received as a response from the web service for all web service calls for the entire run
Count [for Interval]	The total data volume, in bytes, received as a response from the web service for all web service calls during the most recent sample interval
Count [for Run]	The total data volume, in bytes, received as a response from the web service for all web service calls for the entire run
Maximum [for Interval]	The largest data volume, in bytes, received as a response from the web service for all web service calls during the most recent sample interval
Maximum [for Run]	The largest data volume, in bytes, received as a response from the web service for all web service calls for the entire run
Maximum Count [for Run]	The largest cumulated data volume, in bytes, received as a response from the web service for all web service calls for the entire run
Minimum [for Interval]	The smallest data volume, in bytes, received as a response from the web service for all web service calls during the most recent sample interval
Minimum [for Run]	The smallest data volume, in bytes, received as a response from the web service for all web service calls for the entire run
Minimum Count [for Run]	The smallest cumulated data volume, in bytes, received as a response from the web service for all web service calls for the entire run
Rate [per second] [for Interval]	The data volume throughput, in bytes per second, received as a response from the web service for all web service calls within the most recent sample interval
Rate [per second] [for Run]	The data volume throughput, in bytes per second, received as a response from the web service for all web service calls for the entire run
Standard deviation [for Interval]	The standard deviation data volume, in bytes, received as a response from the web service for all web service calls within the most recent sample interval
Standard deviation [for Run]	The standard deviation data volume, in bytes, received as a response from the web service for all web service calls for the entire run

Counter name	Description
Average received bytes for all calls [Bytes] [for Interval]	The average data volume, in bytes, received as a response from the web service for all web service calls during the most recent sample interval
Average received bytes for all calls [Bytes] [for Run]	The average data volume, in bytes, received as a response from the web service for all web service calls for the entire run
Maximum received bytes [Bytes] [for Interval]	The largest data volume, in bytes, received as a response from the web service for a specific web service call during the most recent sample interval
Maximum received bytes [per Interval] [for Run]	The largest data volume, in bytes, received as a response from the web service for a specific web service call for the entire run
Maximum received bytes for all calls [Bytes] [for Interval]	The largest data volume, in bytes, received as a response from the web service for all web service calls during the most recent sample interval
Maximum received bytes for all calls [Bytes] [for Run]	The largest data volume, in bytes, received as a response from the web service for all web service calls for the entire run
Minimum received bytes [Bytes] [for Interval]	The smallest data volume, in bytes, received as a response from the web service for a specific web service call during the most recent sample interval
Minimum received bytes [per Interval] [for Run]	The smallest data volume, in bytes, received as a response from the web service for a specific web service call for the entire run
Minimum received bytes for all calls [Bytes] [for Interval]	The smallest data volume, in bytes, received as a response from the web service for a specific web service call during the most recent sample interval
Minimum received bytes for all calls [Bytes] [for Run]	The smallest data volume, in bytes, received as a response from the web service for a specific web service call for the entire run
Rate received bytes [for Interval]	The data volume throughput, in bytes per second, received as a response from the web service for all web service calls within the most recent sample interval
Rate received bytes [for Run]	The data volume throughput, in bytes per second, received as a response from the web service for a specific web service call for the entire run
Received bytes standard deviation [ms] [for Interval]	The standard deviation data volume, in bytes, received as a response from the web service for a specific web service call within the most recent sample interval
Received bytes standard deviation [ms] [for Run]	The standard deviation data volume, in bytes, received as a response from the web service for a specific web service call for the entire run
Total received bytes [Bytes] [for Interval]	The total data volume, in bytes, received as a response from the web service for a specific web service call during the most recent sample interval
Total received bytes [Bytes] [for Run]	The total data volume, in bytes, received as a response from the web service for a specific web service call for the entire run

Counter name	Description
Average [for Interval]	The average data volume, in bytes, sent as a call from the web service for all web service calls during the most recent sample interval
Average [for Run]	The average data volume, in bytes, sent as a call from the web service for all web service calls for the entire run
Count [for Interval]	The total data volume, in bytes, sent as a call from the web service for all web service calls within the most recent sample interval
Count [for Run]	The total data volume, in bytes, sent as a call from the web service for all web service calls for the entire run
Maximum [for Interval]	The largest data volume, in bytes, sent as a call from the web service for all web service calls during the most recent sample interval
Maximum [for Run]	The largest data volume, in bytes, sent as a call from the web service for all web service calls for the entire run
Maximum Count [for Run]	The largest cumulated data volume, in bytes, sent as a call from the web service for all web service calls for the entire run
Minimum [for Interval]	The smallest data volume, in bytes, sent as a call from the web service for all web service calls during the most recent sample interval
Minimum [for Run]	The smallest data volume, in bytes, sent as a call from the web service for all web service calls for the entire run
Minimum Count [for Run]	The smallest cumulated data volume, in bytes, sent as a call from the web service for all web service calls for the entire run
Rate [per second] [for Interval]	The data volume throughput, in bytes per second, sent as a call from the web service for all web service calls during the most recent sample interval
Rate [per second] [for Run]	The data volume throughput, in bytes per second, sent as a call from the web service for all web service calls for the entire run
Standard deviation [for Interval]	The standard deviation data volume, in bytes, sent as a call from the web service for all web service calls during the most recent sample interval
Standard deviation [for Run]	The standard deviation data volume, in bytes, sent as a call from the web service for all web service calls for the entire run

Counter name	Description
Average sent bytes for all calls [Bytes] [for Interval]	The average data volume, in bytes, sent as a call from the web service for all web service calls during the most recent sample interval

Counter name	Description
Average sent bytes for all calls [Bytes] [for Run]	The average data volume, in bytes, sent as a call from the web service for all web service calls for the entire run
Maximum sent bytes [Bytes] [for Interval]	The largest data volume, in bytes, sent as a call from the web service for a specific web service call during the most recent sample interval
Maximum sent bytes [per Interval] [for Run]	The largest data volume, in bytes, sent as a call from the web service for a specific web service call for the entire run
Maximum sent bytes for all calls [Bytes] [for Interval]	The largest data volume, in bytes, sent as a call from the web service for all web service calls during the most recent sample interval
Maximum sent bytes for all calls [Bytes] [for Run]	The largest data volume, in bytes, sent as a call from the web service for all web service calls for the entire run
Minimum sent bytes [Bytes] [for Interval]	The smallest data volume, in bytes, sent as a call from the web service for a specific web service call during the most recent sample interval
Minimum sent bytes [per Interval] [for Run]	The smallest data volume, in bytes, sent as a call from the web service for a specific web service call for the entire run
Minimum sent bytes for all calls [Bytes] [for Interval]	The smallest data volume, in bytes, sent as a call from the web service for a specific web service call during the most recent sample interval
Minimum sent bytes for all calls [Bytes] [for Run]	The smallest data volume, in bytes, sent as a call from the web service for a specific web service call for the entire run
Rate sent bytes [for Interval]	The data volume throughput, in bytes per second, sent as a call from the web service for all web service calls during the most recent sample interval
Rate sent bytes [for Run]	The data volume throughput, in bytes per second, sent as a call from the web service for a specific web service call for the entire run
Sent bytes standard deviation [ms] [for Interval]	The standard deviation data volume, in bytes, sent as a call from the web service for a specific web service call within the most recent sample interval
Sent bytes standard deviation [ms] [for Run]	The standard deviation data volume, in bytes, sent as a call from the web service for a specific web service call for the entire run
Total sent bytes [Bytes] [for Interval]	The total data volume, in bytes, sent as a call from the web service for a specific web service call during the most recent sample interval
Total sent bytes [Bytes] [for Run]	The total data volume, in bytes, sent as a call from the web service for a specific web service call for the entire run

Counter name	Description
Average [for Interval]	The average time, in milliseconds, required to receive a response from the web service for all web service calls within the most recent sample interval
Average [for Run]	The average time, in milliseconds, required to receive a response from the web service for all web service calls for the entire run
Standard deviation [for Interval]	The standard deviation time, in milliseconds, required to receive a response from the web service for all web service calls within the most recent sample interval
Standard deviation [for Run]	The standard deviation time, in milliseconds, required to receive a response from the web service for all web service calls for the entire run
Maximum [for Interval]	The longest time, in milliseconds, required to receive a response from the web service for all web service calls within the most recent sample interval
Maximum [for Run]	The longest time, in milliseconds, required to receive a response from the web service for all web service calls for the entire run
Minimum [for Interval]	The shortest time, in milliseconds, required to receive a response from the web service for all web service calls within the most recent sample interval
Minimum [for Run]	The shortest time, in milliseconds, required to receive a response from the web service for all web service calls for the entire run

Counter name	Description
Average connection time [ms] [for Interval]	The average time, in milliseconds, required to receive a response from the web service for a specific web service call within the most recent sample interval
Average connection time [ms] [for Run]	The average time, in milliseconds, required to receive a response from the web service for a specific web service call for the entire run
Response time standard deviation [ms] [for Interval]	The standard deviation time, in milliseconds, required to receive a response from the web service for a specific web service call during the most recent sample interval
Response time standard deviation [ms] [for Run]	The standard deviation time, in milliseconds, required to receive a response from the web service for a specific web service call for the entire run
Maximum response time [ms] [for Interval]	The longest time, in milliseconds, required to receive a response from the web service for a specific web service call within the most recent sample interval
Maximum response time [ms] [for Run]	The longest time, in milliseconds, required to receive a response from the web service for a specific web service call for the entire run
Minimum response time [ms] [for Interval]	The shortest time, in milliseconds, required to receive a response from the web service for a specific web service call within the most recent sample interval

Counter name	Description
Minimum response time [ms] [for Run]	The shortest time, in milliseconds, required to receive a response from the web service for a specific web service call for the entire run

Counter name	Description
Count [for Interval]	The number of times web service calls have been invoked during the most recent sample interval
Count [for Run]	The number of times web service calls have been invoked during the entire run
Maximum [for Interval]	The maximum number of times web service calls have been invoked within an interval
Maximum [for Run]	The maximum number of times web service calls have been invoked during the entire run
Rate [per second] [for Interval]	The number of times per second that web service calls have been invoked during the most recent sample interval
Rate [per second] [for Run]	The number of times per second that web service calls have been invoked for the entire run

Counter name	Description
Maximum Rate Web Service Call Started [for Run]	The maximum number of times per second that a specific web service call has been invoked for the entire run
Maximum Web Service Call Started [for Interval]	The maximum number of times a specific web service call has been invoked during the most recent sample interval
Maximum Web Service Call Started [for Run]	The maximum number of times a specific web service call has been invoked for the entire run
Minimum Rate Web Service Call Started [for Run]	The minimum number of times per second that a specific web service call has been invoked for the entire run
Minimum Web Service Call Started [for Interval]	The minimum number of times a specific web service call has been invoked during the most recent sample interval
Minimum Web Service Call Started [for Run]	The minimum number of times a specific web service call has been invoked for the entire run
Rate Web Service Call Started [for Interval]	The number of times per second that a specific web service call has been invoked during the most recent sample interval
Rate Web Service Call Started [for Run]	The number of times per second that a specific web service call has been invoked for the entire run

Counter name	Description
Total Web Service Call Started [for Interval]	The number of times that a specific web service call has been invoked during the most recent sample interval
Total Web Service Call Started [for Run]	The number of times that a specific web service call has been invoked for the entire run

Counter name	Description
Count [for Interval]	The number of times web service calls have returned a valid response during the most recent sample interval
Count [for Run]	The number of times web service calls have returned a valid response during the entire run
Maximum [for Interval]	The maximum number of times web service calls have returned a valid response within an interval
Maximum [for Run]	The maximum number of times web service calls have returned a valid response during the entire run
Rate [per second] [for Interval]	The number of times per second that web service calls have returned a valid response during the most recent sample interval
Rate [per second] [for Run]	The number of times per second that web service calls have returned a valid response for the entire run

Counter name	Description
Maximum Rate Web Service Call Success [for Run]	The maximum number of times per second that a specific web service call has returned a valid response for the entire run
Maximum Web Service Call Success [for Interval]	The maximum number of times a specific web service call has returned a valid response during the most recent sample interval
Maximum Web Service Call Success [for Run]	The maximum number of times a specific web service call has returned a valid response for the entire run
Minimum Rate Web Service Call Success [for Run]	The minimum number of times per second that a specific web service call has returned a valid response for the entire run
Minimum Web Service Call Success [for Interval]	The minimum number of times a specific web service call has returned a valid response during the most recent sample interval
Minimum Web Service Call Success [for Run]	The minimum number of times a specific web service call has returned a valid response for the entire run
Rate Web Service Call Success [for Interval]	The number of times per second that a specific web service call has returned a valid response during the most recent sample interval

Counter name	Description
Rate Web Service Call Success [for Run]	The number of times per second that a specific web service call has returned a valid response for the entire run
Total Web Service Call Success [for Interval]	The number of times that a specific web service call has returned a valid response during the most recent sample interval
Total Web Service Call Success [for Run]	The number of times that a specific web service call has returned a valid response for the entire run

Counter name	Description
Count [for Interval]	The number of times web service calls have failed within the most recent sample interval
Count [for Run]	The number of times web service calls have failed during the entire run
Maximum [for Interval]	The maximum number of times web service calls have failed within an interval
Maximum [for Run]	The maximum number of times web service calls have failed during the entire run
Rate [per second] [for Interval]	The number of times per second that web service calls have failed during the most recent sample interval
Rate [per second] [for Run]	The number of times per second that web service calls have failed for the entire run

Counter name	Description
Maximum Rate Web Service Call Failures [for Run]	The maximum number of times per second that a specific web service call has failed for the entire run
Maximum Web Service Call Failures [for Interval]	The maximum number of times a specific web service call has failed during the most recent sample interval
Maximum Web Service Call Failures [for Run]	The maximum number of times a specific web service call has failed for the entire run
Minimum Rate Web Service Call Failures [for Run]	The minimum number of times per second that a specific web service call has failed for the entire run
Minimum Web Service Call Failures [for Interval]	The minimum number of times a specific web service call has failed during the most recent sample interval
Minimum Web Service Call Failures [for Run]	The minimum number of times a specific web service call has failed for the entire run
Rate Web Service Call Failures [for Interval]	The number of times per second that a specific web service call has failed during the most recent sample interval

Counter name	Description
Rate Web Service Call Failures [for Run]	The number of times per second that a specific web service call has failed for the entire run
Total Web Service Call Failures [for Interval]	The number of times that a specific web service call has failed during the most recent sample interval
Total Web Service Call Failures [for Run]	The number of times that a specific web service call has failed for the entire run

Counter name	Description
Count [for Interval]	The number of times web service calls produced a timeout during the most recent sample interval
Count [for Run]	The number of times web service calls produced a timeout during the entire run
Maximum [for Interval]	The maximum number of times web service calls produced a timeout within an interval
Maximum [for Run]	The maximum number of times web service calls produced a timeout during the entire run
Rate [per second] [for Interval]	The number of times per second that web service calls produced a timeout during the most recent sample interval
Rate [per second] [for Run]	The number of times per second that web service calls produced a timeout for the entire run

Counter name	Description
Maximum Rate Web Service Call Timeouts [for Run]	The maximum number of times per second that a specific web service call produces a timeout for the entire run
Maximum Web Service Call Timeouts [for Interval]	The maximum number of times a specific web service call produces a timeout during the most recent sample interval
Maximum Web Service Call Timeouts [for Run]	The maximum number of times a specific web service call produces a timeout for the entire run
Minimum Rate Web Service Call Timeouts [for Run]	The minimum number of times per second that a specific web service call produces a timeout for the entire run
Minimum Web Service Call Timeouts [for Interval]	The minimum number of times a specific web service call produces a timeout during the most recent sample interval
Minimum Web Service Call Timeouts [for Run]	The minimum number of times a specific web service call produces a timeout for the entire run

Counter name	Description
Rate Web Service Call Timeouts [for Interval]	The number of times per second that a specific web service call produces a timeout during the most recent sample interval
Rate Web Service Call Timeouts [for Run]	The number of times per second that a specific web service call produces a timeout for the entire run
Total Web Service Call Timeouts [for Interval]	The number of times that a specific web service call produces a timeout during the most recent sample interval
Total Web Service Call Timeouts [for Run]	The number of times that a specific web service call produces a timeout for the entire run



Service verification point counters

Service verification point counters provide information about verification points.

The verdict for a verification point can be `Pass`, `Fail`, `Error`, or `Inconclusive`.

- `Pass` indicates that the verification point matched or received the expected response. For example, a response code verification point is set to `Pass` when the recorded response code is received during the run. If your test does not contain verification points, it means that the connection succeeded.
- `Fail` indicates that the verification point did not match the expected response or that the expected response was not received.
- `Error` indicates that the primary request was not successfully sent to the server, no response was received from the server, or the response was incomplete or could not be parsed.
- `Inconclusive` is returned only if you provide custom code that defines an `Inconclusive` verdict.

Some counters produce an aggregate value, where the values for several web service message returns are combined into a single value; others produce values for each web service message return.

- *Aggregate* counters use the values for all the elements in a test to produce a single value for a report. This value is rolled up from all values that satisfy the counter. When you drag an aggregate counter onto a report, one value is displayed. Folders that contain aggregate counters have a clock superimposed on the folder icon: . In the tables that follow, the aggregate counters are listed after the titles that contain this icon.
- *Individual* counters produce values for each item that satisfies the counter, rather than an aggregate value. Folders that contain individual counters have an asterisk superimposed on the folder icon: . In the tables that follow, the individual counters are listed after the titles that contain this icon.

Some counters pertain to *intervals* in the run. You set the **Statistics sample interval** value in the schedule, as a schedule property.

Counter name	Description
Percent Pass	The percentage of all verification points that passed for a specific web service message return for the entire run

Counter name	Description
Count [for Interval]	The number of all verification points classified as <code>ERROR</code> for a specific web service message return during the most recent sample interval
Count [for Run]	The number of all verification points classified as <code>ERROR</code> for a specific web service message return for the entire run

Counter name	Description
Count [for Interval]	The number of all verification points that failed for a specific web service message return during the most recent sample interval
Count [for Run]	The number of all verification points that failed for a specific web service message return for the entire run

Counter name	Description
Count [for Interval]	The number of all verification points classified as <code>Inconclusive</code> for a specific web service message return during the most recent sample interval
Count [for Run]	The number of all verification points classified as <code>Inconclusive</code> for a specific web service message return for the entire run

Counter name	Description
Count [for Interval]	The number of all verification points that passed for a specific web service message return during the most recent sample interval
Count [for Run]	The number of all verification points that passed for a specific web service message return for the entire run

Counter name	Description
Percent All VPs Passed For Interval	The percentage of all verification points that passed during the most recent sample interval
Percent All VPs Passed For Run	The percentage of all verification points that passed for the entire run
Total All VPs Attempted [for Interval]	The number of all verification points executed within the most recent sample interval
Total All VPs Attempted [for Run]	The number of all verification points executed for the entire run
Total All VPs Error [for Interval]	The number of all verification points with a verdict of <code>ERROR</code> during the most recent sample interval
Total All VPs Error [for Run]	The number of all verification points with a verdict of <code>ERROR</code> for the entire run
Total All VPs Failed [for Interval]	The number of all verification points that failed within the most recent sample interval
Total All VPs Failed [for Run]	The number of all verification points that failed for the entire run
Total All VPs Inconclusive [for Interval]	The number of all verification points that were marked as <code>Inconclusive</code> during the most recent sample interval
Total All VPs Inconclusive [for Run]	The number of all verification points that were marked as <code>Inconclusive</code> for the entire run
Total All VPs Passed [for Interval]	The number of all verification points that passed within the most recent sample interval
Total All VPs Passed [for Run]	The number of all verification points that passed for the entire run

Counter name	Description
Percent Pass	The percentage of attachment verification points that passed for a specific web service message return for the entire run

Counter name	Description
Count [for Interval]	The number of attachment verification points classified as <code>ERROR</code> for a specific web service message return during the most recent sample interval

Counter name	Description
Count [for Run]	The number of attachment verification points classified as <code>Error</code> for a specific web service message return for the entire run

Counter name	Description
Count [for Interval]	The number of attachment verification points that failed for a specific web service message return during the most recent sample interval
Count [for Run]	The number of attachment verification points that failed for a specific web service message return for the entire run

Counter name	Description
Count [for Interval]	The number of attachment verification points classified as <code>Inconclusive</code> for a specific web service message return during the most recent sample interval
Count [for Run]	The number of attachment verification points classified as <code>Inconclusive</code> for a specific web service message return for the entire run

Counter name	Description
Count [for Interval]	The number of attachment verification points that passed for a specific web service message return during the most recent sample interval
Count [for Run]	The number of attachment verification points that passed for a specific web service message return for the entire run

Counter name	Description
Percent Attachment VPs Passed For Interval	The percentage of attachment verification points that passed during the most recent sample interval
Percent Attachment VPs Passed For Run	The percentage of attachment verification points that passed for the entire run
Total Attachment VPs Attempted [for Interval]	The number of attachment verification points executed during the most recent sample interval

Counter name	Description
Total Attachment VPs Attempted [for Run]	The number of attachment verification points executed for the entire run
Total Attachment VPs Error [for Interval]	The number of attachment verification points with a verdict of <code>ERROR</code> during the most recent sample interval
Total Attachment VPs Error [for Run]	The number of attachment verification points with a verdict of <code>ERROR</code> for the entire run
Total Attachment VPs Failed [for Interval]	The number of attachment verification points that failed during the most recent sample interval
Total Attachment VPs Failed [for Run]	The number of attachment verification points that failed for the entire run
Total Attachment VPs Inconclusive [for Interval]	The number of attachment verification points that were marked as <code>Inconclusive</code> during the most recent sample interval
Total Attachment VPs Inconclusive [for Run]	The number of attachment verification points that were marked as <code>Inconclusive</code> for the entire run
Total Attachment VPs Passed [for Interval]	The number of attachment verification points that passed during the most recent sample interval
Total Attachment VPs Passed [for Run]	The number of attachment verification points that passed for the entire run

Counter name	Description
Percent Pass	The percentage of contain verification points that passed for a specific web service message return for the entire run

Counter name	Description
Count [for Interval]	The number of contain verification points classified as <code>ERROR</code> for a specific web service message return during the most recent sample interval
Count [for Run]	The number of contain verification points classified as <code>ERROR</code> for a specific web service message return for the entire run

Counter name	Description
Count [for Interval]	The number of contain verification points that failed for a specific web service message return during the most recent sample interval
Count [for Run]	The number of contain verification points that failed for a specific web service message return for the entire run

Counter name	Description
Count [for Interval]	The number of contain verification points classified as <i>Inconclusive</i> for a specific web service message return during the most recent sample interval
Count [for Run]	The number of contain verification points classified as <i>Inconclusive</i> for a specific web service message return for the entire run

Counter name	Description
Count [for Interval]	The number of contain verification points that passed for a specific web service message return during the most recent sample interval
Count [for Run]	The number of contain verification points that passed for a specific web service message return for the entire run

Counter name	Description
Percent Contain VPs Passed For Interval	The percentage of contain verification points that passed during the most recent sample interval
Percent Contain VPs Passed For Run	The percentage of contain verification points that passed for the entire run
Total Contain VPs Attempted [for Interval]	The number of contain verification points executed within the most recent sample interval
Total Contain VPs Attempted [for Run]	The number of contain verification points executed for the entire run
Total Contain VPs Error [for Interval]	The number of contain verification points with a verdict of <i>Error</i> during the most recent sample interval

Counter name	Description
Total Contain VPs Error [for Run]	The number of contain verification points with a verdict of <code>Error</code> for the entire run
Total Contain VPs Failed [for Interval]	The number of contain verification points that failed during the most recent sample interval
Total Contain VPs Failed [for Run]	The number of contain verification points that failed for the entire run
Total Contain VPs Inconclusive [for Interval]	The number of contain verification points that were marked as <code>Inconclusive</code> during the most recent sample interval
Total Contain VPs Inconclusive [for Run]	The number of contain verification points that were marked as <code>Inconclusive</code> for the entire run
Total Contain VPs Passed [for Interval]	The number of contain verification points that passed during the most recent sample interval
Total Contain VPs Passed [for Run]	The number of contain verification points that passed for the entire run

Counter name	Description
Percent Pass	The percentage of equal verification points that passed for a specific web service message return for the entire run

Counter name	Description
Count [for Interval]	The number of equal verification points classified as <code>Error</code> for a specific web service message return during the most recent sample interval
Count [for Run]	The number of equal verification points classified as <code>Error</code> for a specific web service message return for the entire run

Counter name	Description
Count [for Interval]	The number of equal verification points that failed for a specific web service message return during the most recent sample interval
Count [for Run]	The number of equal verification points that failed for a specific web service message return for the entire run

Counter name	Description
Count [for Interval]	The number of equal verification points classified as <code>Inconclusive</code> for a specific web service message return during the most recent sample interval
Count [for Run]	The number of equal verification points classified as <code>Inconclusive</code> for a specific web service message return for the entire run

Counter name	Description
Count [for Interval]	The number of equal verification points that passed for a specific web service message return during the most recent sample interval
Count [for Run]	The number of equal verification points that passed for a specific web service message return for the entire run

Counter name	Description
Percent Equal VPs Passed For Interval	The percentage of equal verification points that passed during the most recent sample interval
Percent Equal VPs Passed For Run	The percentage of equal verification points that passed for the entire run
Total Equal VPs Attempted [for Interval]	The number of equal verification points executed within the most recent sample interval
Total Equal VPs Attempted [for Run]	The number of equal verification points executed for the entire run
Total Equal VPs Error [for Interval]	The number of equal verification points with a verdict of <code>Error</code> during the most recent sample interval
Total Equal VPs Error [for Run]	The number of equal verification points with a verdict of <code>Error</code> for the entire run
Total Equal VPs Failed [for Interval]	The number of equal verification points that failed during the most recent sample interval
Total Equal VPs Failed [for Run]	The number of equal verification points that failed for the entire run
Total Equal VPs Inconclusive [for Interval]	The number of equal verification points that were marked as <code>Inconclusive</code> during the most recent sample interval

Counter name	Description
Total Equal VPs Inconclusive [for Run]	The number of equal verification points that were marked as <code>Inconclusive</code> for the entire run
Total Equal VPs Passed [for Interval]	The number of equal verification points that passed during the most recent sample interval
Total Equal VPs Passed [for Run]	The number of equal verification points that passed for the entire run

Counter name	Description
Percent Pass	The percentage of query verification points that passed for a specific web service message return for the entire run

Counter name	Description
Count [for Interval]	The number of query verification points classified as <code>Error</code> for a specific web service message return during the most recent sample interval
Count [for Run]	The number of query verification points classified as <code>Error</code> for a specific web service message return for the entire run

Counter name	Description
Count [for Interval]	The number of query verification points that failed for a specific web service message return during the most recent sample interval
Count [for Run]	The number of query verification points that failed for a specific web service message return for the entire run

Counter name	Description
Count [for Interval]	The number of query verification points classified as <code>Inconclusive</code> for a specific web service message return during the most recent sample interval
Count [for Run]	The number of query verification points classified as <code>Inconclusive</code> for a specific web service message return for the entire run

Counter name	Description
Count [for Interval]	The number of query verification points that passed for a specific web service message return during the most recent sample interval
Count [for Run]	The number of query verification points that passed for a specific web service message return for the entire run


Counter name	Description
Percent Query VPs Passed For Interval	The percentage of query verification points that passed during the most recent sample interval
Percent Query VPs Passed For Run	The percentage of query verification points that passed for the entire run
Total Query VPs Attempted [for Interval]	The number of query verification points executed within the most recent sample interval
Total Query VPs Attempted [for Run]	The number of query verification points executed for the entire run
Total Query VPs Error [for Interval]	The number of query verification points with a verdict of <code>ERROR</code> during the most recent sample interval
Total Query VPs Error [for Run]	The number of query verification points with a verdict of <code>ERROR</code> for the entire run
Total Query VPs Failed [for Interval]	The number of query verification points that failed during the most recent sample interval
Total Query VPs Failed [for Run]	The number of query verification points that failed for the entire run
Total Query VPs Inconclusive [for Interval]	The number of query verification points that were marked as <code>Inconclusive</code> during the most recent sample interval
Total Query VPs Inconclusive [for Run]	The number of query verification points that were marked as <code>Inconclusive</code> for the entire run
Total Query VPs Passed [for Interval]	The number of query verification points that passed during the most recent sample interval
Total Query VPs Passed [for Run]	The number of query verification points that passed for the entire run


Socket counters

With socket generic counters, which are displayed in the Performance Test Runs view, you can customize your socket reports with dynamic information that is updated for each run.

Socket counters

These counters provide information about connections started, send and receive actions completed, and connection times. Some counters produce an aggregate value, where the values for several send and receive actions are represented in one value; others produce values for each action.

Aggregate counters use the values for all the connections in a test to produce a single value for a report. This value is calculated from all values that satisfy the counter. When you drag an aggregate counter onto a report, one value is displayed. Folders that contain aggregate counters have a clock superimposed on the folder icon: . The following tables that list aggregate counters have this icon in their title.

Individual counters produce values for each connection that satisfies the counter, rather than a single aggregate value. Folders that contain individual counters have an asterisk superimposed on the folder icon: . The following tables that list individual counters have this icon in their title.

Some counters pertain to *intervals* in the run. You set the **Statistics sample interval** value in the schedule as a schedule property.

Bytes Received counters

The counters in this section provide information about the number of bytes that were received.

Counter name	Description
Count [for Interval]	The number of bytes received for a specific connection during the most recent sample interval
Count [for Run]	The number of bytes received for a specific connection for the entire run

Bytes Sent counters

The counters in this section provide information about the number of bytes that were sent.

Counter name	Description
Count [for Interval]	The number of bytes sent for a specific connection within the most recent sample interval
Count [for Run]	The number of bytes sent for a specific connection for the entire run

Connect Attempts counters

The counters in this section provide information about the number of attempts to establish a connection with the server.

Counter name	Description
Count [for Interval]	The number of attempts to establish a connection within the most recent sample interval
Count [for Run]	The number of attempts to establish a connection for the entire run

Connect Times counters

The counters in this section provide information about the connection response time. This is the time, in milliseconds, that elapses from the point that the test client attempts to establish a connection with the server and the moment the connection is established.

Counter name	Description
Average [for Interval]	The average connection time for a specific connection during the most recent sample interval
Average [for Run]	The average connection time for a specific connection for the entire run
Maximum [for Interval]	The maximum connection time for a specific connection during the most recent sample interval
Maximum [for Run]	The maximum connection time for a specific connection for the entire run
Minimum [for Interval]	The minimum connection time for a specific connection during the most recent sample interval
Minimum [for Run]	The minimum connection time for a specific connection for the entire run
Standard Deviation [for Interval]	The standard deviation of connection times for a specific connection during the most recent sample interval
Standard Deviation [for Run]	The standard deviation of connection times for a specific connection for the entire run

Counter name	Description
Average [ms] [for Interval]	The average connection time for all connections within the most recent sample interval
Average [ms] [for Run]	The average connection time for all connections for the entire run

Counter name	Description
Maximum [ms] [for Interval]	The maximum connection time for all connections within the most recent sample interval
Maximum [ms] [for Run]	The maximum connection time for all connections for the entire run
Minimum [ms] [for Interval]	The minimum connection time for a all connections within the most recent sample interval
Minimum [ms] [for Run]	The minimum connection time for a all connections for the entire run
Standard Deviation [ms] [for Interval]	The standard deviation of connection times for all connections during the most recent sample interval
Standard Deviation [ms] [for Run]	The standard deviation of connection times for all connections for the entire run

Connects counters

The counters in this section provide information about the number of connections that were established.

Counter name	Description
Count [for Interval]	The number of established connections for a specific connection during the most recent sample interval
Count [for Run]	The number of established connections for a specific connection for the entire run

Receive Attempts counters

The counters in this section provide information about the number of attempts to receive data from the connection.

Counter name	Description
Count [for Interval]	The number of receive attempts for a specific connection during the most recent sample interval
Count [for Run]	The number of receive attempts for a specific connection for the entire run

Receives counters

The counters in this section provide information about the number of receive actions that were successfully completed that were received.

Counter name	Description
Count [for Interval]	The number of completed receives for a specific connection during the most recent sample interval
Count [for Run]	The number of completed receives for a specific connection for the entire run

Send Attempts counters

The counters in this section provide information about the number of attempts to send data from the connection.

Counter name	Description
Count [for Interval]	The number of send attempts for a specific connection during the most recent sample interval
Count [for Run]	The number of send attempts for a specific connection for the entire run

Sends counters

The counters in this section provide information about the number of send actions that were successfully completed.

Counter name	Description
Count [for Interval]	The number of completed sends for a specific connection during the most recent sample interval
Count [for Run]	The number of completed sends for a specific connection for the entire run

Chapter 9. Troubleshooting Guide

This guide describes how to analyze and resolve some of the common problems that you might encounter while you work with Rational® Service Tester for SOA Quality.

Troubleshooting performance testing

This topic provides information about how to troubleshoot several problems with Rational® Service Tester for SOA Quality.

If you run tests and encounter problems, make sure that you have followed all the Performance testing tips.

If an error message is displayed when you run tests, try looking up the error message in the *Performance testing error messages* section of the online help. Only the most common error messages are listed. If no error message is displayed when you encounter a problem, open the error log by clicking **Window > Show View > Error Log**. If the workbench shuts down while running tests, restart the workbench and examine the error log. By default, warning and error messages are logged. You can increase the default logging level by clicking **Window > Preferences > Logging**. The log file is stored in the `.metadata` directory of your workspace. To avoid excessive logging, the Logging Level should be adjusted for individual Logger Names in the Loggers tab. For example, to get more information about a problem connecting with IBM® Engineering Test Management, increase the Logging Level for `com.ibm.rational.test.lt.rqm.adapter` Logger Name. For the licensing issue, adjust the level for `com.ibm.rational.test.lt.licening` Logger Name. When you no longer need the extra logging, use the **Restore Default** button in the Logging Preferences to reset all the levels to their recommended defaults.

In addition to the online help, you can find workarounds or solutions to problems in the [IBM Support](#) portal.

You might encounter some of these problems while performance testing:

Connectivity problems between workbench and agent computers

If the workbench stops or locks up when you attempt to start running tests, it is important to confirm that all the agent computers are running. Perform the following steps to confirm your installation is properly configured:


- Confirm that there is sufficient disk space available on the workbench computer and the agent computers.
- Restart the workbench computer.
- Verify the network connectivity between the workbench computer and agent computers. To confirm the hostname in `majordomo.config` file can be DNS resolved on the agent machine, use a shell ping to the workbench hostname. If the ping results fail use the IP address of the workbench instead.
- Confirm the server port number on the test workbench computer. Click **Window > Preferences > Server**. This is the port number that should be specified in `majordomo.config` file on the agent machines.
- Restart the agent computers and verify the Majordomo process is running.

- On the agent machines, set the optional debug flag in the majordomo.config file. Set the value equal to true; the default value is false. You do not have to restart the agent. Within about ten seconds it should automatically pick up the changes to majordomo.config.

Look in %TEMP% directory for the majordomo.log file. This file contains information about the attempts to contact the workbench including information about any failures and the reason for the failures.

On the Windows operating system, the %TEMP% directory is typically at %USERPROFILE%\AppData\Local\Temp.

If the majordomo service is configured to log in as Local System Account, then the %TEMP% directory is at %SystemRoot%\TEMP, typically C:\Windows\TEMP.

- You can check the agent status on the workbench computer by clicking the  icon. For the Agent Controller, you can attempt to share files between the workbench computer and agent computers. Click **Window > Preferences > Agent Controller > Hosts**, and then add the agent computers as hosts, and click **Test Connection** to test connectivity to the instances of the Agent Controller that are running on the agent computers.

Recording configuration problems

No HTTP traffic is captured while recording

See Recording reliable HTTP tests for instructions on configuring your web browser. If you are attempting to use Internet Explorer to record tests from a secure website, see Configuring Microsoft Edge for recording on a secure website. Disable firewalls on the workbench computer and the agent computers.

No traffic is captured while recording

Ensure that the recorder type that you select matches the protocol in use by the system under test. For example, do not attempt to use the HTTP recorder if the system under test uses the Citrix protocol.

No test is generated after recording

When the test generator cannot create a test from the recorded traffic, typically an error message is displayed or written to the error log. Try looking up the error message in the *Performance testing error messages* section of the online help. Error messages might also be documented in technotes in the Support Knowledge Base at <http://www.ibm.com/software/awdtools/tester/performance/support/>.

Recorder controls are not available

If you use a workspace from a different version of the product, the recorder controls might not be available. Instead, the recorder controls from the other version of the product are displayed. Click **Window > Reset Perspective** to reset the **Performance Test** or **Service Test** perspective. Alternately, click **File > New > Other** to select the wizard to use.

Problems running large tests or long-run tests

If a test runs but ends with errors, check that the workbench computer and agent computers meet the hardware and software requirements that are detailed in the installation guide. Pay close attention to the memory and disk space requirements. See [Increasing memory allocation on page 401](#) for more information on how to set the maximum heap size to avoid out-of-memory errors. Monitor processor and memory usage on the workbench and agent computers and watch for excessive processor use or excessive memory use by `javaw.exe` or `java.exe` processes. If error messages pertain to processes stopping unexpectedly, see this support article: <http://www.ibm.com/support/docview.wss?uid=swg21395486>.

Run tests with fewer virtual users that use the default schedule settings to determine whether the behavior is linked to the number of users. Examine the test log for error messages that the system under test generates. Run tests with a single virtual user and make sure that the system under test is not generating errors, before you attempt to run tests with a large number of users. If you encounter problems, restart the workbench and agent computers before attempting to run tests again.

If the workbench shuts down while running tests, search for file names that begin with `javacore`. The name of `javacore` files includes the date, time, and process ID. If you find a `javacore` file with a date, time, and process ID matching the workbench, open the file in a text editor. You can find the reason for failure at the beginning of the `javacore` file.

Data correlation errors

If you can record tests successfully, but the expected behavior is not triggered in your application when you run tests, you might need to perform manual data correlation. Typically when additional data correlation is needed, the test log includes messages similar to this message: `Unable to extract the value`. To troubleshoot data correlation problems, try running tests using only one virtual user running on the workbench computer, and compare the playback to the recorded test to determine which responses from the system under test are unexpected. See [Debugging HTTP tests](#) to learn how to use the test log and the **Protocol Data** view to troubleshoot HTTP tests. To learn more about data correlation, see [Correlating response and request data on page 315](#).

Common errors integrating with IBM® Engineering Test Management

All modes of the adapter use the Eclipse error log. You can view the log by opening the workbench and clicking **Window > Show View > Error Log**. By default, warning and error messages are logged. You can turn on more detailed logging for the adapter by clicking **Window > Preferences > Logging**. The log component for the adapter is named `com.ibm.rational.test.lt.rqm.adapter`.

If you are running the adapter as a Windows™ service or from the command line, you can view the `adapter.log` file without opening the test workbench.

Problem	Solution or cause
Where do you look for errors or warnings?	In the workbench, click Window > Show View > Error Log .

Problem	Solution or cause
You do not see the adapter available for selection.	<ul style="list-style-type: none"> • Verify that the Engineering Test Management server address that is provided to the adapter is correct. Provide the correct address. • Check the provided login and password. Provide the correct password.
The adapter continuously fails to connect to Engineering Test Management.	Make sure that the server is running. If necessary, restart the server or check network connectivity.
The adapter is displayed as red in the selection dialog.	<ul style="list-style-type: none"> • The adapter is not communicating with the server. • The adapter might already be in use.
You attempt to import a script from the adapter but no scripts are found.	<ul style="list-style-type: none"> • Make sure the project path that is entered in Engineering Test Management is a project under the workspace that is associated with the running adapter. You have to enter only the project name. This is less error prone than typing the complete project path, but either forms are acceptable. • If running from the command line or as a service, be certain the <code>WORKSPACE_DIR</code> environment variable that is set in the <code>adapter.config</code> file is the same path as seen in the select workspace dialog when running the test workbench. Be careful not to set the path to a project folder under the workspace directory. • Make sure that you are not using a workspace that contains a project that was copied from a shared location. A workspace that contains projects from shared locations cannot be used for projects that are not shared.
The adapter is running from the command line or as a service, and tests continue to fail.	Run the adapter in GUI mode so that you can see what happens when the test workbench runs the test script.
Adapter Windows™ services does not start. A error message states that the service failed to start in a timely fashion.	Ensure that the computer has .NET 2.0 or later. This platform can be installed from the Windows™ Update Site or manually. For more information on installing .NET, see http://support.microsoft.com/kb/923100 .
When testing shared assets, the execution fails with and an <code>IOException</code> message is displayed.	The most likely cause is that the Engineering Test Management to UNC shared location is not set up correctly.

Problem	Solution or cause
When testing shared assets, the execution fails with a low level model error.	<ul style="list-style-type: none"> • From Engineering Test Management, ensure that you can access the UNC shared directory without being prompted for a password. You might have to map a drive on Windows™ for the Engineering Test Management system to log into the UNC share. • Ensure that you have defined the shared resource in Engineering Test Management under Admin > System Properties > Resources. • Ensure that the test-script points to a shared location that still exists. If you have associated a Engineering Test Management test script with a shared location that has changed (for example if the IP address has been reassigned) you might need to reassociate every test script • Ensure that the UNC shared directory that is specified in Engineering Test Management points to a project.
Service tests that were created in a previous version of the product cannot be run.	Upgrade every SOA asset to the latest version.
The adapter cannot connect to the server, and one of the following error messages is displayed:	<ul style="list-style-type: none"> • When using Engineering Test Management 3.0 or later, the server URL that is configured for the adapter must exactly match the public URI of the Engineering Test Management server. The server public URI is available on the Engineering Test Management administration page. By default the administration page is at <code>https://server-name:9443/qm/admin</code>. • The adapter user must be a member of the Engineering Test Management project area. Open the project area administration page on the Engineering Test Management server to determine whether the adapter user is a member of the project area. For Engineering Test Management 3.0 and later, the adapter user must be a member in the test team member role, not the test team contributor role. This error can also occur if you have modified these roles from their defaults.

- `Communications error with server`
- `Error occurred while registering the adapter`

Performance testing error messages

Find more information about the error messages.

PRXE0101W

%1
terminating
due
to
exception:
%2

PRXE4943W

Transaction
[%1]
has
been
aborted.

PRXE4951I

User
group
[%1]
was
not
found.

RMSE0003W

RMSE0003W
There
are
currently
no
selected
counters
for
the
source
named
{0}.

Explanation: The source has no counters selected.

System action: Execution of the schedule will continue but the information related to this source won't be collected.

User response: Consider selecting at least one counter from the Resource Monitoring Service web console.

RMSE0004W

RMSE0004W

The
source
named
{0}
is
no
longer
available.

Explanation: This source has been removed from the Service web console after it was added to this schedule.

System action: Execution of the schedule will continue but the information related to this source won't be collected.

User response: Consider adding it back, then edit the schedule to update the sources to be monitored during its execution.

RMSE0005W

RMSE0005W

The
source
named
{0}
is
reporting
the
error
message
{1}.

Explanation: Look at the reported error.

System action: Execution of the schedule will continue but the information related to this source won't be collected.

User response: Consider fixing it from the Resource Monitoring Service web console.

RMSE0006W

The
server
does
not
support
resource
monitoring
labels.

Explanation: The server does not support resource monitoring labels.

System action: Execution of the schedule will continue but the resource monitoring counters won't be collected.

User response: Consider using a service that supports this feature.

RPAC0001W

The
JAR
%1
referenced
in
preferences
could
not
be
found.
Preferences
on
the
cloud
workbench
will
be
cleared.

Explanation: The Resource Monitoring preferences list a JAR file that is required for an instrumented application server type. This JAR file must be mapped to a new location and transferred to the cloud workbench. But this transaction failed, because the file could not be found locally.

System action: Execution in the cloud will continue but the instrumented application server types that require the listed JAR file might fail.

User response: Open the child preference page under Test -> Performance Resource Monitoring. Ensure that the listed files exist and can be found in a valid location.

RPHD1032E

Error
occurred
while
instructing
Rational®
Performance
Tester
engine
to
enable
real-
time
protocol
data
for
user:
%1.
It's
possible
that
no
data
will
be
seen
for
this
user
in
the
Protocol
Data
view.

Explanation: There was a general error when starting real-time browsing in the Protocol Data View.

System action: The Protocol Data View will not be updated in real-time during this run. This does not affect test execution or post-run usage of the view.

User response: Ensure there is a stable connection with the Performance Test Agent and System Under Test. If problem persists, contact support.

RPHD1034E

Error
occurred
while
instructing
Rational@
Performance
Tester
engine
to
disable
real-
time
protocol
data
for
user:
%1.
It's
possible
that
data
for
this
user
will
continue
to
be
displayed
in
the
Protocol
Data
view.

Explanation: = There was a general error when ending real-time browsing in the Protocol Data View.

System action: None.

User response: If the Protocol Data View no longer updates for additional runs or when the test editor selection is changed, closing the view and reopening it may help.

RPHE0001E

example
of
translatable
error
message
%1

RPHE0116E

A
cisterna
logging
error
was
encountered
for
request
%1.
The
exception
is
%2.

Explanation: An error occurred trying to log large data attachments. You will not be able to see this data when this happens.

System action: The data was not logged.

User response: Contact customer support if you need to see the data that was not logged

RPHE0010W

Unknown
authentication
scheme
'%1'
discovered
in
HTTP
401
response,
ignoring.

RPHE0011W

Unrecognized
authentication
header
'%1'
discovered
in
HTTP
401
response,
ignoring.

RPHE0012W

No
authentication
headers
found
in
HTTP
401
response,
ignoring.

RPHE0013W

The
server
requested
NTLM
authentication
but
no
NTLM
authentication
context
was
supplied
with
this
request.
Authentication
is
not
possible.

RPHE0014W

NTLM
authentication
failed
for
this
request.
Verify
that
the
NTLM
authentication
context
values
for
this
request
are
correct.

RPHE0100W

Host
name
'%1'
can
not
be
resolved.

Explanation: A connection could not be established with the host. This can occur if the testing environment changes so that the host name is no longer correct. This can also occur when running a test on a different computer, such as an agent computer, from the workbench computer that was used for recording, if the new computer cannot resolve the host name.

User response: If the host name is incorrect due to a change in the testing environment, update the host name in the test. Otherwise, try to resolve the host name using the command `nslookup <hostname>`. Run `nslookup` on the agent computer if the error is happening on the agent computer. If `nslookup` is also unable to resolve the name, contact your network administrator. If `nslookup` resolves the host name, but the test continues to fail, try changing the host name to a fully-qualified host name. Alternatively, edit the hosts file.

RPHE0101W

Encountered
error
while
updating
dynamic
cookie
cache
while
interpreting
'Set-
Cookie'
header
with
value
'%1'
sent
from
web-
server
'%2'
retrieving
URI
'%3'.
Explanation
message:
'%4'.
Cache
not
updated
to
include
this
cookie
value.

RPHE0102W

Unexpected
challenge(HTTP
status
code=401)
received
during
HTTP
playback
to
web-
server
'%1'
retrieving
URI
'%2'.
This
behavior
differs
from
the
behavior
recorded
during
test
creation.
For
authentication
to
playback
correctly
a
challenge
must
be
recorded
during
test
creation.

RPHE0103W

Authentication
failed
during
HTTP
playback
to
web-
server
'%1'
retrieving
URI
'%2'.
Probable
cause:
username
'%3'
and/
or
password
'%4'
incorrect.

RPHE0104W

Exception
occurred
during
attempt
to
write
request
to
web-
server
'%1'
getting
url
'%2'.
Explanation:
%3

RPHE0105W

General
un-
handled
exception
occurred
during
socket
I/O
read
from
web-
server
'%1'
retrieving
URI
'%2'.
Explanation
message:
'%3'.

Explanation: This error occurs when the server abruptly closes the connection to the virtual user. Servers might close connections if the virtual user is detected as a security risk due to an invalid cookie, failed SSL negotiation, or an improperly formatted request.

User response: Compare the request that was sent at run time (in the test log) to the one that is in the test. To determine if differences between the requests are valid, record the test again and compare the two requests.

RPHE0106W

A
read
time-
out
occurred
during
a
socket
I/O
read
from
web-
server
'%1'
retrieving
URI
'%2'.
Since
this
URI
is
the
primary
request
for
the
current
page
all
secondary
requests
will
be
skipped
and
the
next
page
will
be
attempted.
Current
time-
out
value
of

Explanation: The server did not return the response data before the timeout interval elapsed. If the server is under heavy load, the behavior can be caused by bottlenecks on the server or the agent computers. This error can also occur if an incorrect request is sent and the server is unable to respond.

User response: If the server is under heavy load, examine the server and agent computers to find and fix bottlenecks. Increase the timeout value. To stop tests or virtual users when this error occurs, enable error handling in the test and configure the server timeout error condition. If the server is not under heavy load, examine the request to ensure that it is valid and accurate.

RPHE0107W

A
read
time-
out
occurred
during
a
socket
I/O
read
from
web-
server
'%1'
retrieving
URI
'%2'.
This
secondary
request
will
be
skipped.
Current
time-
out
value
of
'%3'
milliseconds
should
be
increased
if
long
delays
are
expected
on
this
request.

Explanation: The server did not return the response data before the timeout interval elapsed. If the server is under heavy load, the behavior can be caused by bottlenecks on the server or the agent computers. This error can also occur if an incorrect request is sent and the server is unable to respond.

User response: If the server is under heavy load, examine the server and agent computers to find and fix bottlenecks. Increase the timeout value. To stop tests or virtual users when this error occurs, enable error handling in the test and configure the server timeout error condition. If the server is not under heavy load, examine the request to ensure that it is valid and accurate.

RPHE0108W

A
connect
time-
out
occurred
during
a
socket
I/O
connect
to
web-
server
'%1'
attempting
to
retrieve
URI
'%2'.
Since
this
URI
is
the
primary
request
for
the
current
page
all
secondary
requests
will
be
skipped
and
the
next
page
will
be
attempted.

Explanation: This error can occur if the server or agent computer is under heavy load. This error can also occur if the server or host computer is not configured with enough connections, or if the agent computer is not configured with enough sockets.

User response: Examine the server and agent computers to find and fix bottlenecks. To stop tests or virtual users when this error occurs, enable and configure error handling in the test.

RPHE0109W

A
connect
time-
out
occurred
during
a
socket
I/O
connect
to
web-
server
'%1'
attempting
to
retrieve
URI
'%2'.
This
secondary
request
will
be
skipped.

Explanation: This error can occur if the server or agent computer is under heavy load. This error can also occur if the server or host computer is not configured with enough connections, or if the agent computer is not configured with enough sockets.

User response: Examine the server and agent computers to find and fix bottlenecks. To stop tests or virtual users when this error occurs, enable and configure error handling in the test.

RPHE0110W

Unexpected challenge(HTTP status code=407) received while accessing HTTP proxy '%1' retrieving URI '%2'. This behavior differs from the behavior recorded during test creation. For authentication to playback correctly a challenge must be recorded during test creation.

Explanation: When the test was recorded, no basic authentication was required on the proxy server. When the test is run, the proxy server is requesting basic authentication information that is not in the test.

User response: Record the test again to capture basic authentication information. Play back the new test, or add the basic authentication information to the request in the original test.

RPHE0111W

Authentication
failed
accessing
proxy-
server
'%1'
retrieving
URI
'%2'.
Probable
cause:
username
'%3'
and/
or
password
'%4'
incorrect.

Explanation: Basic authentication failed when connecting to the proxy server. This can occur if an incorrect user name or password is supplied.

User response: Ensure that user name and password are correct.

RPHE0112W

An
error
occurred
during
decoding
of
content
received
from
web-
server
'%1'
attempting
to
retrieve
URI
'%2'.
Explanation
message:
'%3'.

RPHE0113E

Error
encountered
during
the
process
of
URI
substitution
for
host=
%1
and
URI
=
%2 .
Data
correlation
supplied
a
malformed
URI=
%3 .
Explanation:
%4.
If
you
attempted
to
perform
a
custom
data
substitution
on
this
URI
ensure
it
has
proper
URI
syntax.

If
you
did
not

RPHE0113W

An
error
occurred
during
encoding
of
an
annotated
execution
history
event
property.
Explanation
message:
'%1'.

RPHE0114E

An
error
was
encountered
during
transform
of
response
data.
%1

Explanation: The response data was not in a format that the data transformer could interpret. This can occur when an error is returned from the server instead of valid response data.

User response: Examine the response data for errors.

RPHE0114W

Exception
occurred
during
attempt
to
write
request
to
proxy-
server
'%1'
getting
URL
'%2'
on
host
'%3'.
Explanation:
%4.

RPHE0115E

An
error
was
encountered
during
un-
transformation
of
request
data.
%1

Explanation: The transformed request data could not be converted into the format required by the server. This can occur because of a faulty data substitution. This can also occur if you manually edit the request data and invalidate the transformed data format.

User response: Correct the faulty substitution or the invalid data formatting.

RPHE0115W

Unable
to
successfully
establish
a
connection
to
web-
server
'%1'
retrieving
URI
'%2'.
Web-
server
closing
the
connection
after
connection
was
just
established.

RPHE0117W

Unexpected
exception
occurred
during
connection
close
to
web-
server
'%1'
retrieving
URI
'%2'.
Explanation:
%3.

RPHE0118W

HTTP
parsing
error
encountered
while
retrieving
URI
'%1'
from
web-
server
'%2'.
If
this
URI
is
the
primary
request
for
the
current
page
all
secondary
requests
will
be
skipped
and
the
next
page
will
be
attempted.

RPHE0119E

IP
aliasing
is
enabled
but
no
IP
address
was
found
for
virtual
user
%1.
Verify
correct
network
interface
name(s)
are
specified.

RPHE0120E

Exception
occurred
during
attempt
to
connect
to
proxy-
server
'%1'
getting
URL
'%2'
on
host
'%3'.
Explanation:
%4.

RPHE0121E

Unable
to
authenticate
with
the
proxy-
server.
Possible
solution:
re-
record
test
due
to
possible
proxy-
server
'%1'
authentication
changes.

RPHE0122W

Web-server
'%1'
unexpectedly
closed
the
connection
while
in
the
process
of
retrieving
URI
'%2'.
The
response
body
MAY
be
incomplete
due
to
a
missing
"chunk".
If
missing
chunk
was
last
(zero
length)
chunk,
data
is
complete.

RPHE0123W

Infinite
redirection
loop
detected
getting
URL
'%1'.
If
this
is
expected
and
understood
increase
RPT_VMARGS
rptMaxRedirection
parameter.
Redirected
history
%2

RPHE0124W

Unexpected
server
redirection
occurred
getting
URL
'%1'.
We
were
redirected
to
the
same
URI
which
issued
this
request.
Redirected
history
%2

RPHE0125W

An
error
occurred
during
the
log
of
the
request
content.
Explanation
message:
'%1'.

Explanation: Something went wrong while logging an HTTP payload with a content-type that might not be checkmark_perf.jpg by the underlying log system (cisterna). Such an attachment will simply not be logged with the other events and won't be available in the Test Log explorer.

System action: No system action.

User response: In case the content type is a commonly used one, consider creating a feature request to get this content-type checkmark_perf.jpg by the product.

RPIB0007E

%1

RPIH0014E

Create
the
AllowSimulationAPI
'REG_DWORD'
key
with
value
1
in
HKLM\>
Software\>
[WOW6432Node]\>
Citrix\>
ICA
Client\>
CCM
folder
of
the
windows
registry.

Explanation: Recording of RPT Citrix activity requires a AllowSimulationAPI key in HKLM\> Software\> [WOW6432Node]\> Citrix\> ICA Client\> CCM. This specification is documented by Citrix in article CTX123616.

System action: Users either need to create the AllowSimulationAPI 'REG_DWORD' key with value 1 manually or need to provide appropriate write permissions to 'users' group for this folder, with appropriate permissions in place RPT workbench can create these keys before a recording.

User response: Users either need to create the AllowSimulationAPI 'REG_DWORD' key with value 1 manually or need to provide appropriate write permissions to 'users' group for this folder, with appropriate permissions in place RPT workbench can create these keys before a recording.

RPKG0090E

Exception
thrown
while
creating
connection
variables

Explanation: Exception thrown while creating connection variables

System action: Can not create the connection variable

User response: None required

RPKG0100E

Exception
thrown
by
the
launch
configuration
core

Explanation: Exception thrown by the launch configuration

System action: None required

User response: None required

RPKG0101E

Exception
thrown
during
an
update
to
a
launch
configuration

Explanation: Exception thrown during an update to a launch configuration

System action: None required

User response: None required

RPKG0110E

The
data
source
type
%1
is
not
expected

Explanation: The data source type %1 is not expected

System action: None required

User response: None required

RPSE0014W

SAP
Calendar
dialog
could
have
unpredictable
behavior
during
playback,
set
the
date
directly
in
the
field
using
string
format.

Explanation: SAP Scripting Calendar object is not safe, in hide mode replay could fail.

System action: No specific action during test generation. Recorded actions are kept.

User response: Date should be set in corresponding field as String value, ex: 10.25.2021

RPSF0114E

SAP
GUI
Application
creation
failed

Explanation: SAP GUI is not installed with recommended scripting options.

System action: Recording is stopped.

User response: Install SAP GUI with scripting options as recommended by SAP.

RPSF0172E

Rational®
Performance
Tester/SAP:
Unable
to
start
SAP
GUI,
please
check
SAP
GUI
installation.

Explanation: SAP GUI can't be reached.

System action: The test is stopped.

User response: Install SAP GUI with scripting options as recommended by SAP.

RPSF0195E

Connection
with
SAP
GUI
existing
session
or
shortcut
not
allowed
during
schedule
execution.

Explanation: Connection on existing SAP GUI session or shortcut are impossible in a performance schedule execution, these are reserved for test or compound test.

System action: The test is stopped.

User response: Connection string or SAP logon must be used for schedule mode.

RPSF0197E

No
matching
SAP
GUI
Scripting
object
found
for
%1.

Explanation: No matching SAP GUI Scripting object found with SAP Identifier containing Regular Expression.

System action: Regular Expression will be ignored in SAP GUI Scripting Identifier.

User response: Check the Regular Expression in SAP GUI test.

RPSF0198E

SAP
GUI
Scripting
Identifier
with
Regular
Expression
are
strongly
not
recommended
in
schedule
execution.

Explanation: For performance purpose, SAP GUI Scripting Identifier with Regular Expression are strongly not recommended in schedule execution.

System action: No action (execution not stopped).

User response: Use original SAP GUI Scripting Identifier, avoid Regular Expression in SAP GUI test.

RPSF0199E

Invalid
Regular
Expression
detected

Explanation: SAP GUI Scripting Identifier with invalid Regular Expression detected during execution.

System action: Regular Expression will be ignored in SAP GUI Scripting Identifier.

User response: Check the Regular Expression in SAP GUI test.

RPTA0000W

%1

RPTA0001I

Setting
the
log
verbosity
left
me
with
%1
users

RPTA0002E

A
Test
cannot
be
launched
on
the
specified
Driver

RPTA0003E

%1

RPTA0004E

A
Test
could
not
be
launched
on
Driver:
%1.
The
Test
Execution
Framework
was
not
able
to
deliver
an
Executor.
This
is
an
internal
error,
please
contact
support.
For
more
information,
see
the
Troubleshooting
section
of
the
online
help.

Explanation: The Test and Performance Tools Platform (TPTP) infrastructure did not produce an executor for the test. This error message might display if firewalls are active on the local computer or the agent computer.

User response: Disable firewalls on both the local computer and the agent computer. If you do not want to disable firewalls, you can instead enable a firewall-aware connection. For more information on enabling a firewall-aware connection, see *Running with a workbench behind a firewall*. On the local computer, check the properties of the location that represents the agent computer. This error can occur if the deployment root directory is not specified correctly in the location that represents the agent computer. Check the Error Log for further information on the error. To open the Error Log, click *Window > Show View > Error Log*. Restart the Agent Controller. Restart the application.

RPTA0009E

A
Test
could
not
be
launched
on
Driver:
%1
due
to
an
internal
error.
Please
see
Problem
Determination
Log.
For
more
information,
see
the
Troubleshooting
section
of
the
online
help.

Explanation: An exception was thrown during an attempt to obtain the operating system attribute of the location asset.

User response: Check the Error Log for further information on the error. To open the Error Log, click Window > Show View > Error Log. Open the location asset representing the agent computer in the Test Navigator, and verify that all information and properties are correct. Delete the location asset representing the agent computer in the Test Navigator, and create a new location asset. You might need to delete the location and create a new one, if the location asset representing the agent computer asset is corrupted.

RPTA0010E

An
error
has
been
encountered
while
launching
a
Test
on
Driver:
%1.
Please
see
Problem
Determination
Log.
For
more
information,
see
the
Troubleshooting
section
of
the
online
help.

Explanation: An exception was thrown while starting the test. The exception did not contain an error message.

User response: Check the Error Log for further information on the error. To open the Error Log, click Window > Show View > Error Log. Restart the Agent Controller. Restart the application.

RPTA0011E

An
error
has
been
encountered
while
launching
a
Test
on
Driver:
%1.
An
Executor
was
not
returned
and
neither
was
an
error
message.
This
is
an
internal
error,
please
contact
support.

Explanation: The Test and Performance Tools Platform (TPTP) infrastructure produced neither an executor for this test nor error messages.

User response: Check the Error Log for further information on the error. To open the Error Log, click Window > Show View > Error Log. Restart the Agent Controller. Restart the application.

RPTA0012E

An
error
has
been
encountered
while
launching
a
Test
on
Driver:
%1.
There
are
no
Data
Processors
present.
This
is
an
internal
error,
please
contact
support.

RPTA0013E

An
error
has
been
encountered
while
launching
a
Test
on
Driver:
%1.
Data
Processors
have
not
been
configured
correctly.
This
is
an
internal
error,
please
contact
support.
For
more
information,
see
the
Troubleshooting
section
of
the
online
help.

Explanation: The test application was unable to configure the Data Processor for either the test log or the statistics portion of the test infrastructure.

User response: Check the Error Log for further information on the error. To open the Error Log, click Window > Show View > Error Log. Restart the Agent Controller. Restart the application.

RPTA0014E

A
Test
could
not
be
launched
on
Driver:
%1.
The
Test
Execution
Framework
encountered
an
Exception.
This
is
an
internal
error,
please
contact
support.

RPTA0015E

An
error
was
encountered
while
launching
a
Test
on
%1.
\nPlease
examine
your
Deploy
Directory:
%2,
the
error
could
be
caused
by
one
of
the
following:
\n
\n1.
The
Deploy
Directory
path
must
be
absolute
(start
with
Drive
Letter
or
"/").
\n2.
The
Deploy
Directory
path

RPTA0016E

An
error
has
been
encountered
while
launching
the
test.
A
required
dataset
%1
is
missing
or
invalid
in
your
project.

RPTA0017E

An
error
has
been
encountered
while
launching
the
test.
A
required
dataset
%1
has
been
replaced.
One
or
more
test(s)
are
referencing
a
different
version
of
the
dataset.

RPTA0018E

ready

RPTA0019E

not
ready
on
port

RPTA0020E

Check
Agents
Failed

RPTA0021E

%1
deployment
directory
%2
format
not
compatible
for
operating
system
%3.
For
more
information,
see
the
Troubleshooting
section
of
the
online
help.

Explanation: The deployment directory that is specified in the location asset representing the agent computer is incorrect for the operating system that is specified in the location asset.

User response: Open the location representing the agent computer in the Test Navigator, and edit the deployment directory or the operating system.

RPTA002E

Timed
out
after
%1
seconds
waiting
for
the
license
server.
Check
network
connectivity
to
the
license
server
and
ensure
the
license
server
is
running.
For
more
information,
see
the
Troubleshooting
section
of
the
online
help.

Explanation: The test application was unable to connect to the license server in the allotted time.

System action: The test run stops.

User response: Run the __RLKA_NAME__ to check for connectivity to the license server or to point to a different server.

RPTA0023E

Virtual
users
have
exited
prior
to
stage
completion.

At
the
end
of
stage
%1
there
were
%2
users
running
when
%3
were
expected.

A
common
reason
for
this
is
a
schedule
which
has
assigned
an
insufficient
amount
of
work
(for
one
or
more
User
Groups),
to

Explanation: During schedule execution, at the end of the current stage, the actual number of users running did not match the expected number of users. For example, if the current stage specifies that 100 users should run for 1 hour and only 90 users are running at the end of the hour, this message is displayed.

User response: Check the Error Log for further information on the error. To open the Error Log, click Window > Show View > Error Log. Typically, this message is displayed when virtual users did not have enough work to do for the duration of the stage. For schedules that contain more than one stage, verify that the workload under each user group is contained inside an infinite loop. Use infinite loops because the stage duration is controlled by the time when users stop. If virtual users have sufficient workload, look in the test log for more information about why virtual users stopped. The virtual users that stopped might have encountered errors. By default, this message is displayed when the number of expected users does not match the number of actual users running at the end of a stage. You can change this setting to specify the percentage of users that may stop during a stage without being considered an error. To change the error condition, create the -DrptStopTolerance property in the eclipse.ini file in the installation directory. For example, -DrptStopTolerance=80 specifies that 80% of the users may stop unexpectedly during stage execution without being considered an error.

RPTA0024E

Exception
encountered
adding
or
removing
users.

Explanation: This error message is displayed when a dataset reference between a test and a dataset is broken. Whenever a dataset is used, a reference is created in the test. The reference is a link that points to the physical dataset file in the test project. This link can break if the test is copied or imported into another project without copying or importing the associated dataset file. This link can also break if the dataset file is deleted.

System action: None.

User response: Do not copy or import individual test assets. Instead, copy or import entire projects. If you have already copied or imported individual test assets, copy the dataset from the previous project or create a new dataset that contains the same information. Open the test with the broken reference and link the dataset to the test. \nDo not delete dataset files.

RPTA0025E

The
schedule
has
no
user
group.

RPTA0026E

The
RPT_VMARGS
option
rptPre811PageResponseTimes
is
specified
on
at
least
one
location
and
is
missing
from
at
least
one
other
location.
Please
ensure
that
either
all
locations
include
this
option
or
none
do.
See
"adjusted
page
response
time
for
increased
accuracy"
in
the
help
for
more
information

RPTA0025IRun
Completed
(%1)

RPTA0026IRun
Terminated
(%1)

RPTA0027I%1:
%2

RPTA0031ELocation
template
file
%1
is
not
found
(referenced
from
location
file
%2)

Explanation: A location template file referenced by a location file is missing or inaccessible.

User response: Create a location template file with the given name. If the location template file exists but is in a closed project, open the project.

RPTA0032I

Found
location
template
[%1]
for
[%2]
(instances
found:
%3)

RPTA0033I

%1
remote
location(s)
associated
with
location
template
[%2]

RPTA0034E

Cannot
change
stage
duration
if
Until
Finished
specified

RPTA0035E

Duration
time
specified
is
less
than
what
has
already
elapsed

RPTA0036E

Schedule
must
be
in
the
Running
state
to
change
stage
duration

RPTA0037E

Agent
%1
not
ready,
time
of
last
contact:
%2

Explanation: The specified agent is not in contact with the workbench. The schedule cannot run until all agents that are used in the schedule are actively connected.

System action: Install and configure the Rational® Performance Tester load generation agent on the agent computer.

User response: Ensure that the specified agent has a load generation agent installed and is properly configured to this workbench. Restart the schedule. See the online help for information about how to install and configure the load generation agent.

RPTA0038E

No
successful
contact

RPTA0039E

Unknown
host
'%1'

Explanation: The specified agent name is not resolving in the Domain Name System (DNS).

User response: Ensure that the agent name is spelled correctly in the location.

RPTA0040E

Unable
to
complete
deployment
to
agents
because
of
an
unexpected
error
in
the
publish
phase.
%1

Explanation: A deployment error occurred that is likely a low-level I/O error or an unrecoverable internal error.

User response: Check the exception messages for possible causes such as a lack of hard-disk space.

RPTA0041E %1

Explanation: The specified agent is not in contact with the workbench. The schedule cannot run until all agents that are used in the schedule are actively connected.

System action: Install and configure the Rational® Performance Tester load generation agent on the agent computer.

User response: Ensure that the specified agent has a __PT_RR_SHORTNAME__ load generation agent installed and is properly configured to this workbench. Restart the schedule. See the online help for information about how to install and configure the load generation agent.

RPTA0042E Agent
version
%1
incompatible
on
host
%2.
Minimum
agent
version
%3
required.

Explanation: The version of the __PT_AGENT_ACRONYM__ is not compatible with a feature in the schedule.

System action: The schedule cannot be launched so schedule execution ends.

User response: Upgrade the __PT_AGENT_ACRONYM__ on the machine specified to match the workbench version.

RPTA0043E Error
encountered

Explanation: An unexpected error occurred.

User response: Look for more details about the error in the message posted.

RPTA0100W

Failed
to
delete
file
%1

RPTA0518E

An
error
has
been
encountered
while
launching
the
test.
A
required
dataset
%1
is
missing
or
invalid
in
your
project.

Explanation: A test contains a link to a dataset that cannot be found or that is corrupted. This can happen when a project is not imported completely, or when a file is deleted.

System action: The test run does not start.

User response: Open the test. On the Common Options page, fix the broken link so that it points to a valid dataset file or delete the link.

RPTA1050E

Rational®
Service
Tester
for
SOA
Quality
is
licensed
to
only
support
single
user
execution.
Please
adjust
the
number
of
Users
to
1
in
the
schedule
and
rerun.
Contact
IBM
regarding
the
use
of
Rational®
Performance
Tester
for
your
load
testing
needs.

RPTC0003E

Wrong
type
of
project
'%1'.

RPTC0004E

Unable
to
access
test
variable
initialization
file.
Make
sure
the
specified
file
path
is
accessible:
%1

RPTC0005E

Error
while
processing
XML
file
containing
variable
initializations.
Make
sure
the
file
contains
valid
XML
of
the
expected
format:
%1

RPTC0006E

Error
while
gather
test
variable
initializations.
No
variable
initializations
will
be
honored
for
this
run.

RPTC0007E

Error
processing
license
request
for
feature
'%1'.
This
feature
will
not
be
available.

Explanation: The workbench could not find the ibmrpt_pvu license. Either the license does not exist with the license server or activation kit or the workbench was unable to acquire it. As a result, the capabilities that this license enables are not be available.

User response: To enable the capabilities for the license, ensure that the ibmrpt_pvu license is available for the workbench.

RPTC0008I

Setting
Variable
[name='%1',
value='%2',
source='%3',
user
group='%4',
location='%5']

RPTC00020E Unexpected
I/O
error
while
communicating
with
workbench
%1

Explanation: During test-log transfer a network error occurred on the agent communicating to the workbench.

System action: The agent re-attempts to communicate with the workbench.

User response: If the problem persists, inspect error and take corrective action.

RPTC1001W The
file
path
specified
for
the
Zip
Utility
is
invalid.

RPTC1002W Could
not
get
the
classpath
for
project
'%1'.

RPTC1009I Undefined

RPTC1011l

%1:
Request
delivered

RPTC1012l

%1:
successfully
added
%2
to
the
configuration
file

RPTC1013l

%1:
successfully
removed
%2
from
the
configuration
file

RPTC1014l

%1:
%2
is
already
in
the
configuration
file

RPTC1015I

%1:
Request
timed
out

RPTC1016I

%1:
Agent
not
ready

RPTC1017I

%1:
Agent
not
known

RPTC1018I

%1:
Unknown
host
exception

RPTC1019I

%1:
%2

RPTC1020I

License
type:
%1

Explanation: Lists the brand of licensing being used (either HCL or IBM).

System action: License checkouts will attempt to acquire a license of the corresponding type.

User response: No action required.

RPTC1021I

License
valid:
%1

Explanation: Indicates whether a valid license was successfully acquired (true/false).

System action: If true, the functionality associated with the acquired license will be enabled.

User response: If false, check your license configuration.

RPTC1030E

Unable
to
replace
dataset
'%1'
with
'%2':
%3.

Explanation: An error occurred attempting to replace datasets.

System action: Execution will complete with error.

User response: Refer to the error message for more details, change the command line options related to replacing datasets.

RPTC1031E

The
dataset
'%1'
doesn't
exist.

Explanation: Unable to locate the specified dataset referenced in the dataset command line option.

System action: Command line execution will be cancelled.

User response: Change the command line options related to replacing datasets.

RPTC1032E

The
dataset
'%1'
is
incompatible
with
existing
dataset
'%2'.

Explanation: The specified replacement dataset does not have compatible columns, type, etc.

System action: Execution will complete with error.

User response: Ensure the dataset has the same columns of the dataset it is replacing.

RPTD0011W

The
regular
expression
you
provided
does
not
find
the
highlighted
string.
\nThe
string
found
was:
%1

Explanation: You have provided a regular expression that does not match the string you want to extract.

System action: None. We will let you use that regular expression but at execution time you might not get the string you are expecting.

User response: Fix the regular expression so that it matches the highlighted string. You can use external tools to help you.

RPTE0005W

Unable
to
attach
requirements
report
into
RQM
result,
because
the
default
requirements
report
has
been
deleted.
You
can
recreate
the
default
reports
by
click
restore
defaults
button
on
the
Default
Reports
preference
page.

Explanation: When a test run started by __QM_NAME__ completes, the default report is attached to the __QM_NAME__ execution results. This error occurs when the report selected as the default report on the Default Report preferences page does not exist.

System action: No report is attached to the __QM_NAME__ execution results.

User response: Click Window > Preferences > Test > Performance Test Reports > Default Report to open the Default Report preferences page. Check that the selected report exists. Click Restore Defaults to reset the default reports.

RPTE0011W

Unexpected
error
while
releasing
system
resources
for
test
log
export.
This
may
cause
an
increased
memory
footprint,
until
Rational®
Performance
Tester
is
restarted.

Explanation: Test log export has completed (possibly with errors described earlier in the workspace log), but when releasing assets used during the export operation, there was an unexpected error.

System action: Memory allocated to this operation may not have been freed. Previous errors are likely to be present explaining the root cause.

User response: It is advisable to restart the application to free memory allocated during this operation. The exported test log file may be available but there may be errors.

RPTE0147E

The
password
saved
for
an
encrypted
column
in
dataset
"%1"
was
invalid.
Set
a
new
password
in
the
Automation
Security
preference
page.

Explanation: The value saved in the Automation Security preference page for the specified dataset was not correct. It will be ignored.

System action: The password in the preference is ignored. If running from the workbench, it will prompt for a password before execution. Otherwise, execution will fail.

User response: Update the password in the Test - Test Execution - Automation Security preference page.

RPTE0150E

The
feature
%1
used
in
test
%2
is
not
checkmark_perf.jpg
in
the
current
installation/
platform.

Explanation: The execution failed because the specified feature is not checkmark_perf.jpg in the current installation of the product.

System action: Ensure feature is selected during installed. Ensure feature is checkmark_perf.jpg on the given architecture/operating system.

User response: No user action is required.

RPTH0130I

No
sample
time
closely
matches
request
at
time=
%1

RPTH049E

A
statistical
adapter
is
missing
reference
to
the
target
result.

Explanation: This is an internal error when loading results files. It could indicate that the result is corrupted, or it could only be a timing issue.

System action: The result cannot be opened.

User response: Close all reports and restart the workbench. If the result still does not open, kill any CPU-intensive processes running in the background.

RPTI0069E

Local
on
premise
agent
%1
not
in
contact
with
this
workbench.

RPTI0070E

See
Error
Log
for
more
details.

RPTl0071l

There
was
an
error
while
updating
the
workspace
after
downloading
remote
files.

RPTl0072E

Modify
majordomo.config
on
%1
and
configure
it
to
poll
this
workbench.

RPTl0072l

Remote
Launch
Status:
%1

RPTl0073E

Project
is
NULL

RPTI0074E

Exception
occurred
while
creating
and
unzipping
project:
%1

RPTI0075E

Error
running
schedule.
Could
not
find
schedule
%1
in
project
%2.

RPTI0110I

Provision
time
(MM:SS):
%1

RPTI0111I

Launch
time
(MM:SS):
%1

RPTl0112l

Execution
time
(MM:SS):
%1

RPTl0113l

Results
transfer
time
(MM:SS):
%1

RPTl0141E

```

-----
\nError
Dialog
\n
%1:
%2\nConsult
workspace
error
log
({workspace}/.metadata/
.log)
for
further
information.
\n-----
\n

```

Explanation: This message is displayed to the command-line output when an error occurs during execution. It displays details about the error and directs the user where to find additional information.

System action: None.

User response: This message occurs as a generic way to display errors during command-line execution. Consult the workspace log for further details including additional error messages.

RPTI0142E

The
Usage
Metrics
version
%1
required
by
the
licensed
component
%2
is
not
available.

Explanation: The license that you are using requires Usage Metrics reporting for a later version of the product.

System action: The execution will not start.

User response: Update the product to a newer version, or obtain a license that is applicable to the current version of the product.

RPTI0143E

The
licensing
system
failed
to
return
Usage
Metrics
enablement
for
component
%1.

Explanation: An error occurred while determining if the license requires Usage Metrics reporting.

System action: The execution will not start.

User response: Verify that the license is not meant for a newer version of the product. Otherwise, contact support.

RPTl0144W

No
RTCP
instance
is
available
to
report
Usage
Metrics.
No
Usage
Metrics
will
be
reported
for
this
execution.

Explanation: The license enables Usage Metrics reporting, but either the preference for the Usage Metrics server is not set, or it is set but the server is not active or reachable.

System action: The execution will be done normally, but the Usage Metrics will not be logged. This is allowed by the license you are using.

User response: If you have set up `__QUALITY_SERVER__`, go to Preferences > Test > `__QUALITY_SERVER__`, and fill in the server details for Usage Metrics reporting. Verify that the server can be reached from this machine by going to `http://servername:7828` in a browser on the local machine.

RPTI0145E

No
RTCP
instance
is
available
to
report
Usage
Metrics.
Per
license
policy,
execution
cannot
happen
unless
a
RTCP
is
defined
and
running.

Explanation: The license requires Usage Metrics reporting, but either the preference for the Usage Metrics server is not set, or it is set but the server is not active or reachable.

System action: The execution will not start.

User response: Install __QUALITY_SERVER__ (if not done already), then go to Preferences > Test > __QUALITY_SERVER__, and fill in the server details for Usage Metrics reporting. Verify that the server can be reached from this machine by going to <http://servername:7828> in a browser on the local machine.

RPTI0146E

TPTP
Datapools
and
Datasets
cannot
coexist
in
the
same
test.
Test
run
aborted.

Explanation: A legacy datapool and a new dataset were both detected in the same test.

System action: Test execution will be aborted and will not be successful until the test contains only one of the two asset types (dataset or datapool).

User response: With the latest version of this product, convert the datapool to a dataset, then open the test containing the legacy datapool in the test editor and save it. Then, restart test execution.

RPTJ0063E

An
IOException
was
encountered
while
creating
the
Annotation
File
on
Driver:
%1

RPTJ0075E

An
IOException
was
encountered
while
creating
the
Execution
Log
File
on
Driver:
%1 ::
%2

RPTJ1002E

Driver
%1
returned
an
unrecognized
response:
%2.
The
last
command
sent
was:
%3

RPTJ1003E

While
waiting
for
an
acknowledgement
from
the
Driver,
an
unrecognized
response
was
received.

RPTJ1004E

The
workbench
was
waiting
for
an
Acknowledgement
from
the
IBM
Agent
Controller
on
Driver
%1
and
none
was
received.

Explanation: A required response from an agent was not received.

System action: Execution ends because the required acknowledgement from the agent was not received.

User response: Monitor resource usage on the agent. Add additional agents if memory or CPU usage is high on a any agent.

RPTJ1005E

Error
while
processing
a
message
from
the
IBM
Agent
Controller.

Explanation: An unexpected error occurred while handling a command from a load generating agent.

System action: Execution ends because of an unexpected error while communicating with an agent.

User response: Check the workbench Error Log for more information.

RPTJ1006E

Execution
failure.
No
status
received
from
location
%1
in
%2
seconds.
Workbench
memory
usage
at
%3
percent
of
the
configured
JVM
heap.
Possible
location
or
workbench
overload.
For
more
information,
see
the
Troubleshooting
section
of
the
online
help.

Explanation: The workbench cannot communicate with the agent computer.

User response: Try running the schedule again, using default values for all parameters and running at reduced user load levels. It is possible one agent computer is overloaded. If you can run successfully with the default values, make changes to the schedule settings or user load incrementally to determine the cause of failure. Increase the statistics interval to 60 seconds and try running the schedule again. Check the error log for messages that might indicate the cause of the failure. Click Window > Show View > Error Log to open the error log.

RPTJ1007E

The
Driver:
%1
has
encountered
a
communication
error.
Please
refer
to
Problem
Determination
Log
for
more
details.
For
more
information,
see
the
Troubleshooting
section
of
the
online
help.

Explanation: The agent computer that the message specifies encountered a problem when trying to run a command sent from the workbench.

User response: Check the error log for messages from the agent computer that the error message specifies. Click Window > Show View > Error Log. Check the test log for any failures from virtual users. This message might be displayed when you add or remove users manually or by means of schedule stages.

RPTJ1008E

The
Driver:
%1
has
become
unresponsive,
possibly
due
to
an
out-
of-
memory
condition.
At
last
notification
this
Driver
was
using
%2
percent
of
its
allocated
memory.
Please
refer
to
the
"Increasing
memory
allocation"
Help
topic
for
information
on
how
to
increase
memory
allocation.
For
more

Explanation: The workbench cannot communicate with the agent computer. The agent computer might have a memory allocation problem.

User response: Try running the schedule again, using the default values for Test Log and Problem Determination log levels. Follow the instructions in Increasing memory allocation. Set the memory allocation to the size of physical memory minus 256 megabytes, up to a limit of 1500 megabytes. For example, on an agent computer with one gigabyte of physical memory, set the memory allocation to 756 megabytes.

RPTJ1009E

The
Driver:
%1
is
running
%2,
however
the
user
selected
%3
as
the
Drivers
operating
system.

RPTJ1010E

Error
while
transferring
file
on
Driver:
%1.
Transfer
FROM:
%3
TO:
%2

RPTJ1011E

The
'%1'
Protocol/
Feature
is
not
checkmark_perf.jpg
on
the
%2
platform,
so
the
Test
%3
can't
be
executed
on
location
%4.
For
more
information,
see
the
Troubleshooting
section
of
the
online
help.

Explanation: The test includes a feature or protocol that is not checkmark_perf.jpg on one of the agent computers where it is scheduled to run.

User response: Edit the schedule and associate the user groups that include the problem test with agent computers that support the feature or protocol.

RPTJ1012E

The
operating
system
(%1)
for
location
%2
is
not
recognized.
Please
use
an
operating
system
that
matches
or
begins
with
the
name
of
one
of
the
recognized
platforms:
%3

RPTJ1013E

No
valid
license
key
for
%1
Protocol/
Feature
found.
The
Test
%3
cannot
be
executed.
For
more
information,
see
the
Troubleshooting
section
of
the
online
help.

Explanation: The test includes a feature or protocol that requires a license for the number of virtual users that are included in the run.

System action: The test run stops.

User response: Run the IBM License Key Administrator and check for available license keys for the feature or protocol and number of users that you want. To learn more about license keys, see the installation guide.\nAdd the required license key or point to a server that has the required license key.

RPTJ1014E

Execution
on
the
%1
Platform
requires
a
license
and
no
valid
license
key
(%2)
was
found
to
enable
it,
so
the
Test
%3
can't
be
executed
on
location
%4.
For
more
information,
see
the
Troubleshooting
section
of
the
online
help.

Explanation: One of the agent computers that is specified for the test run requires a license, but no license key was available for that platform.

User response: Run the Rational License Key Administrator and check for available license keys for the platform that you want. To learn more about license keys, see the installation guide. Add the required license key or point to a server that has the required license key or run the test on a different platform.

RPTJ1015E

The specified operating system (%1) for location %2 is inconsistent with the actual platform (%3) running at that location. Please update the operating system to match and then try again. For more information, see the Troubleshooting section of the online help.

Explanation: The operating system that is specified in the agent computer asset does not match the operating system that is running on the computer at the specified address.

User response: 1. Open the schedule in the test editor. 2. Select the user group that runs on the location mentioned in the error message. 3. In the Schedule Element Details, click the Locations tab, and then select the location mentioned in the error message. 4. Click Edit. 5. Select the appropriate value from the Operating system list. 6. Click OK.

RPTJ1016E

After
deploying
File:
%2
to
Driver:
%1,
%3
Byte(s)
where
found
on
the
socket.
Please
refer
to
the
Problem
Determination
Log
for
more
details.

RPTJ1017E

An
IOException
occurred
while
deploying
File:
%2
to
Driver:
%1.
Please
refer
to
the
Problem
Determination
Log
for
more
details.

RPTJ1018E

A
SocketException
occurred
while
deploying
File:
%2
to
Driver:
%1.
Please
refer
to
the
Problem
Determination
Log
for
more
details.

RPTJ1019E

An
UnsupportedEncodingException
occurred
while
deploying
File:
%2
to
Driver:
%1
Please
refer
to
the
Problem
Determination
Log
for
more
details.

RPTJ1020E

An
IOException
occurred
while
deploying
File:
%2
to
Driver:
%1.
\nA
possible
cause
is
that
the
IBM
Agent
Controller
was
started
by
a
non-
root
user.
\nThe
Agent
Controller
needs
to
be
started
by
the
root
user.

Explanation: Deployment of test assets to an agent failed.

System action: Execution ends because required test assets could not be copied to an agent.

User response: Ensure that the Majordomo process is started by the root user.

RPTJ1021E

An
InactiveAgentException
has
occurred
while
deploying
to
Driver:
%1.
Please
refer
to
the
Problem
Determination
Log
for
more
details.
For
more
information,
see
the
Troubleshooting
section
of
the
online
help.

Explanation: The Test and Performance Tools Platform (TPTP) infrastructure threw an InactiveAgentException when the TPTP infrastructure attempted to communicate with the Agent Controller.

User response: Check the Error Log for further information on the error. To open the Error Log, click Window > Show View > Error Log. Restart the Agent Controller on the agent computer.

RPTJ0121I

Send
RATEGENERATORS
to:
%1,
string
'%2'

RPTJ1022E

The
workbench
received
notification
that
the
execution
process
on
Driver
%1
has
terminated.

Explanation: The process running on the agent computer ended unexpectedly.

User response: Ensure that there is at least one successful test run, possibly with fewer virtual users, so that the maximum memory value for the agent is set correctly. Check the javacore* file on the agent computer or the logs in the deployment directory for further information on the process failure.

RPTJ1023E

Communication

with

Driver

%1

has

been

lost,

possibly

due

to

an

out-

of-

memory

condition.

At

last

notification

this

Driver

was

using

%2

percent

of

its

allocated

memory.

Please

refer

to

the

"Increasing

memory

allocation"

Help

topic

for

information

on

how

to

increase

memory

allocation.

For

RPTJ1024E

Error
during
initialization
of
annotation
transfer
progress
listener.

RPTJ1025I

Run
Completed
(%1)

RPTJ1026I

Run
Terminated
(%1)

RPTJ1030E

Non-
fatal
internal
exception
occurred
during
code
generation
optimization.
Code
generation
will
not
use
meta-
cache.

RPTJ1040E

The
license
required
for
the
'%1'
Protocol/
Feature
and
%2
virtual
users
could
not
be
checked
out,
so
the
Test
%3
cannot
be
executed.
For
more
information,
see
the
Troubleshooting
section
of
the
online
help.

Explanation: The test application was unable to check out a license for a feature or protocol in the test run.

User response: Run the Rational License Key Administrator and check for available license keys for the platform to run the feature or protocol. Add the required license key, point to a server that has the required license key, or run the test on a different platform.

RPTJ1041E

The
'%1'
Protocol/
Feature
is
disabled
due
to
a
licensing
configuration
error.

RPTJ1042E

%1
Failure
checking
out
license
for
'%2'
Protocol/
Feature
and
%3
virtual
users.
The
Test
%4
cannot
be
executed.

RPTJ1043E

%1
The
'%2'
Protocol/
Feature
is
not
checkmark_perf.jpg
on
the
%3
platform,
so
the
Test
%4
can't
be
executed
on
location
%5.

RPTJ1044E

Timed
out
after
%1
seconds
while
waiting
for
the
license
server.
Ensure
that
network
connectivity
to
the
license
server
exists
and
that
the
license
server
is
running.
For
more
information,
see
the
Troubleshooting
section
of
the
online
help.

RPTJ1100I

A
hang
has
been
avoided
during
execution
history
receipt
with
%1
by
a
forceful
load
test
executor
state
change

RPTJ1101E

A
session
on
driver
%1
did
not
release
promptly.
Please
check
the
agent
controller.

RPTJ1102W

The
testLog
event
loader
thread
in
the
workbench
has
ended
before
processing
all
testLog
events
from
%1.
The
testLog
may
be
incomplete.

RPTJ1103W

The
test
executor
for
%1
has
been
artificially
set
to
HISTORY_COMPLETE
because
the
testLog
event
loader
thread
is
not
longer
running.

Explanation: A monitoring process indicates that the test log loader stopped prematurely. This is not a definite indication of a problem.

User response: Check that the expected events exist at the end of the test log. If so, no further action is necessary.

RPTJ1104E

Remote
debug
never
received
event
%1,
process
exit
value
%2

Explanation: Expected remote debug event was not received

User response: Check the Error Log for remote process failure reason

RPTJ1141E

Temporary
dataset
file
%1
not
created.

Explanation: Temporary dataset file can't be created on the system.

System action: Original dataset is used.

User response: Check corresponding file properties on the system.

RPTJ1142E

Temporary
dataset
data
are
not
generated:
%1

Explanation: Error reached during data generation.

System action: Original dataset is used.

User response: Check corresponding connection information.

RPTJ1200W

Failed
to
delete
file
%1

RPTJ1220E

An
InactiveAgentException
has
occurred
attempting
to
send
[%1]
to
driver
%2

RPTJ1221E

The
CommandHandler
for
%1
has
encountered
an
exception
while
processing
%2

RPTJ1240E

Driver
%1
has
reported
a
NOK.
The
last
command
sent
to
that
driver
was:
%2

Explanation: A schedule command sent from the workbench to the agent computer could not be run by the agent computer.

User response: Run the schedule using the default settings. Look for unusual assignments of numbers of virtual users to agent computers at stage transitions.

RPTJ1241E

Driver
%1
has
reported
a
NOK
with
the
message:
%2.
The
last
command
sent
to
that
driver
was:
%3

RPTJ1242E

Driver
%1
has
reported
a
%2
status

RPTJ1244E

The
AgentCommandListener
for
%1
has
encountered
an
exception
while
processing
%2

RPTJ1245E

Driver
%1
has
reported
that
it
is
no
longer
receiving
messages
from
the
workbench.
The
previous
message
received
from
this
driver,
%2
milliseconds
ago,
was
%3.
At
present
no
commands
have
been
sent
to
this
driver.

Explanation: Schedule commands sent from the workbench to the agent computer were not received by the agent computer.

User response: Ensure that there is at least one successful test run, possibly with fewer virtual users, so that the maximum memory value for the agent is set correctly. Use more agent computers to run the schedule.

RPTJ1261E

The
ResponseHandler
for
%1
has
encountered
an
exception
while
processing
%2

RPTJ1270E

Failure
attempting
to
launch
test
execution.

RPTJ1271E

The
process
executing
the
test
has
ended
unexpectedly.

Explanation: The process that runs tests could not start, or it stopped before the test run ended.

System action: The test run stops.

User response: Check the core files or the logs for further information on the process failure. If you are using Java Virtual Machine (JVM) arguments, check the argument syntax and try running tests without the arguments. Run the test inside a schedule.

RPTJ1280E

The communication path for returning test results from %1 has not been established. Check network connectivity between that machine and the workbench including any firewalls.

RPTJ1400I

%1%
%2/%3
files
%4/%5
bytes
deployed

RPTK0000I

%1

RPTK1001E

Rational®
Performance
Tester
has
detected
the
presence
of
an
invalid
Virtual
Tester
license
key.
If
you
have
recently
upgraded
Rational®
Performance
Tester,
note
that
this
is
a
new
check
performed
by
release
7.0.1
or
later,
and
instructions
for
replacing
invalid
Virtual
Tester
license
keys
should
have

Explanation: Invalid Virtual Tester license key(s).

System action: System will not execute schedule run(s) that require a Virtual Tester license if one is not available.

User response: You must replace all invalid Virtual Tester license keys. If you need further assistance, please contact your sales representative or Technical Support.

RPTK1016E

The
specified
license
server's
version
level
is
not
compatible
with
this
version
of
Rational®
Performance
Tester.

Explanation: The specified license server's version level is not compatible with this version.

System action: Incompatible version.

User response: Check the license server's version.

RPTK1019E

Unable
to
verify
system
time.

Explanation: The system time has been tampered with since the last successful license check.

System action: Future license checks will automatically fail.

User response: Contact Technical Support.

RPTK1020E

Unable
to
locate
license
directory.

Explanation: Unable to locate license directory.

System action: Stop execution.

User response: Please verify that the license directory exists.

RPTK1021E

License
has
expired.

Explanation: An expired license was found.

System action: Request a license key from user.

User response: Enter a new license key.

RPTK1022E

Invalid
license
file.

Explanation: A valid license was not found.

System action: Request a license key from user.

User response: Enter a valid license key.

RPTK1023E

Unable
to
find
a
license
supporting
%1
virtual
users.

Explanation: The currently installed license key(s) do not support enough VUs for this operation.

System action: Request a license key from user.

User response: Enter another license key to enable more VUs.

RPTL0001W

Unable
to
retrieve
data
from
the
test.

RPTL0002W

Failed
to
store
test
data
into
annotations.

RPTL0003W

Failed
to
attach
the
annotation
to
the
test.

RPTL0004W

Unable
to
open
test
annotation
to
read
data.

Explanation: The test appears to be corrupted.

System action: Attempts to open the test fail.

User response: Make sure your disk has enough space. If it does, try recreating the test from the recording.

RPTL0005W

Failed
to
create
a
temporary
file
to
save
test
data.

RPTL0006W

Failed
to
load
test.
Path
%1
is
invalid.

RPTL0007W

Failed
adding
element
from
an
un-
registered
feature
%1.

RPTL0008E

Cannot
load
a
test
created
by
a
future
version
%1.
Please
upgrade
your
install.

RPTL0009I

Test
%1
is
of
an
older
version
%2.

RPTL0010E

Error
creating
metadata
cache.

RPTL0011E

Error
reading
metadata
cache
for
%1.

RPTR0000W

%1

RPTR0001W

Failed
to
add
annotation
to
execution
history
for
file
%1

RPTR0002W

Unexpected
error
in
data
validity
check
of
LoadTimeEObjectConsumer

RPTR0003W

Failed
to
add
properties
to
parent
id
%1

RPTR0004W

Failed
to
delete
temp
file
%1

RPTR2001E

Unexpected
exception
in
container
complete
loader.
Heap
growth
likely.

RPTR2003W

Execution
Variables
-
Output

RPTS1000E

Unable
to
start
the
agent
communication
service
because
of
an
error:
%1.
RPT
will
not
be
able
to
execute
schedules.

Explanation: The agent communication service could not start. This service is a lightweight web server that agents use to communicate with the workbench and to serve web reports. Typically, this error occurs when a server process on the workbench computer is listening on the same port that Rational® Performance Tester requires. This error can also occur when two instances of Rational® Performance Tester run on the same workbench.

User response: If multiple instances of the Rational® Performance Tester workbench are running on the same computer, close all but one instance. These instances include Rational® Performance Tester workbenches that are running on multiple user desktop systems. If the error message RPTS1002E_PORTS_CONSUMED is also displayed in the error log, see the message for that error. After the error is resolved, restart Rational® Performance Tester.

RPTS1002E

RPT
is
unable
to
execute
a
schedule
because
one
of
the
ports(%1)
it
uses
to
communicate
with
the
agents
has
been
taken
by
another
RPT(or
other
server)
process.
Ensure
only
one
RPT
instance
is
running.

Explanation: Typically, this error occurs when a server process on the workbench computer is listening on the same port that Rational® Performance Tester requires. This error can also occur when two instances of Rational® Performance Tester run on the same workbench.

User response: Identify and stop the other process or service on the workbench that is using the ports that Rational® Performance Tester requires. Restart Rational® Performance Tester. You can also change the ports that Rational® Performance Tester uses by configuring the workbench and all agent computers. To change the ports, click Window > Preferences > Test > Server, and click Preferences > Test > Performance Test Reports > Web Reports.

RPTS1510E

Unable
to
stop
the
agent
communication
service
because
of
an
error:
%1

RPTS1001I

Agent
communication
service
listening
on
ports(%1)

Explanation: The agent communication service requires these local server ports to communicate with agents.

System action: No system action is required.

User response: This message is for informational purposes only.

RPTS1004

The dataset service failed to shutdown cleanly. Please shutdown the java process that contains dataset-service in its command line.

Explanation: Sometimes the dataset service does not shutdown cleanly due to java.

System action: No system action is required.

User response: Check in the task manager to make sure the dataset service is not still running when the workbench shuts down. If it is running kill the process before restarting the workbench.

RPTX0001E

The
combination
of
transformer
and
feature
you
have
selected
is
invalid.
Transformer
(%1)
was
not
expecting
data
type
(%2).

RPTX0002E

The combination of feature and transformer you have selected is invalid. Feature (%1) was not expecting data type (%2) to be returned by transformer (%3).

RPTX0003E

Transformer (%1) has experienced a fatal error. Additional information (%2).

RPTX0004E

Feature
(%1)
has
experienced
a
fatal
error.
Additional
information
(%2).

RPTX0005E

No
class
can
be
found
for
the
specified
transformer
id
(%1).
Please
check
to
make
sure
you
have
installed
this
transformer.

RPTX0006E

Class
definition
missing.
Please
add
jar
that
contains
definition
of
(%1)
to
the
classpath
of
the
test
project.

Explanation: Some requests or responses contain data that is encoded for Google Web Toolkit (GWT). To decode the data, Rational® Performance Tester requires access to the class definition.

User response: Add the JAR file that contains the class definitions to the classpath of the test project.

RPTX0007E

The
transformation
raised
a
GWT
serialization
exception:
%1

Explanation: The Google Web Toolkit (GWT) transformation could not be applied because of the indicated reason.

User response: Verify that the test elements containing the GWT encoded or decoded data are correct

RPTX0008E

The
Silverlight
decoder
raised
an
exception:
%1

Explanation: The Microsoft Silverlight decoder did not work because of the indicated reason.

User response: Verify that the test elements containing the Silverlight encoded data are correct

RPTX0009E

The
Silverlight
encoder
raised
an
exception:
%1

Explanation: The Microsoft Silverlight encoder did not work because of the indicated reason.

User response: Verify that the elements containing the Silverlight decoded data are correct

RPTX0010E

The
GraniteDS
transformer
made
an
error
when
encoding
or
decoding:
%1

Explanation: The GraniteDS encoder did not work because of the indicated reason.

User response: Verify that the elements containing the GraniteDS encoded or decoded data are correct.

RPXD0022W

The
time
to
extract
references
seems
excessive.
It
was
%1
milliseconds.

Explanation: It is taking a long time to extract data from your response for your references.

System action: None.

User response: Examine each of the regular expressions for your references. Make sure they don't have .* with no qualifiers or other poorly formed constructs. When you write the regular expression in the test you can click verify to get an idea of how long it is taking to execute.

RPXE0061I

Loop
iteration
started
late
by
%1
milliseconds

Explanation: A scheduled loop iteration started execution later than expected given the specified rate.

System action: Execution continues along with attempt to catch up in order to maintain desired rate.

User response: Add additional users or agents to increase capacity in order to maintain desired rate.

RPXE5502E

An
exception
occurred
while
logging
an
event
to
Jaeger.

Explanation: An error occurred when attempting to log an event to Jaeger. The event will not be available in Jaeger traces.

System action: Jaeger logging will continue for the next events.

User response: Contact support.

RPTX1010I

Start
of
RPT
project
resolve.
Repository=<
%1>,
Bootstrap=<
%2>

RPTX1011I

Attempting
to
resolve
asset=<
%1>

RPTX1012l

End
of
RPT
project
resolve.
No
detected
errors

RPTX1017l

Downloaded
asset
%1
from
remote
repository,
local
asset
created.

RPTX1018l

Using
local
cached
version
of
asset
%1.

RPTX1019l

RPT
testsuite=<
%1>
found
the
following
dependencies=<
%2>

RPTX1081E

Exception
occurred
while
uploading
Mobile
report.

Explanation: A low-level exception occurred uploading the mobile report. It is unexpected.

System action: The RQM report will fail to upload.

User response: If possible take corrective action, otherwise contact support.

RPTX1082E

An
error
occurred
when
generating
the
HTML/
zip
report.

Explanation: The HTML generator for the Execution Report has failed.

System action: No execution report uploaded into RQM results

User response: Ensure that the temporary directory is accessible on your file system.

RPTX2001E

Adapter
unable
to
start
test
because
Rational®
Performance
Tester
is
already
executing
a
test.

Explanation: The adapter received a request to start a test while another test on the adapter is in-progress.

System action: The adapter ignores the request to launch another test.

User response: Wait for the test which is currently executing on the adapter to complete, then re-initiate the launch.

RPTX2002E

Error
encountered
parsing
RQM
adapter
preferences:
%1.
Please
enter
proper
credentials
in
the
Eclipse
Quality
Adapter
preference
page
(Windows-
>Preferences).

RPTX2003E

Project
<
%1>
could
not
be
found
during
RQM
import.

RPTX2004E

Test
log
is
unavailable,
no
test
results
returned
to
RQM.

RPTX2005E

Statistics
log
is
unavailable,
no
statistic
results
returned
to
RQM:
%1

RPTX2006W

Display
unavailable,
no
Rational®
Performance
Tester
HTML
reports
will
be
attached
to
RQM
execution
results.

Explanation: The adapter requires access to a virtual display to generate HTML reports. The adapter was unable to successfully create a display so HTML reports may be unavailable.

System action: HTML reports are not generated at the end of execution.

User response: If HTML reports are required, start the adapter with display access. Refer to documentation on how to start the adapter with a display.

RPTX2007I

Start
RQM
Execution
Request
Project=
%1
Name=
%2

RPTX2008I

Start
RQM
Import
Request
Project=
%1

RPTX2009I

End
RQM
Execution
Request

RPTX2010I

End
RQM
Import
Request

RPTX2011E

Unable
to
interpret
RQM
configuration
file
%1.
If
file
was
hand
edited
make
sure
parameters
are
the
correct
format.
If
you
are
unable
to
get
this
file
into
the
correct
format,
please
erase
and
re-
configure.

RPTX2012E

Invalid
RQM
connection
parameter:
%1.
Adapter
was
not
launched.

RPTX2013E

Adapter
was
stopped
while
a
test
was
executing.
The
results
of
this
test
may
be
unreliable.

RPTX2014E

Adapter
was
stopped
while
preparing
to
run
an
RQM
script.
There
are
no
results
for
the
attempted
test
script
run.

RPTX2015E

Testsuite
'%1'
or
project
'%2'
does
not
exist.
Ensure
workspace
started
by
adapter
contains
project
and
testsuite.

RPTX2016l %1

RPTX2017E %1
Reason:
 %2

RPTX2018W %1

RPTX2019l The
 RQM
 Adapter
 has
 been
 disconnected.

RPTX2020l The
 RQM
 Adapter
 has
 stopped.

RPTX2021E Unexpected
 error
 occurred
 while
 executing
 RQM
 test
 script.

RPTX2022E

Unexpected
error
occurred
while
processing
an
import
request
from
RQM.

RPTX2023W

Error
occurred
while
update
the
run
status
back
to
the
RQM
server.
This
may
cause
the
RQM
test
progress
page
to
contain
inaccurate
data.

RPTX2024E

Unable
to
attach
the
following
file
to
the
RQM
results.
This
may
cause
the
attached
HTML
report
not
to
render
correctly.
File
name:
%1

RPTX2025E

Error
occurred
while
registering
the
adapter:
%1.

RPTX2026E

Error
occurred
setting
the
default
adapter
name.
Please
set
the
name
in
the
Eclipse
Quality
Adapter
preference
page
(Windows-
>Preferences).

RPTX2027W

Multiple
test
runs
were
detected
when
the
stop
request
was
received
from
RQM.

RPTX2029W

Was
unable
to
perform
stop
request
from
RQM.
Likely
the
run
was
already
shutting
down
when
the
request
came
in.

RPTX2030I

Request
to
stop
the
test
is
being
delayed
until
the
appropriate
run
state
is
reached.

RPTX2031

A
request
to
stop
the
currently
running
test
has
been
received
by
RQM.

RPTX2032

Successfully
issue
a
stop
command
to
the
running
test.
Please
wait
for
the
test
to
end.

RPTX2033E

Error
attempting
to
stop
a
test.

RPTX2034E

Unable
to
create
directory
%1
no
further
information.
Ensure
user
has
permission
to
create
directory
in
that
location.

RPTX2035E

Error
occurred
while
attempting
to
automatically
update
pre-8.0
asset
%1
for
RQM
execution.

RPTX2036E

RQM
remote
resource
access
is
not
checkmark_perf.jpg
for
pre-8.0
SOA
assets.
Please
update
your
entire
SOA
project
to
8.0
or
greater
before
sharing.

RPTX2037E

Launch
was
aborted:
%1

RPTX2050E

Unable
to
download
remote
asset
%1
into
local
workspace.
Remote
repository
%2.
Ensure
RQM
system
has
connectivity
to
the
remote
repository
and
the
file
exists.

RPTX2051E

Unable
to
browse
%1
in
remote
repository
%2.
Ensure
RQM
system
has
connectivity
to
the
remote
repository
and
the
directory
exists.

RPTX2055E

Error
occurred
reading
the
adapter
connection
file.

RPTX2056E

Error
occurred
saving
the
adapter
connection
file.

RPTX2057E

Unable
to
complete
import
operation
because
the
specified
path
is
not
in
the
adapters
workspace.
Try
specifying
the
only
the
project
name.

RPTX2058E

The
specified
script
<
%1>
is
not
in
the
workspace
currently
being
used
by
the
adapter.
You
can
only
execute
scripts
which
are
in
the
adapters
workspace.

RPTX2060E

The script path specified by RQM does not seem to be valid. Please ensure the RQM test script has a script path which contains the project and script name.

RPTX2061W

Run
verdict
is
inconclusive
because
no
performance
requirements
exist
in
the
last
user
stage
for
the
associated
VU
Schedule.

RPTX2062W

Run
verdict
is
inconclusive
because
there
are
zero
performance
requirements
in
the
last
user
stage
of
the
associated
VU
Schedule.

RPTX2063W

No
time
range
was
generated
for
the
user
stage
of
the
associated
VU
Schedule.
Performance
requirements
reported
to
RQM
will
be
based
on
the
default
time
range.

RPTX2070E

Error occurred while setting the RQM project area. Make sure a valid project area is specified on the Quality Manager Adapter preference page. The adapter is attempting to connect to RQM using the default project area.

RPTX2071E

Error occurred while retrieving list of project areas. Please verify Quality Manager connection information. See error log for more details.

RPTX2072E

Error occurred calling for the web analytics dashboard link.

Explanation: Rational® Performance Tester could not open the external URL for the dashboard that references Rational Quality Manager records. This error occurs when a problem exists with the classpath for the result analysis.

User response: Ensure that Rational Quality Manager is version 4.0 or later and Rational® Performance Tester is version 8.3 or later. If the error log contains startup errors, resolve the errors and check whether the problem is resolved.

RPTX2073E

Error
occurred
while
translating
RQM
server
execution
variables
to
Rational®
Performance
Tester.

Explanation: An unexpected error occurred while setting up execution variables.

System action: Execution variables are unavailable during execution.

User response: Contact support if this error persists.

RPTX2074E

Error
connecting
RTW
adapter
and
successful
connecting
RPT
adapter.
This
suggest
RQM
does
not
support
RTW
script
type
introduced
in
4.0.3.
If
RTW
adapter
is
not
required
it
can
get
disabled
by
adding
-DrtwStartAdapter=false
in
eclipse.ini.

Explanation: Error connecting RTW adapter and successful connecting RPT adapter. This suggest RQM does not support RTW script type introduced in 4.0.3. If RTW adapter is not required it can get disabled by adding -DrtwStartAdapter=false in eclipse.ini.

User response: Use a RQM system supporting RTW script type. Add -DrtwStartAdapter=false in eclipse.ini to disable RTW script type. To import and execute RTW assets under the RPT adapter script type also add -DrptAvoidRQMImportFiltering in eclipse.ini

RPTX2075E

Unable
to
interpret
expression
<
%1>
from
RQM
control
file
<
%2>.
Ignoring.
Reason
<
%3>

Explanation: The RQM control file is of an invalid format.

System action: The control file instruction will be ignored.

User response: Change the file so it follows the specified format supplied by support.

RPTX2077E

Unable
to
browse
to
<
%1>.
Make
sure
it
exists
on
the
shared
location.

Explanation: The project referenced does not exist on the shared location.

System action: The RQM execution will stop.

User response: Ensure all required projects exist on the shared location.

RPTX2079W

Errors
attempting
to
load
available
SmartCard
aliases.
See
documentation
on
how
to
configure
your
system
to
use
SmartCard
to
authenticate
to
Quality
Manager.

Explanation: An error occurred while attempting to load SmartCard aliases.

System action: User is unable to configure SmartCard using the preference UI.

User response: Ensure com.ibm.security.capi.IBMCAC is listed as provider 1 in file *{install}\SDP\jdk\jre\lib\security\java.security*. See documentation on manual steps required to configure adapter for SmartCard usage. Contact support if issues persist.

RPWF0011E

Error
occurred
while
completing
test
generation

RPWF0012E

Error
occurred
while
processing
a
packet
at
test
generation

RPWF0021E

WSDL
Exception
raised
while
processing
WSDL
source

RPWF0032E

Error
while
generating
test
from
Axis
recording

RPWF0051E

Error
occurred
while
setting
classpath
entry
for
recorder

RPWF0052E

I/O
exception
occurred
while
resolving
keystore
or
truststore
path

RPWF0056E

Error
occurred
while
launching
web
services
HTTP
proxy

RPWF0066E

Error
occurred
while
launching
axis
client
recorder
agent

RPWF0071E

Exception
thrown
while
creating
a
wizard
page
control

RPWF0072E

Exception
thrown
while
parsing
URL:
%1

RPWF0074E

Exception
thrown
while
finishing
the
axis
recording
wizard

RPWF0075E

Exception
thrown
while
looking
for
an
available
port

RPWF0076W

Exception
thrown
while
adding
SOA
Tester
certificate
to
the
trustore
%1

RPWF0081W

A
proxy
authorization
%1
is
used
without
any
proxy

RPWF0082W

No
free
name
can
be
found;
reusing
%1

RPWF0083E

Resource
file
%1
not
found
in
workspace
%2

RPWF0084E

Workspace
location
cannot
be
determined

RPWF0085E

Cannot
retrieve
the
operation
name
from
the
envelope
%1

RPWF0101E

Core
exception
thrown
using
org.eclipse.debug.core
plugin

RPWF0102E

Exception
thrown
during
launch
configuration
update

RPWF0103E

Exception
thrown
while
resolving
a
bundle
entry
path

RPWF0104E

Exception
thrown
while
identifying
localhost
IP
address

RPWF0111E

Exception
thrown
while
creating
a
substitution:
%1

RPWF0112E

Exception
thrown
while
creating
a
reference:
%1

RPWF0121W

Unknown
format.
Skipping
the
test
generation
for:
%1

RPWF0122W

Skipped
call:
%1

RPWF0123W

Skipped
request:
It
could
be
that
provided
password
was
not
ok

RPWF0124W

Attachments
not
generated.

RPWF0130W

Could
not
find
project
for
URI:
%1

RPWF0131W

Loading
XSD
Schema
failed:
%1

RPWF0132E

Error
while
generating
test
from
Generic
Service
Client:
Can't
show
wizard

RPWF0140E

An
error
has
occurred:
%1

RPWH0007W

Unhandled
Security
Algorithm
'%1'

RPWH0009W

Unable
to
serialize
data

RPWH0010W

Unable
to
deserialize
data

RPWH0012E

Unable
to
open
editor
for
'%1'

RPWH0014E

Parse
Error
in
'%1'

RPWH0015E
Unable
to
create
resource
'%1'

RPWH0016E
Failed
to
export
source
text
'%1'

RPWH0017E
A
connection
error
occurred
on
'%1',
please
check
the
URL
or
the
network
configuration

Explanation: A connection error occurred.

System action: URL can not be reached, action is aborted.

User response: Check the URL or the network configuration.

RPWS0001E

Exception
raised
during
data
harvest
execution

Explanation: Reference can't be performed.

System action: Reference is not performed: get empty data.

User response: Check the corresponding reference.

RPWS0002E

Exception
raised
during
data
substitution
execution

Explanation: Substitution can't be performed.

System action: Substitution is not performed: write recorded data.

User response: Check the corresponding substitution.

RPWS0003E

Exception
raised
on
harvest
data
management

Explanation: Reference can't be performed.

System action: Reference is not performed: get empty data.

User response: Check the corresponding reference.

RPWS0004E

Exception
raised
on
substitution
data
management

Explanation: Substitution can't be performed.

System action: Substitution is not performed: write recorded data.

User response: Check the corresponding substitution.

RPWS0005E

Exception
raised
during
WebSocket
read
action

Explanation: Read action can't be performed.

System action: No data are receive.

User response: Check the application side, may be the server closes the connection.

RPWS0006E

Exception
raised
during
WebSocket
write
action

Explanation: Write action can't be performed.

System action: No data are sent.

User response: Check the application side, may be the server closes the connection.

RPWS0007E

Unable
to
get
WebSocket
connection

Explanation: WebSocket connection information is wrong.

System action: No data will be receive or sent on this connection.

User response: Check the WebSocket connection, may be the test is corrupted.

RPWS0008E

Unable
to
read
from
a
closed
connection

Explanation: WebSocket connection is closed.

System action: No data will be sent on this connection.

User response: Check why the WebSocket server closed connection.

RPWY0002E

An
exception
occurred
in
%1

Explanation: An exception was detected.

System action: Current action is aborted.

User response: Check the cause of the exception.

RPWY0003I

Information:
%1
(%2)

RPWY0004W

Warning:
%1
(%2)

RPWY0005E

An
error
occurred
while
importing
external
schema
%1

RPWY0006E

Unable
to
correlate
automatically

RPWY0007E

An
exception
%1
occurred
in
%2

Explanation: An exception was detected.

System action: Current action is aborted.

User response: Check the cause of the exception.

RPWZl0002E

Exception
raised
during
WebSocket
connection
creation.

Explanation: The workbench could not create a connection for WebSocket elements in split test.

System action: No connection are created by the workbench.

User response: The user need to create manually the connection, or to get the upgraded HTTP request in the split selection.

RPXD0001E

Unknown
Segment
Offset/
Length
for
Segmented
Dataset:
%1

RPXD0002E

Bad
Dataset
Mode:
%1

RPXD0003E

Dataset
not
initialized:
%1

RPXD0004E

End
of
non-
wrapped
dataset
reached:
%1

RPXD0005E

Dataset
with
multiple
Equivalence
Classes
cannot
be
segmented

RPXD0006E

segmented
DatapoolMap
null:
%1

RPXD0007F

No
registered
data
correlation
handler
for
this
IKAction

RPXD0017W

Pattern
matching
failed
for:
regex
(%1)
str
(%2)

RPXD0018E

Skipping
substitution,
reference
value
was
null.
original
string:
(%1)
offset:
(%2)

RPXD0019E

Data
Correlation:
Failed
Substitution
\nReference[%1]\nSubstitution[%2]\n
\nDetails:
\n
\nA
failed
reference
occurred
in
a
prior
request.
Since
the
reference
named
[%3]
was
null,
we
were
unable
to
substitute
a
new
value
for
the
substituter
named
[%4],
original
string
[%5]
at
offset
[%6]
and
this
request
may
have

Explanation: A reference for an expected data substitution is null.

System action: None.

User response: To find the failed reference, open the test and go to the substitution site. Right-click the substitution site and select Go To > Reference. When troubleshooting failed references, start with the first error message. The first failed reference can cause subsequent failed references. Search the test log "for unable to extract" to find the first error message.

Examine the request that generated the response. The request contains a value that might need to be correlated. For example, the request might contain a username that must be unique to play back the test successfully. In that case, use a dataset to provide a list of unique username values. You might need to manually correlate a value by using the Test Data Sources view. Values that typically are correlated include timestamps, dates, ids, and other alphanumeric strings.

If you no longer need the data correlation mentioned in the message, remove that data correlation from the test.

RPXD0020E

Data
Correlation:
Failed
Extraction
\nReference[%1]\n
\nDetails:
\n
\nWe
were
unable
to
extract
the
value
for
the
reference
named
[%2],
with
the
regular
expression
[%3].
This
could
mean
a
later
request
will
fail.
Please
compare
the
response
in
the
test
log
to
the
corresponding
response
in
the

Explanation: The response received during playback is different from the response received when the test was recorded. The data correlation code was unable to use the regular expression expected value.

System action: None.

User response: Examine the request that generated the response. The request contains a value that might need to be correlated. For example, the request might contain a username that must be unique to play back the test successfully. In that case, use a dataset to provide a list of unique username values. You might need to manually correlate a value by using the Test Data Sources view. Values that typically are correlated include timestamps, dates, ids, and other alphanumeric strings. \nIf you no longer need the data correlation mentioned in the message, remove that data correlation from the test.

RPXD0021E

Dataset
%1
is
accessed
using
different
dataset
modes
by
different
tests.

RPXD0021W

Setting
variable
%1
to
value
%2.

RPXE0001W

%1

RPXE0010W

Engine
shutdown
problem
joining
workers

RPXE0011W

Failed
to
report
exception

RPXE0012W

Schedule
failed
to
load

RPXE0013W

Unable
to
create
test

RPXE0014W

Setting
log
level
to
%1

RPXE0015W

Attempt
to
add
object
to
Schedule
which
is
not
a
UserGroup

RPXE0016W

Virtual
User
%1
experienced
error
%2

RPXE0017W

Connect
timeout
for
action
%1
(%2)
user
%3

RPXE0018W

Read
timeout
for
action
%1
(%2)
user
%3

RPXE0019W

Connect
exception
for
action
%1
(%2)
user
%3

RPXE0021W

Read
exception
for
action
%1
(%2)
user
%3

RPXE0023W

Iterating
over
keys
exception

RPXE0024W

CancelledKeyException

RPXE0025W NullPointerException

RPXE0027W UserGroup
exception

RPXE0028W User
Group
%1
does
not
implement
createTesterWorkload()

RPXE0029W Worker
caught
throwable

RPXE0030W Connection
leak,
I/O
state
%1

RPXE0031W Exception
finishing
connection
for
action
%1
(%2)
user
%3

RPXE0033W

Finish
read
get
buffer
interrupted
for
action
%1
(%2)
user
%3

RPXE0035W

Finish
read
exception
for
action
%1
(%2)
user
%3

RPXE0036W

Engine
thread
startup
exception

RPXE0037W

Engine
request
to
report
exception

RPXE0038W

Exception
creating
cache
file,
cacheFileName:
%1,
extension:
%2,
dir:
%3

RPXE0039W

User
%1
experienced
exception
%2

RPXE0040W

User
%1
caught
exception
trying
to
report
severe
error.

RPXE0041W

Engine
hard
stop
after
%1
second
timeout

RPXE0042I

%1
received
request
to
stop

RPXE0043I

Forced
stop
of
action
%1

RPXE0044W

No
IP
address
was
found
for
the
local
host

RPXE0045W

Ignoring
invalid
network
interface
%1

RPXE0046W

Could
not
find
any
usable
network
interfaces

RPXE0047E

SyncPointSubsystem
Unknown
sync
point:
%1

RPXE0048W

%1
STOPUSERS
users=
%2
stagger=
%3
timelimit=
%4
active
users=
%5

RPXE0049W

%1
had
%2
non-
sampled
users
asked
to
stop
active
users=
%3

RPXE0050W

%1
had
%2
sampled
users
asked
to
stop

RPXE0051W

%1
after
wait
for
compliance
active
users=
%2
target=
%3

RPXE0052W

%1
abandon
user
%2

RPXE0053W

%1
abandoned
%2
users

RPXE0054W

%1
end
stop
%2
users
SUCCESS
active
users=
%3

RPXE0055W

%1
end
stop
%2
users
FAIL
active
users=
%3

RPXE0056W

%1
occurred
in
%2.
Message:
%3

RPXE0057E

Exception
while
reading
test
variable
initialization
file:
%1

RPXE0058E

Exception
while
initializing
virtual
users
test
variables.

RPXE0059E

Unable
to
get
Kerberos
ticket
from
KDC
for
server
%1.

RPXE0060E

Failed
to
load
test
from
'%1'
due
to
exception:
%2

Explanation: While trying to find and load class files required to execute the test a problem was encountered.

User response: See exception description for failure reason.

RPXE0100W

%1
terminated
due
to
exception:
%2

RPXE0102W

IKAction:
%1
(%2)
caught
Exception
in
preFinish()
for
%3
(%4)

RPXE0103W

IKAction:
%1
(%2)
caught
Exception
in
postFinish()
for
%3
(%4)

RPXE0104W

KernelChannel
connect(),
exception
while
trying
to
bind
to
local
address
%1:
%2

RPXE2501E

An
error
occurred
while
attempting
to
handshake
with
the
server
using
protocol
%1
and
cipher
suite
%2.
This
type
of
failure
is
often
related
to
a
mismatch
between
the
requested
protocol
or
cipher
suite
and
the
ones
the
server
is
expecting
or
may
be
related
to

Explanation: An SSL connection between a client and server is set up by a handshake, the goals of which are: To satisfy the client that it is talking to the right server (and optionally visa versa). Also, for the parties to have agreed on a cipher suite, which includes which encryption algorithm they will use to exchange data. These goals were not achieved.

System action: Execution ends because a secure connection cannot be established with the server.

User response: If the server requires a client digital certificate work with the server administrator to obtain one. If the server requires strong ciphers work with customer support to obtain the required and restricted ciphers.

RPXE2550E

The
digital
certificate
RCS
file
'%1'
was
not
found
or
was
corrupt:
%2

RPXE2552I

digital
certificate
alias

RPXE2900E

The
server
rejected
the
client's
digital
certificate.

RPXE2901W

The
server
closed
the
connection
abruptly.
This
is
probably
due
to
an
overloaded
server
or
to
a
problem
negotiating
a
digital
certificate
or
cipher
suite.
Check
the
web
server's
SSL
error
log
for
more
details.

RPXE4000W

Schedule
or
Test
not
found.
May
not
have
compiled.
-
%1

RPXE4001E

Runner
Exception
occurred

RPXE4002E

Communications
Error:
Invalid
Logging
Level

RPXE4003E

Communications
Error:
Invalid
TestLog
Level
for
%1
events

RPXE4004E

Communications
Error:
Invalid
Statistics
Level
or
Interval

RPXE4005E

Runner
Exception
occurred
-
See
problem
determination
log

RPXE4006E

Communications
Error:
Invalid
Dataset
information

RPXE4007E

Communications
Error:
No
communication
from
the
workbench
in
%1
milliseconds.
For
more
information,
see
the
Troubleshooting
section
of
the
online
help.

RPXE4008E

Attempt
to
change
statistic
interval
ignored.

RPXE4008l

Think:
requested
time
%1
milliseconds,
actual
time
%2
milliseconds

RPXE4009l

Delay:
requested
time
%1
milliseconds,
actual
time
%2
milliseconds

RPXE4010I

Schedule
completed.
See
Performance
Report,
Verification
Points
Report,
and/
or
Percentile
Report
to
further
evaluate
the
results
of
this
run
according
to
your
success
criteria.

RPXE4011E

Communications
Error:
Invalid
Stop
timeout

RPXE4013I

Additional
events
from
%1

RPXE4014E

Communications
Error:
Invalid
RunStagger
information
for
%1
(pairCount)

RPXE4015E

Communications
Error:
Invalid
RunStagger
information
for
%1
(pair
%2)

RPXE4016E

Failed
to
start
users
for
user
group:
%1.

RPXE4017I

Additional
execution
history
events
from
%1
are
available,
but
they
have
been
stored
separately
upon
user
request.
See
file
%2.
Refer
to
the
most
current
version
of
the
product
release
notes
for
information
on
how
to
access
and
view
them.

RPXE4018E

Failed
to
write
message
to
workbench
[%1]

RPXE4019E

Failed
to
remove
users
for
user
group:
%1.

RPXE4020E

Failed
to
add
users
for
userGroup:
%1
numUsers=[%2]
startId=[%3]

RPXE4021E

Failed
to
add
users
because
the
runner
is
not
in
a
runnable
state.

RPXE4022E

failed
to
add
desired
number
of
users

RPXE4023E

failed
to
reach
target
number
of
users
ramping
down

RPXE4024E

not
runnable
or
command
failed

RPXE4025E

failed
to
set
the
DataView
state
of
user
%1[%2]
to
%3.

RPXE4026E

DataView
command
%1
is
not
yet
implemented.

RPXE4027E

DataView
command
%1
is
not
recognized.

RPXE4028E

MessageEventFilter
command
parsing
error
in
token
%1[%2]
of
command
[%3]

RPXE4029E

The
testLog
message
event
filter
specified
by
[%1]
cannot
be
constructed.
This
filter
element
will
be
ignored.

RPXE4050I

Operating
System
Info:
name
[%1]
architecture
[%2]
version
[%3]

RPXE4100W

Cannot
open
execution
history
cache
file
[%1],
execution
history
will
not
be
cached

RPXE4101E

Error
closing
execution
history
cache
file
[%1]

RPXE4102E

Error
reading
%1
bytes
from
execution
history
cache
file
[%1]

RPXE4103E

Error
writing
%1
bytes
to
execution
history
cache
file
[%2]

RPXE4104E

Error
opening
execution
history
cache
file
[%1]
for
reading

RPXE4105E

Error
testing
execution
history
cache
file
[%1]
for
available
input

RPXE4106E

Unexpected
EOF
reading
%1
bytes
from
execution
history
cache
file
[%2]

RPXE4107E

Exception
processing
execution
history
event

RPXE4108E

Error
reading
execution
history
cache
file
[%1]

RPXE4109E

Error
writing
to
TestLog
cache
file
[%1]

RPXE4110E

Error
closing
TestLog
cache
file
[%1]

RPXE4111W

Cannot
open
testLog
cache
file
[%1]
for
random
access
writing,
the
testLog
may
contain
bad
data.

RPXE4112W

Error
removing
testLog
event
from
cache
file
[%1].
Writing
%2
bytes
at
offset
%3.

RPXE4120E

Error
writing
to
TestLog
[%1]

RPXE4150E

Error
opening
execution
history
annotation
file
[%1]

RPXE4151E

Error
writing
%1
bytes
to
execution
history
annotation
file
[%2]

RPXE4152E

Error
flushing/
closing
history
annotation
file
[%1]

RPXE4153E

Error
deleting
history
annotation
file
[%1]

RPXE4200W

Warning:
Statistics
delivery
thread
running
behind
statistics
interval
by
%1
milliseconds

RPXE4201W

Warning:
Statistics
delivery
thread
over
slept
by
%1
milliseconds

RPXE4202E

Error:
Statistics
delivery
thread
over
slept
by
%1
milliseconds

RPXE4203E

Error:
Statistics
collection
time
too
long:
%1
bytes
%2
milliseconds

RPXE4204W

Warning:
Statistics
collection
time
too
long:
%1
bytes
%2
milliseconds

RPXE4205E

Error:
Statistics
write
time
too
long:
%1
bytes
%2
milliseconds

RPXE4208E

Error:
Could
not
create
agent
measurements
file
%1.

Explanation: It is not possible to create a file on the file system.

System action: Unable to create a file. The agent measurements will not be available.

User response: You do not have the permissions on your file system or it is full.

RPXE4209I

Error:
Statistics
collection
thread
was
interrupted

Explanation: An error occurred which caused the interruption of the statistics collection.

System action: Statistics may be incomplete.

User response: Run the test again.

RPXE4210E

Error:
A
severe
error
occurred
when
processing
statistics.

Explanation: An exception occurred on the agent while processing statistics and/or sending them to the server.

System action: Statistics will be incomplete.

User response: Contact support.

RPXE4211E

Error:
A
severe
error
occurred
when
sending
statistics.

Explanation: An exception occurred on the agent while sending statistics to the server.

System action: Statistics will be incomplete.

User response: Contact support.

RPXE4212E

Error:
A
severe
error
occurred
when
closing
statistics.

Explanation: An exception occurred on the agent while completing the statistics processing.

System action: Statistics may be incomplete.

User response: Contact support.

RPXE4213E Statistics
 sub-
 system
 error:
 %1

Explanation: A severe error occurred during writing to the agent measurements file.

System action: The agent detailed measurements will not be available.

User response: Start a test execution again.

RPXE4214W Statistics
 sub-
 system
 warning:
 %1

Explanation: Warning message to the user during writing to the agent measurements file.

System action: The agent detailed measurements may be affected by a problem.

User response: Fix the problem given by the message or contact support.

RPXE4215E Statistical
 counter
 descriptors
 file
 not
 found:
 %1.

Explanation: Unable to find the counter descriptors file in the deployment directories.

System action: Statistics will not be available.

User response: Start the test again, contact support if the problem persists.

RPXE4215I

Statistics
sub-
system
message:
%1

Explanation: This message is displayed in debug mode.

System action: No action.

User response: You can report this message to the support.

RPXE4216E

Problem
in
statistical
counter
descriptors
file:
%1.

Explanation: The counter descriptors file has a problem.

System action: Statistics will not be available.

User response: Start the test again, contact support if the problem persists.

RPXE4217E

Submitted
value
%1
is
out
of
range
of
allowed
values
for
the
counter
type
%2.

Explanation: The value is out of the limits of the counter.

System action: Measurements and statistics for the specified counter will not be available.

User response: If you are a protocol developer, fix the problem. Otherwise, contact support.

RPXE4218E

In
order
to
use
this
method,
the
runtime
type
of
the
counter
must
be
either
STATIC
or
RATE.

Explanation: A protocol is using a legacy API to change the value of a counter.

System action: Measurements and statistics for the specified counter will be inaccurate.

User response: If you are a protocol developer, use a runtime counter type to STATIC or RATE. Otherwise, contact support.

RPXE4219E

Mismatch
between
runtime
type
%1
and
static
counter
type
%2.

Explanation: The type of the counter in runtime and in the statistic definition do not match.

System action: The runtime type will be applied.

User response: If you are a protocol developer, change the declared counter type, or the runtime type. Otherwise, contact support.

RPXE4220E

No
static
declaration
found
for
counter
%1.

Explanation: Unable to find a definition for the counter.

System action: The counter values will be ignored.

User response: Add a definition for the counter, or use an undeclared counter.

RPXE4221E

Attempt
to
create
an
undeclared
counter
%1
(type
%2)
over
a
declared
counter
of
a
different
type
(%3).

Explanation: An attempt to create an undeclared counter was made, but a counter declaration with another type already exists.

System action: The undeclared counter values will be ignored.

User response: Use another path for the undeclared counter that does not conflict with the existing declared counter.

RPXE4900I Test
 execution
 completed
 with
 no
 reported
 problems

RPXE4901I %1
 ERROR
 verdicts
 reported

RPXE4902I %1
 FAIL
 verdicts
 reported

RPXE4903I %1
 INCONCLUSIVE
 verdicts
 reported

RPXE4904I All
 reported
 verdicts
 PASSed

RPXE4905l %1
ERROR
verdict
reported

RPXE4906l %1
FAIL
verdict
reported

RPXE4907l %1
INCONCLUSIVE
verdict
reported

RPXE4908l %1
FAIL
verdict
roll-
up

RPXE4909l %1
ERROR
verdict
roll-
up

RPXE4910l %1
INCONCLUSIVE
verdict
roll-
up

RPXE4911I

%1
PASS
verdict
roll-
up

RPXE4912I

%1
ERROR
verdicts
reported
from
driver
%2

RPXE4913I

%1
FAIL
verdicts
reported
from
driver
%2

RPXE4914I

%1
INCONCLUSIVE
verdicts
reported
from
driver
%2

RPXE4915I

%1
ERROR
verdict
reported
from
driver
%2

RPXE4916I

%1
FAIL
verdict
reported
from
driver
%2

RPXE4917I

%1
INCONCLUSIVE
verdict
reported
from
driver
%2

RPXE4918I

duration

RPXE4920I

%1
was
successfully
invoked.
This
does
not
indicate
the
pass/
fail
verdict
of
the
test
itself,
only
that
the
invocation
of
the
test
was
successful.
Expand
to
inspect
verdicts.

RPXE4921I

%1
was
invoked.
This
does
not
indicate
the
pass/
fail
verdict
of
the
test
itself,
only
that
the
invocation
of
the
test
was
successful.
No
verdicts
will
be
reported
from
the
test.

RPXE4930I

The
%1
testLog
level
was
pushed
from
%2%3
to
%4%5.

RPXE4931I

The
%1
testLog
level
was
popped
from
%2%3
to
%4%5.

RPXE4932I

The
%1
testLog
level
was
changed
from
%2%3
to
%4%5.

RPXE4940I

Transaction
[%1]
started
%2
milliseconds
after
start
of
test
run.

RPXE4941I

Transaction
[%1]
stopped
%2
milliseconds
after
start
of
test
run.
Elapsed
time:
%3
milliseconds.

RPXE4942I

Transaction
[%1]
aborted.

RPXE4944W

Transaction
[%1]
is
already
started.

RPXE4945W

Transaction
[%1]
has
not
been
started.

RPXE4948W

Execution
Variables
-
Input

RPXE4950I

Null
user
group
name.

RPXE4952E

Unable
to
find
target
loop
named
'%1'.
Error
handler
did
not
complete
properly.

Explanation: The loop name specified in the loop handler does not exist.

System action: The user will not follow the loop error handler and will continue execution at the next action.

User response: Change the loop handler to point to an existing loop.

RPXE5301E

Error
encountered
while
loading
Native
Library:
%1

RPXE5305E

A
required
customer-
supplied
file
was
not
found.
Please
check
the
"external_files"
folder
and
your
installation
instructions
for:
%1

RPXE5330E

Unable
to
apply
dataset
swap:
%1

Explanation: An error occurred attempting to parse the data set swap command-line option.

System action: The data set swap will not occur.

User response: See the command-line usage to ensure the command syntax is correct.

RPXE5500W
Unable
to
apply
Open
Tracing
context.
The
root
Jaeger
span
will
be
unparented.
%1

Explanation: An error occurred when attempting to create an Open Tracing span context from the properties starting with OPENTRACING_CTX_.

System action: Jaeger logging will still occur but the root span will be linked to a parent span.

User response: Make sure the content of properties starting with OPENTRACING_CTX_ is correct.

RPXE5501W

Transaction
times
for
this
run
do
not
include
failing
transactions,
according
to
workbench
Test
Execution
preferences.

Explanation: A failing transaction will not be added to stats. This will only be logged once per transaction, but multiple instances may have failed.

System action: Execution will continue as normal. This is not an error condition.

User response: If this behavior is not desired, uncheck preferences at Test > Test Execution.

RRIT0001E

Environment
variable
INTEGRATION_TESTER_AGENT_HOME
not
set
to
Rational@
Integration
Tester
Agent
installation
location,
or
does
not
contain
expected
RunTests(.exe)
program.

Explanation: The test execution cannot find the Rational@ Integration Tester Agent.

System action: None.

User response: Set the environment variable INTEGRATION_TESTER_AGENT_HOME to point the root installation directory of the Rational@ Integration Tester agent. This must be done on each location used in a schedule.

RRIT0002E

Error
unzipping
__IT_PRODUCT_NAME__
project.

Explanation: The __IT_PRODUCT_NAME__ project cannot be deployed.

System action: None.

User response: Verify that there is enough disk space on the executing location.

RRIT0003E

__IT_PRODUCT_NAME__
library
not
found.

Explanation: The library required to communicate with the __IT_PRODUCT_NAME__ agent is missing in the installation.

System action: None.

User response: Contact your support.

RRIT0004E

Error
processing
messages
received
form
the
__IT_PRODUCT_NAME__
client.

Explanation: A communication error has occurred with the __IT_PRODUCT_NAME__ agent.

System action: None.

User response: Try again, contact your support if problem persist.

RRIT0005E

Some
tag
values
are
missing.

Explanation: A value cannot be assigned to a tag defined in an __IT_PRODUCT_NAME__ test during execution.

System action: None.

User response: Verify that each tag of each __IT_PRODUCT_NAME__ test maps to a variable in the schedule or compound test.

RRIT0006E

Interrupted
RunTests(.exe)
start
sequence

Explanation: When Rational Integration Tester, an unexpected interrupt exception happened.

System action: None

User response: Try again

RRIT0007E

Test
execution
can
not
continue

Explanation: The Rational Integration Tester test execution can not continue. Maybe because Rational Integration Tester is not installed on the machine, maybe because the communication channel could not be established.

System action: None.

User response: Verify that Rational Integration Tester is installed and try again.

RRITU1002W

Open
__IT_PRODUCT_NAME__
resources
has
been
disabled
in
Test
>
__IT_PRODUCT_ACRONYM__
Integration
preferences

Explanation: User as disabled __IT_PRODUCT_NAME__ resource but want to open this kind of resource.

System action: None.

User response: Open Test > __IT_PRODUCT_ACRONYM__ Integration preference and enable open resources by checking __IT_PRODUCT_NAME__ is installed on this machine .

DCRC0001E

Missing message for log entry '{0}' in class: {1}

DCRC0002E

Cannot
get
Log
key
'{0}':
SecurityException
raised

DCRC0003E

Cannot
initialize
Log
key
'{0}'

DCRC0008W

Warning:
field
'{0}'
is
not
defined
in
class:
{1}

DCRC0009W

Warning:
cannot
get
check
message
versus
log
key
mapping
for
'{0}'
of
class
{1},
SecurityException
raised

DCRC0010E

Unexpected
exception,
please
check
Error
Log
view:
{0}

DCUI0001E

unexpected
exception

Explanation: An exception that could not be handled occurs during processing.

User response: Close rule editor and report exception to product support.

DCUI0003E

Error
getting
persistent
property
'{0}'

DCUI0004E

Error
setting
persistent
property
'{0}'

DCUI0006E

Cannot
reload
resource
'{0}'

DCUI0007W

Failed
to
encode
model
to
clipboard.

DCUI0008W

Failed
to
decode
model
from
clipboard.

DCUI0009E

None
of
the
attribute
providers
own
attribute
id
'{0}'.

Explanation: Rule file refer to an unknown rule attribute id. File may be edited on a system having more protocol extension rather than current one.

User response: Rule file should not be edited on this product installation.

DCUI0010E

Missing
IRuleUIProvider
extension
point
for
'{0}'

Explanation: Rule file refer to a rule that is unknown on this product installation.

User response: Rule editor is able to display that rule on the tree but not able to edit it contents.

DCUI0011E

Missing
IConditionUIProvider
extension
point
for
'{0}'

Explanation: Rule file refer to a rule condition that is unknown on this product installation.

User response: Rule editor is able to display that rule condition on the tree but not able to edit it contents.

DCUI0012E
Cannot
save
editor
'{0}'

DCUI0013E
Missing
IRulePassUIProvider
extension
point
for
'{0}'

Explanation: Rule file refer to a rule pass that is unknown on this product installation.

User response: Rule editor is able to display that rule pass on the tree but not able to edit it contents.

DCUI0014E
Missing
IRuleArgumentUIProvider
extension
point
for
'{0}'

DCUI0015E
Missing
IRuleArgumentContainerUIProvider
extension
point
for
'{0}'

DCUI0016E
Try
rule
failed

DCUI0017E
Try
rule
failed:
'{0}'

DCUI0998E
Cannot
load
file
'{0}'

Additional error messages

You can find the additional error messages.

Address already in use
Address
already
in
use.

Explanation: Typically, this error message is displayed when all available TCP/IP ports have been exhausted.

System action: None.

User response:

- If the schedule, in this context, is used to refer to VU Schedule and Rate Schedule contains loops, move the loops into tests. If a schedule, in this context, is used to refer to VU Schedule and Rate Schedule contains loops, at the beginning of a loop iteration each virtual user closes existing connections and opens new connections. This can cause the agent computer to exhaust all available TCP/IP ports. If a test contains loops, virtual users attempt to re-use existing connections. Re-use of existing connections can take advantage of keep-alive connections.
- Increase the number of TCP/IP ports available. The number of TCP/IP ports on a Windows™ computer is limited to 5000 by default. To increase the number of TCP/IP ports available on a Windows™ computer:
 1. Create the following Registry key: `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\MaxUserPort`. This key does not exist by default.
 2. Specify the type as `DWORD`.
 3. Set the value to `65000`.
 4. Reboot the computer.

Browser profile in use

Your
browser
profile
is
already
in
use.
You
must
close
the
browser
or
select
another
browser
from
Application
to
Record
in
Preferences.

Explanation: The browser configured for recording is already running, or a lock file exists that prevents the browser from running.

System action: The test run does not start.

User response:

- Close the browser configured for recording, and record again or select another web browser in the **Test From Recording** wizard.
- Search for and remove any lock files from the browser profile directory.

Cannot open test Cannot
open
the
test
testname
because
it
contains
the
following
features
that
are
not
installed:
featurename.

Explanation: The test that you are attempting to open requires the installation of one or more protocol extensions.

System action: The test is not opened.

User response: To install a protocol extension:

1. Close the product.
2. Launch IBM® Installation Manager.
3. Click **Modify**.
4. Navigate to the **Modify Packages** window.
5. Select the required protocol extensions.
6. Follow the steps in the wizard to complete the installation of the protocol extensions.

A protocol license key is required to use protocol extensions other than HTTP. See the installation guide for more information on managing protocol license keys.

Connection closed

Web
server
computername
unexpectedly
closed
the
connection
while
in
the
process
of
retrieving
URI
URIname

Explanation: This message is displayed when the HTTP server being tested has become too busy and has closed the connection without completing the response.

System action: None.

User response: If you receive this error message multiple times for a particular HTTP request or when the server is not under load, contact the administrator of the server under test to determine potential causes of the behavior.

Dataset accessed using different modes

Dataset
datapoolname
is
accessed
using
different
dataset
modes
by
different
tests.

Explanation: Two tests are accessing the same dataset. These two tests use different access methods for the dataset.

System action: None.

User response: Examine your tests to find the dataset mentioned. Check the access mode for the dataset by double-clicking the dataset name in the test. The access mode will be **Random**, **Shuffled**, or **Sequential**. Ensure that the access mode is the same for all tests that use the dataset.

Error binding to port

Error
Binding
to
Port:*portnumber*
Exception:
java.net.SocketTimeoutException:
Accept
timed
out.

Explanation: The listener port for HTTP recording is in use by another application.

System action: None.

User response: Change the listener port for HTTP recording to a port that is not in use by another application. See [Changing HTTP recording preferences on page 540](#) for more information. On the **Browsers Recording** preferences page, edit the value of the **Proxy Recorder Local Port** field.

No local agent controller

Local
computer
is
not
running
the
Agent
Controller.

Explanation: The agent controller coordinates schedule playback between the workbench and agent computers. The agent controller is not running on the local computer.

System action: No tests run.

User response: Make sure that the agent controller is installed and running on the workbench computer. On Windows™ computers, the agent controller is a process called ACWinService. On Linux™ computers, the agent controller is a process called RAServer.

To start the agent controller, do one of the following steps:

- On Windows™, at a command prompt, enter this command:

```
net start "IBM Rational Agent Controller"
```

- On Linux™, change to the directory `AgentController/bin` in the product installation directory; then enter this command:

```
./RAstart.sh
```

Performance Test Errors were found in the project

Performance

Test

Errors

were

found

in

the

project.

Explanation: This error can occur when workspaces are shared on different computers or when the installation location of the product has changed since the project was created.

System action: None.

User response: To work around this error, clean up the Java™ build path and the generated Java™ source files.

1. Click **Window > Open Perspective > Resource** to open the **Resource** perspective.
2. Select the project in the **Project Explorer** view, and then right-click and select **Properties** to view the project properties.
3. Select the **Java Build Path** property.
4. Click the **Libraries** tab.
5. Select each entry that is displayed with a red X and remove that entry from the build path. If a referenced library is pointing to user-specific libraries that have moved or are not present, edit the entry so that it points to the correct location. Do not remove entries for user-specific libraries.
6. Click **OK**.
7. Expand the `src` folder in the project.
8. Remove all generated Java™ source files that are displayed with a red X. If the source file is user-specific, then edit the build path to point to the correct dependent libraries.

Test run aborted

Execution
failure.
No
status
received
from
location
computername
in
interval
seconds.

Explanation: The workbench has lost communication with one or more agent computers.

System action: The test run stops.

User response: For each agent computer:

- Check that the playback Java™ process is running. If it is still running, the problem might be on the workbench computer. Stop the Java™ process and all related typeperf and vmstat processes.
- Examine the problem determination log for error messages or exception messages.
- If the playback Java™ process is not running, search for `javacore.*` files. The contents of these files might help you determine the cause of the problem.
- Run tests again and monitor the memory size of the playback Java™ process. If the playback Java™ process is consistently running at its maximum heap size, the process might not have enough memory.

If the problem is not on an agent computer, check that the workbench computer has sufficient memory. To increase the available memory, either increase the workbench heap size or reduce the level and amount of execution history.

Test run aborted due to error

Virtual
users
have
exited
prior
to
stage
completion.

Explanation: In a schedule with multiple stages, at the end of a stage that is configured with n virtual users, there were fewer than n virtual users running. The schedule did not assign enough work to the virtual users to keep them active for the duration of the stage, or the virtual users stopped because of an a different error. The workload problem can occur if you use multiple stages but do not enclose the workload in an infinite loop.

System action: The test run stops.

User response: On the workbench computer, examine the workload for each user group in the schedule. Check that the entire workload is inside an infinite loop, so that the virtual users always remain active regardless of the stage duration. If the tests already use infinite loops, then the agent computers might have encountered errors while running tests. To determine why virtual users stopped before the stage completed:

- Examine the test log for error or exception messages.
- Increase the test log level and decrease the number of users. Try running the schedule again. Examine the test log for error or exception messages.
- Simplify the workload by running one user in one stage. Check to see whether the single user takes the expected amount of time to complete all actions. If not, examine the test log and problem determination log for error or exception messages.

Testgen completed with warnings

Testgen
completed
with
warnings.

Explanation: The response data has been truncated according to the setting in the HTTP Test Generation preferences.

System action: None.

User response: To adjust how response data is truncated, click **Window > Preferences > Test > Test Generation > HTTP Test Generation > Test Generation Options**, and edit the value of the **Save only the first 4KB of responses larger than** field.

Variable not initialized

Variable
variablename
has
not
been
initialized
for
this
test.

Explanation: A variable was used in a test, but the variable had never been initialized to a value.

System action: Depending on the value of the **Run-time error if variable not initialized** setting, the system will either do nothing, issue a warning, issue a test log error, or exit the test.

User response: Check the variable to determine where the initialization should have happened. Make sure that the test that contains the initialization of the variable occurs before the test trying to use the variable.

Chapter 10. Reference Guide

This guide describes, additional topics to gain more knowledge about Rational® Service Tester for SOA Quality.

Accessibility features

Accessibility features help users who have physical disabilities, such as visual and, hearing impairment, or limited mobility, to use the software products successfully.

Accessibility features are product dependent and might include one or more of the following aspects:

- Keyboard-only operation
- Screen reader usage
- Color and typeface preferences



Note: The accessibility features mentioned here apply to the Windows operating system. Some of these features might also work on Linux, but are not officially checkmark_perf.jpg.

Accessibility compliance

To understand the accessibility compliance status for Rational® Service Tester for SOA Quality, refer to [Accessibility Conformance Reports](#).

For more information about IBM and accessibility, refer to [IBM Accessibility](#).

The product documentation is published by using Oxygen XML WebHelp Responsive. To understand the accessibility compliance status for Oxygen XML WebHelp Responsive, refer to [WebHelp Responsive VPAT Accessibility Conformance Report](#).

Accessing UI elements

Rational® Service Tester for SOA Quality supports navigation in the UI by using different methods such as a mouse, keyboard, or touchpad.

You can use the keyboard keys such as **Tab**, arrow keys such as **UP**, **DOWN**, **LEFT**, and **RIGHT** to navigate to the different pages in the **Navigation** pane or to the different action labels in the right pane on the UI.

Keyboard shortcuts for performance and service testing

The keyboard shortcuts for performance and service testing are available when you record or edit a test or a schedule.

Key combination	Description
Ctrl+S	Save the test or schedule.

Key combination	Description
Alt+Shift+T, G	Generate a test from the selected recording (.recmodel) file.
Alt+Shift+T, R	Create a report (the test must be selected in the Test Navigator).
Alt+Shift+T, T	Test connection (a location must be selected in the Test Navigator).
Alt+Shift+X, B	Run the test (a test must be selected in the Test Navigator).
Alt+Shift+X, C	Run the schedule (a schedule must be selected in the Test Navigator).
Del	Delete the selection
Ctrl+Del	Delete the selection
Insert	Insert a new element (same as the Insert push button).
Ctrl+Insert	Add a new element (same as the Add push button).
Ctrl+Up Arrow	Move the element up.
Ctrl+Down Arrow	Move the element down.
Ctrl+Alt+<, Ctrl+Alt+>	Resize the test editor and schedule editor windows. The new size is retained when you reopen the window.
Ctrl+Shift+F1	During HTTP recording, insert a comment.
Ctrl+Shift+F2	During HTTP recording, insert a screen capture.
Ctrl+Shift+F3	During HTTP recording, insert a synchronization point.
Ctrl+Shift+F4	During HTTP recording, start a transaction.
Ctrl+Shift+F5	During HTTP recording, end a transaction.
Ctrl+Shift+F6	During HTTP recording, insert a split point.
Ctrl+Shift+F7	During HTTP recording, set the name of the current page.

The following keyboard shortcuts are available when you record Citrix performance tests:

Key combination	Description
Tab or Shift+Tab	Cycle the focus through the UI elements
Arrows	Select a push button
Space	Click a push button or toggle between selections

When you record Citrix performance tests and you work in image synchronization mode, you can use these keys:

Key	Description
Space	Set the origin of selection area
Arrows	Move the cursor

Key	Description
Enter	Select the image synchronization area and set the synchronization area (press twice)
Esc	Cancel the selection

General reference for performance testing

See these performance testing topics for general reference.

Data correlation rules

You can customize how data is correlated by using data correlation rules.

Rules that create elements

Create a built in data source

Inserts a built-in data source in the test.

Create a custom code

Inserts a custom code element in the test.

Create a dataset column

Creates a dataset column that can be used by substitution sites.

Create a reference

Creates a reference in data that matches a specified regular expression.

Create a substitution

Creates a substitution site in data that matches a specified regular expression.

Create a variable assignment

Inserts a variable assignment in the test.

Create a variable declaration

Creates a variable that can be used by substitution sites.

Rules that change elements

Encode a substitution

Specifies whether substitution fields are encoded or decoded.

Rename a data source site

Changes the name of a data source that matches a regular expression.

Rename a substitution site

Changes the name of a substitution site that matches a regular expression.

Replace reference regular expression

Changes the regular expressions that are used by references.

Unlink a substitution

Removes the links between substitution sites and references and other data sources.

Rules that find elements**Find a reference**

Returns a reference. Add child conditions to specify the reference to find.

Find a substitution

Returns a substitution site. Add child conditions to specify the substitution site to find.

Find a variable

Returns a variable. Add child conditions to specify the variable to find.

Rules that remove elements**Remove a built in data source**

Deletes data sources from the test. Add child conditions to specify the data sources to delete. If you do not add child conditions, this rule deletes all data sources in the test.

Remove a custom code

Deletes custom code elements from the test. Add child conditions to specify the custom code elements to delete. If you do not add child conditions, this rule deletes all custom code elements in the test.

Remove a reference

Deletes references from the test. Add child conditions to specify the references to delete. If you do not add child conditions, this rule deletes all references in the test.

Remove a substitution

Deletes substitution sites from the test. Add child conditions to specify the substitution sites to delete. If you do not add child conditions, this rule deletes all substitution sites in the test.

Remove a variable assignment

Deletes variable assignments from the test. Add child conditions to specify the variable assignments to delete. If you do not add child conditions, this rule deletes all variable assignments in the test.

Remove a variable declaration

Deletes variables from the test. Add child conditions to specify the variables to delete. If you do not add child conditions, this rule deletes all variables in the test.

Error conditions

Error conditions include verification point failures, connection failures, server timeouts, custom code alerts, custom code exceptions, and problems with data correlation. You can specify an action to take when the error condition occurs. The Errors report displays the error conditions and error behavior that occurred in a test or schedule.

Page Title Verification Point Failure [HTTP]

The returned title for the primary request for an HTTP page does not match the expected title. The default value of the expected page title is what is returned between the <title></title> tags during recording. See [Specifying the expected page title](#) for more information.

Response Code Verification Failure [HTTP]

The returned response code does not match the expected response code. You can specify an exact match or a relaxed match. See [Specifying the expected response code](#) for more information.

Response Size Verification Failure [HTTP]

The number of bytes returned does not match the expected number of bytes. You can control how closely the returned response size must match the recorded response size. See [Specifying the expected response size](#) for more information.

Content Verification Point Failure

The received data does not match the expected data. Content verification point controls are protocol-specific.

Connection Failure

The workbench or agent computers cannot connect to the server under test.

Authentication Failure

An attempt to log in to the server under test failed.

End of Dataset reached

The last row of the dataset is reached. See [Dataset overview on page 288](#) for more information.

Reference Extraction Failure

The response received during playback is different from the response received when the test was recorded. Data correlation failed because the regular expression that is associated with the reference did not match the expected value.

Substitution Failure

A reference for an expected data substitution is a null reference.

Server Timeout

The server under test does not respond before the timeout interval elapses.

Dropped Connection

When the system under test drops the connection.

Custom Verification Point Failure

A custom verification point did not return a Pass status after performing a verification written in Java™ code. See [Reporting custom verification point failures on page 428](#).

Custom Code Alert

Custom code reported an **RPTCondition.CustomCodeAlert** condition. The following code reports a custom code alert:

```
tes.getTestLogManager().reportErrorCondition(RPTCondition.CustomCodeAlert);
```

See the `ITestLogManager` Javadoc for more information.

The Javadoc for the test execution services interfaces and classes can be accessed from the product by clicking **Help > Help Contents > IBM Rational Performance Tester API Reference**.

Custom Code Exception

The custom code in a test has an exception. By default, Rational® Performance Tester exits the user whenever there is an exception in the custom code. For information on setting different actions, see [Error-handling behavior](#)

Related information

[Specifying error-handling behavior](#)

Resource monitoring data sources

Resource monitoring data can be captured or imported from a number of sources.

IBM® Tivoli® Monitoring

IBM Tivoli® Monitoring monitors and manages system and network applications on a variety of platforms and keeps track of the availability and performance of all parts of your enterprise network. IBM® Tivoli® Monitoring provides reports that you can use to track trends and troubleshoot problems.

Not all IBM® Tivoli® Monitoring agents are [checkmark_perf.jpg](#). Over 100 IBM® Tivoli® Monitoring agents are available from IBM® and non-IBM vendors. The following IBM® Tivoli® Monitoring agents are [checkmark_perf.jpg](#) for resource monitoring data collection:

- Operating system agents
 - Monitoring Agent for Linux™ OS
 - Monitoring Agent for UNIX™ OS
 - Monitoring Agent for Windows™ OS
 - Monitoring Agent for z/OS®
- Application agents
 - Monitoring Agent for Citrix
 - Monitoring Agent for IBM® DB2®
 - Monitoring Agent for IBM® Tivoli® Composite Application Manager for WebSphere®
 - Monitoring Agent for IBM® WebSphere® Application Server

- Monitoring Agent for IBM® WebSphere® MQ
- Monitoring Agent for Oracle Database
- Monitoring Agent for SNMP-MIB2 (only)

IBM® DB2® Monitoring

IBM DB2® collects information from the database manager, its databases, and any connected applications. The snapshot monitor captures the state of database activity at a particular point in time.

IBM® WebSphere® Performance Monitoring Infrastructure

IBM WebSphere® Application Server collects performance data and provides interfaces so that external applications can monitor that performance data. To help identify performance problems and help tune an environment that runs web applications, data is collected through the Performance Monitoring Infrastructure (PMI). The Performance Monitoring Infrastructure is the underlying framework in WebSphere® Application Server that gathers performance data from various runtime resources, such as Java™ Virtual Machine (JVM) and Thread Pools, and application components, such as servlets and Enterprise JavaBeans™ (EJB) components.

Java™ Management Extensions

Java Management Extensions (JMX) can monitor performance characteristics of application servers and applications that are run on application servers. The following application servers support JMX monitoring:

- Apache HTTP Server
- Apache Tomcat
- JBoss Application Server
- Oracle WebLogic Server
- SAP NetWeaver

Java™ Virtual Machines also support JMX monitoring.

Oracle Database Metrics

Oracle Database collects metrics that are related to database health and workload.

UNIX™ rstatd

With the rstatd daemon, users can collect performance statistics remotely from networked UNIX™ (or Linux™) computers. The rstatd daemon collects statistics that are related to network, virtual memory, interrupt, disk, and processor usage.

Simple Network Management Protocol (SNMP) agents

The Simple Network Management Protocol (SNMP) is typically used to monitor network health, performance, and hardware. SNMP agents are software components that are installed on managed devices and collect management information.

Windows™ Performance Monitor

Windows Performance Monitor (PerfMon) collects data from performance objects. The Microsoft™ Windows™ operating system provides performance objects for the major hardware components: memory, processors, and so on.

Each performance object provides specific performance counters. For example, the `Memory` object provides a `Pages/sec` counter that tracks the rate of memory paging. Other programs on the computer, including Internet Information Services (IIS) and Microsoft™ SQL Server, can install their own performance objects. For example, a mail server program might install a mail performance object. The specific counters depend on the version of the Windows™ operating system and on the additional programs that are installed on the computer.

Response time breakdown data sources

Response time breakdown data can be imported from a number of sources.

IBM® Tivoli® Composite Application Manager for Application Diagnostics

IBM Tivoli® Composite Application Manager for Application Diagnostics enables users to view the health of web applications and servers, then drill down to diagnostic information for specific application requests to identify the root cause of problems.

IBM® Tivoli® Composite Application Manager for Response Time Tracking

IBM Tivoli® Composite Application Manager for Response Time Tracking measures the level of service that the application delivers to users. It does this by monitoring the availability and response time that users experience at the client desktop. It works with a wide range of web-based, e-business, and Microsoft™ Windows™ applications that run in many different environments.

IBM® Tivoli® Composite Application Manager for WebSphere®

IBM Tivoli® Composite Application Manager for WebSphere® provides immediate problem determination, availability monitoring, and performance analysis for enterprise WebSphere® applications running on Windows™, UNIX™, OS/400®, and z/OS® environments. IBM® Tivoli® Composite Application Manager for WebSphere® monitors heterogeneous environments consisting of both mainframes and distributed systems.

IBM® Tivoli® Monitoring for Transaction Performance

IBM Tivoli® Monitoring for Transaction Performance is a centrally managed suite of software components that monitor the availability and performance of web-based services and Windows™ applications.

Related information

[IBM Tivoli Composite Application Manager for Applications](#)

UI preferences

Read the UI preferences topics.

VU Schedule editor reference

Most VU Schedule editor settings apply either to the entire schedule or to individual user groups.

Schedule properties

When you open a schedule, you can set its properties.

User Load page

Right-click in the table, and select **Add** to add a stage. To modify a stage, select the row, and then click **Edit** or click the user icon in the first column.

Users

Enter the total number of users to be active in the stage (not the number of users to add or subtract to those currently running).

Run for specified period of time

Enter the length of time (and the time units) for the stage to run. When the specified number of users is achieved, the users will run for up to this time. When the time expires, the users continue to run if they are required for the next stage; otherwise, they are stopped gracefully.

Click **Show Advanced** to set further options to prepare the system under test before the users actually enter the stage:

Change Rate

Enter a number to set a delay between adding or removing each user, rather than adding them or subtracting them all at once. Staggering users avoids overloading the system, which can cause connection timeouts. The **User Load Preview** shows this delay in black.

Settle Time

A system under test might react to a sudden change in user population. With a defined settle time, which starts when the target number of users is reached, the system under test can settle into a steady state so that it can accurately reflect the user population. The **User Load Preview** shows this time in black.

Time limit for a user to respond to a stop request

Optionally enter a value. When a virtual user is asked to stop, it completes its current action (such as an HTTP request) and then finishes. If a virtual user has not finished within the specified time limit, the user is forced to finish.

User Load Preview

Previews the user population stages over time. The red line segments indicate that the total number of users has been achieved for the state.

Think Time page

Use the recorded think time

Select to play back a test at the same rate that it was recorded. This option has no effect on the think time.

Specify a fixed think time

Each user's think time is exactly the same value: the value that you specify. Although this does not emulate users accurately, it is useful if you want to play a test back quickly.

Increase/decrease the think time by a percentage

Type a percentage in **Think time scale**. Each user's think time is multiplied by that percentage. A value of 100 causes no change in think times. A value of 200 doubles the think times, so the schedule plays back half as fast as it was recorded. A value of 50 reduces the think times by half, so the schedule plays back twice as fast. A value of 0 indicates no delays.

Vary the think time by a random percentage

Each user's think time is randomly generated within the upper and lower bounds of the percentages that you supply. The percentage is based on the recorded think time. For example, if you enter 10 in **Lower limit** and enter 90 in **Upper limit**, the think times will be between 10 percent and 90 percent of the original recorded think time. The random time is uniformly distributed within this range.

Maximum think time

Setting a maximum think time is useful with tests that emulate actual think times. By setting a maximum, you do not have to search for and edit each long think time within a test. Numerous factors can generate long think times, for example, you might be interrupted while recording. To restore the original think times, clear this check box.

Resource Monitoring page**Enable resource monitoring**

Select to activate resource monitoring. The available data sources are captured from these sources:

- Apache HTTP Server Managed Beans
- Apache Tomcat Managed Beans
- IBM® Tivoli® monitoring agents
- IBM® DB2® snapshot monitors
- The IBM® WebSphere® Performance Monitoring Infrastructure
- JBoss Application Server Managed Beans
- Java™ Virtual Machine Managed Beans
- Oracle Database
- Oracle WebLogic Server Managed Beans
- SAP NetWeaver Managed Beans
- the UNIX™ rstatd monitor
- Simple Network Management Protocol (SNMP) agents
- Windows™ Performance Monitor

Resource monitoring data can provide a more complete view of a system to aid in problem determination.

Ignore invalid resources when executing the schedule

Select this setting to suppress any error messages that invalid resources cause, such as unreachable hosts or invalid host names. If you select this option, you must view logs to see error messages.

Statistics page

Statistics log level

These options are listed in order of the increasing amount of data that they collect for the test log.

None

Collects minimal statistical data. Use this option to run a schedule quickly for testing purposes.

Schedule Actions

Reports the number of active and completed users in the run.

Primary Test Actions

For HTTP tests, this option reports page-related actions (attempts, hits, and verification points). For SAP tests, this option reports information on SAP screens.

Secondary Test Actions

For HTTP tests, this option reports information that is related to page elements. This option does not apply to SAP tests.

All

Provides statistics for all actions.

Statistics sample interval

Sets the sampling interval for reports. When you run a schedule, the reports show such information as response time during a specific interval, the frequency of requests being transferred during an interval, and the average response trend during an interval. You set this sampling interval here.



Note: In Rational® Performance Tester 9.1.1 and later, during a test or schedule run, the **Elapsed Time** value is updated every 5 seconds irrespective of a value set in the **Statistics sample interval** field. You can view the **Elapsed Time** value that is changing in the **Run Summary** section on the **Performance Summary** tab of the report.

Only store All Hosts statistics

Select this option unless you are running a performance test over different WANs, and you are interested in seeing the data from each remote computer.

Discard time measurements for unhealthy transactions

Use the option to discard the elapsed time that is recorded for unhealthy transactions. When you select Discard time measurements for unhealthy transactions option, the time measurements for unhealthy transactions are not recorded.

When you define error conditions for a transaction, you can set whether the behavior of the errors affects the health of the test run. If one condition is met in a transaction, the transaction is marked unhealthy.

Discard response time for unhealthy pages (HTTP tests only)

Use the option to discard the page response times for the pages that fail. When you select Discard response time for unhealthy pages (HTTP tests only) option, the page response time is not recorded for unhealthy pages.

If the pages of your application display errors and the pages fail, then the page is not reported as healthy.

Variable Initialization

Use this page to initialize variables at the schedule level. When you initialize variables at the schedule level, all the user groups in the schedule use the variable initial values, except those for which a specific value is defined.

Add

Add a variable and initialize a value. The **Used by** column displays the test name that uses the corresponding variable. A warning icon is displayed for a variable that overrides the value specified at the schedule level or user group level and uses the value defined at the test level with the visibility set to **This test only**. Hover the cursor over the warning icon to view the tests that override the variable initial values.

Export

Export the variables defined at the schedule level to a file.

Use variable initial values file

Select this check box to use the variable values from a file. Click **Browse** to select an existing file or click **New** to create a file.

Performance Requirements page**Enable Requirements**

Select to enable the use of performance and functional requirements for this schedule.

Name

Specifies the name of this set of requirements. This name is used in the Requirements report. By default, the name is `Performance Schedule -schedule_name`.

Use Defaults

Click to reset **Name** to the default value.

Requirement

All the requirements are displayed in the table. Shaded requirements are not defined for this schedule.

To define a requirement, set an **Operator** and **Value**.

Expand the **Custom** section, and then double-click the row to add the counter information generated by using the custom code to the requirement.

Operator

Click this field to display a list of mathematical operators. Select an operator for this requirement.

Value

Click this field to set a value for the requirement.

Standard

Select to mark the requirement as standard. If a standard requirement is not met, the schedule run will have a verdict of fail, and this verdict will roll up to the entire run, like a verification point failure. Clear to make the requirement *supplemental*. In general, supplemental requirements are those that are tracked internally. A supplemental requirement cannot cause a run to fail, and its results are restricted to one page of the Requirements report.

Hide Undefined Requirements

Select to see only the requirements that you have defined. This hides the shaded rows.

Clear

Select one or more requirements and click to remove the definition. The requirement is still available and can be redefined.

Test Log page

The default setting, to log all errors and warnings and primary test actions, fits most purposes. However, you can log any type of information, from no information to all information from all users, although neither is typical.

- To see only errors and warnings, set the first two **What to Log** check boxes to **All**; then clear the third check box, **And also show all other types**, to avoid logging successful events.
- To check the structure of a schedule, when you are not interested in the test execution results, set all three **What to Log** check boxes to **Schedule Actions**.

Both choices and the default setting limit the size of the test log and reduce the total time to run the schedule by significantly reducing the test log transfer time at the end of a test.

If you are debugging a test, you might set all three **What to Log** fields to **All** or **Action Details**. These settings produce large test logs, especially if your tests are long or you are running a large number of users. Large test logs, in turn, increase the test log transfer time, and might even cause your computer to run out of disk space.

To reduce transfer times and the likelihood of running out of disk space, sample information from a very small subset of users; smaller even than the default of 5 users per user group. A fixed sampling rate samples the same number of virtual users from each group. A percentage sampling rate samples a percentage of virtual users from each group, but guarantees that at least one user is sampled from a group.

Response Time Breakdown page

Enable collection of response time data

Select to activate response time breakdown collection. This data shows you the response time breakdown for each page element.

Detail level

Select **Low** or **Medium** to limit the amount of collected data.

Only sample information from a subset of users

If you set the detail level to **High** or **Medium**, set a sampling rate to prevent the log from getting too large.

Fixed number of users

The number that you select is sampled from each user group. Unless you have specific reasons to collect data from multiple users, select **Fixed number of users**, and specify one user per user group.

Percentage of users

The percentage that you select is sampled from each user group, but at least one user is sampled from each user group.

Problem Determination page

Problem determination log level

In general, change the problem determination level only when asked to by IBM® Software Support. However, under certain conditions, you might want to change the problem determination level. For example, if problems occur when a run reaches a certain number of users, you might increase the level to **Config**, which is the most detailed level to use without consulting IBM® Software Support.

Only sample information from a subset of users

Select this option to set a sampling rate.

Fixed number of users

Specify the number of users to sample from each user group.

Percentage of users

The percentage that you select is sampled from each user group, but at least one user is sampled from each group.

User group properties

When you open a user group, you can set these properties.

Group size

Specifies either an absolute number of users, or a percentage of users, which you control dynamically.

Locations

Run this group on the local computer

Indicates that the user group should be run on your computer.

Run this group at the following locations

Indicates that the user group should be run on one or more remote computers, at the indicated locations. Typically, you run a user group at a remote location if you are running a large number of virtual users.

Options

Use the Options page to override the think time behavior of your schedule for a specific user group and to specify protocol specific options.

Override think time option

Select this check box to specify a think time behavior for the current user group.

Use the recorded think time

Select to play back a test at the same rate that it was recorded. This option has no effect on the think time.

Specify a fixed think time

Each user's think time is exactly the same value: the value that you specify. Although this does not emulate users accurately, it is useful if you want to play a test back quickly.

Increase/decrease the think time by a percentage

Type a percentage in the **Think time scale**. Each user's think time is multiplied by that percentage. A value of 100 causes no change in think times. A value of 200 doubles the think times, so the schedule plays back half as fast as it was recorded. A value of 50 reduces the think times by half, so the schedule plays back twice as fast. A value of 0 indicates no delays.

Vary the think time by a random percentage

Each user's think time is randomly generated within the upper and lower bounds of the percentages that you supply. The percentage is based on the recorded think time. For example, if you select a **Lower limit**

of 10 and an **Upper limit** of 90, the think times will be between 10 percent and 90 percent of the original recorded think time. The random time is uniformly distributed within this range.

Limit think times to a maximum value

Setting a maximum think time is useful with tests that emulate actual think times. By setting a maximum, you do not have to search for and edit each long think time within a test, if, for example, you are interrupted during recording. No think time used will be greater than the maximum limit you set, even if you have chosen to vary the think time by a percentage that would exceed this maximum. To restore the original think times, clear this box.

Protocol-specific options

Click **Edit options** to set protocol-specific options for all tests in the user group. These settings override the protocol-specific options of the schedule.

Variable Initialization

Use this page to initialize variables at the user group level. When you initialize variables at the user group level, all the tests in the user group use the variables. If the same variable is defined at the schedule level, precedence is given to the variable at the user group level.

Add

Add a variable and initialize a value. The **Used by** column displays the test name that uses the corresponding variable. A warning icon is displayed for a variable that override the value specified at the schedule level or user group level and uses the value defined at the test level with the visibility set to **This test only**. Hover the cursor over the warning icon to view the tests that overrides the variable initial values.

Export

Export the variables defined at the user group level to a file.

Use variable initial values file

Select this check box to use the variable values from a file. Click **Browse** to select an existing file or click **New** to create a file.

Generic service client references

Read these reference topics for Generic service client.

Generic service client call details

In the generic service client, service calls contain the content and the transport information for the call. The contents are made of the SOAP envelope. The transport information refers to the information that is required to send and receive and answer depending on the selected protocol.

Message

This page shows the XML content of the request and provides access to data correlation. The same content is presented in three different ways.

Form

This view provides a simplified view of the message that focuses on editing the values of the XML content. Use the **Schema** menu to enable assistance with editing XML content so that the XML is valid and complies with the XSD specification.

In the **Form** view, add the XML headers that are required for standard web service calls. On the **Header** bar, click **Add** (+) to create the default XML header structure for WS-Addressing, WS-ReliableMessaging or WS-Coordination requests, or click **More** for other standards. You can enable or disable XML header elements and specify the correct values for each XML element. Checks are performed to ensure that the XML content is valid.



Note: To add XML headers to calls in IBM® Security AppScan®, add a **Static XML Headers** algorithm on the **Request Stack** tab of the request.

Tree

This view provides a hierarchical view of the XML structure of the message, including elements, namespaces, and the associated values. You can use **Add**, **Insert**, **Remove**, **Up**, and **Down** to edit the XML elements and namespaces in the tree.

Use **Skip if Empty** column to select the empty XML elements that you want to skip. This column is visible only if you selected the **Display the 'Skip if Empty' column in XML tree viewer** check box in **Window > Preferences > Test > Test editor > Service test**.


Click **Filter** to hide or show namespace, attribute, or text nodes, depending on your requirements.

Click **Allow only valid modifications** to enable smart editing, based on a specified XML schema document (XSD). To specify a set of XSD documents for the workbench, in the test navigator, right-click the project and select **Properties** and **Schema Catalog**. Disable **Allow only valid modifications** if you do not have an XSD or if you want to bypass the schema.

You can right-click an XML element to convert it to an XML fragment. This enables you to perform data correlation (use datasets and create references) on the entire XML fragment instead of only on the value.

Source

This view displays the source XML content of the message or plain text content. To format XML content, click **Format XML text**. To wrap XML content into a single line, click **Pack XML text to single line**. Similar controls are available for JSON content.

 **Important:** In the Source view, do not edit the tags that start with `SoaTag`. If you delete or change these tags, any references and substitutions in the test will be broken. You cannot recreate these tags after you delete them.

Attachments

This page lists the MIME or DIME attachments that are attached to the request. The contents of this view conform to the Multipurpose Internet Mail Extensions (MIME) or Direct Internet Message Encapsulation (DIME) specification. You can use this page to add workbench resources as MIME or DIME attachments and change properties.

The **Content ID** is the identifier that the request uses to refer to the attachments. The method for using this identifier depends on your server requirements.

MIME or DIME

Select whether the attachment conforms to the Multipurpose Internet Mail Extensions (MIME) or Direct Internet Message Encapsulation (DIME) specification

Use MTOM transmission mechanism

By default, the request uses SOAP Messages with Attachments (SwA) to handle attachments. Select this option to handle attachments with the SOAP Message Transmission Optimization Mechanism (MTOM).

Transport

This page covers the transport settings used to send the request. The transport protocol settings apply to a transport configuration, which can be either HTTP, Java™ Message Service (JMS), WebSphere® MQ, or Microsoft .NET. You can create several configurations for each protocol so that you can easily switch protocols or variants of protocols.



Note: If you are using IBM® Security AppScan®, only the HTTP transport protocol is available.

HTTP

Select **HTTP** to use the HTTP transport for the request. At the request level, you can update a URL or SOAP action and the reference to the global configuration of a test.

Protocol configuration

Click **Change** to specify a predefined transport configuration or to create a configuration. HTTP transport configurations contain proxy and authentication settings that can be reused.

URL

Specify the URL end point of the service request.

Rest mode

Use this check box to split the REST URL so that it is easy to understand the different parts of REST URL. When you use this option, the main section of URL is placed in the URL field, the resource part is placed in the **Resource** field, and the parameters are placed in the **Parameters** field. Use the **Add** button to manually add more parameters.

Method

Specify the HTTP method to be used to invoke the service request.

Headers

Specify the names and values of any custom HTTP headers that are required by the service. Click **Add**, **Edit** or **Remove** to modify the headers list.

Cookies

Specify the names and values of any cookies that are required by the service. Click **Add**, **Edit** or **Remove** to modify the cookies list.

JMS

Select **JMS** to use the Java™ Messaging Service transport for the request. This page enables you to add string properties that are attached to the request for a JMS configuration. These will be sent as message properties through JMS.

Protocol configuration

Click **Change** to specify a predefined transport configuration or to create a configuration. JMS transport configurations contain generic end point, reception point, and adapter settings that can be reused.

Properties

Specify the names and values of any string properties that are required by the request for the current JMS transport configuration. These are sent as message properties through JMS. Click **Add**, **Edit** or **Remove** to modify the properties list.

WebSphere® MQ

Select **MQ** to use the IBM® WebSphere® MQ transport for the request. This page enables you to specify the SOAP action and override the settings for the WebSphere® MQ configuration selected at the test level.

Protocol configuration

Click **Change** to specify a predefined transport configuration or to create a configuration. WebSphere® MQ transport configurations contain generic queue, header, and SSL settings that can be reused.

SOAP Action

Specifies the SOAP action to be used to invoke the WebSphere® MQ request.

Override MQ protocol configuration values

Select this option to configure the fields of the WebSphere® MQ message. You can replace a subset of an MQ message descriptor with a custom format for use with other server types, specifically when using an XML message request.

Customize message header

Select this option to specify custom headers for the transport for the SOAP over MQ feature that is provided by WebSphere® MQ. This feature uses a predetermined MQ message format (RFH2), therefore, when selected, other

Message Descriptor options are disabled.

Message descriptor

These settings replace the message descriptor and header settings of the MQ protocol configuration. Refer to WebSphere® MQ documentation for information about message descriptors.

Microsoft™ .NET

Select **Microsoft .NET** to use the Microsoft .NET Framework transport for requests based on Windows™ Communication Foundation (WCF). This page enables you to override the settings for the Microsoft™ .NET configuration selected at the test level.

Item

Click **Add** to specify the name and value of the WCF actions that are required by the service. This table is automatically generated when you import a Microsoft .NET WSDL file. Refer to the Microsoft™ .NET WCF documentation for more information.

Security for Request

In this page you can edit the security algorithm stacks that the security protocol uses. Security stacks are a set of algorithms that are executed in a given order.

Override stack

By default, you edit the security algorithm stack attached to a specific WSDL file in the WSDL Security Editor. Select this option to specify a different security algorithm stack only for the current response.

Show response stack

Clear the **Show response stack** option to hide the **Response Stack** page.

Security Algorithm Details

Click **Add**, **Insert**, or **Remove** to add or remove security algorithms in the stack. Click **Up** and **Down** to change the order of a selected algorithm in the security stack. The following security algorithms can be added to the security stack:

XML Encryption

The XML encryption security algorithm specifies how the XML document is encrypted. For details on security algorithms, refer to the web service security specification.

Actor / Role name

Specify the name of the recipient of the algorithm header element, if required.

Must understand

Select whether it is mandatory that the algorithm header is processed by the recipient, if required. The recipient is either the Actor name or the server.

Identifier type

Select the type of key identifier to be used for the encryption. The following key identifiers are available, as defined in the Web Services Security (WSS) specification X509 profile and the OASIS WSS 1.1 specification:

- ISSUER_SERIAL
- BST_DIRECT_REFERENCE
- X509_KEY_IDENTIFIER
- SKI_KEY_IDENTIFIER
- EMBEDDED_KEYNAME
- THUMBPRINT_IDENTIFIER
- ENCRYPTED_KEY_SHA1_IDENTIFIER

User XPath part selection

This enables you to specify an XPath query that describes parts of the XML document that can be subjects of the algorithm. By default, the body is the subject.

Key

Select the key used for the encryption. The details of each key vary.

- **x509 key**: This specifies the name and password of the x509 key and the keystore where it is located.
- **Raw key**: This specifies the name and the byte value of your SecretKey in hexadecimal.
- **Encrypted key**: This specifies a reference to an encrypted key that was previously defined in the security stack. Click **Insert a new encrypted key** to create a new encrypted key definition block.

Encoding Algorithm Name

Specify the encryption method to be used as defined in the XML Encryption Syntax and Processing specification.

Key Encoding Algorithm

Specify the standard algorithm for encoding the key as defined in the XML Encryption Syntax and Processing specification.

Encrypted Key

This block defines an encrypted key that can be used in an XML signature or XML encryption block. The encrypted key block must be before a block that uses the encrypted key.

Actor / Role name

Specify the name of the recipient of the algorithm header element, if required.

Must understand

Select whether it is mandatory that the algorithm header is processed by the recipient, if required. The recipient is either the Actor name or the server.

Key name

Specify the name of the encrypted key.

Identifier type

Select the type of key identifier to be used for the key. The following key identifiers are available, as defined in the the Web Service Security (WSS) specification X509 profile and OASIS WSS 1.1 specification:

- ISSUER_SERIAL
- BST_DIRECT_REFERENCE
- X509_KEY_IDENTIFIER
- THUMBPRINT_IDENTIFIER
- SKI_KEY_IDENTIFIER

Key size

Specify the size of the key in bits.

Key encoding algorithm name

Specify the algorithm to be used for encoding the key.

Keystore

Select a keystore or click **Edit Security** to define a new keystore or to manage the existing keystores.

Name

Select a key contained in the specified keystore.

Password

Type the password for the selected key name.

Custom Security Algorithm

If you want to use a Java™ class as a custom security algorithm, then use this stack element to apply the custom algorithm to the service.

Java™ Project

If you have not implemented a custom Java™ class, select **Java Project**, type a name for the new project, and click **Generate** to create a new Java™ class with the default structure for custom security implementations.



Note: If you are using IBM® Security AppScan®, this field is not available.

Implementation class

Specify the name of the class that implements the custom security algorithm. Click **Browse Class** to select an existing Java™ class from the workspace.

Properties

Use this table to send any specific properties and associated values to the custom security algorithm.

Request Stack

Use this page to specify the stack that applies security and addressing parameters and algorithms to service requests before they are sent. Stacks are a set of algorithms that are executed in a given order. Use the WSDL security editor to define a stack for each WSDL. The stack will be applied to all requests that use the WSDL.

Override stack

By default, you edit a stack which attached to a specific WSDL file in the WSDL security editor. Select this option to specify a different security algorithm stack only for the current service request.

Show response stack

The **Request Stack** page contains algorithms that are applied only to outgoing service requests. Select **Show response stack** to add a **Response Stack** page. The **Response Stack** page allows you to edit security and addressing parameters and algorithms that are applied to incoming responses.

Security Algorithm Details

Click **Add**, **Insert**, or **Remove** to add or remove security algorithms in the stack. Click **Up** and **Down** to change the order of a selected algorithm in the security stack. The following security algorithms can be added to the security stack:

Static XML Headers

Use this algorithm to add the XML headers that are required for web service standard calls. On the **Header** bar, click **Add** (+) to create the default XML header structure for WS-Addressing, WS-ReliableMessaging or WS-Coordination requests, or click **More** for other standards.

You can enable or disable XML elements in the Header section and specify the correct values for each XML element. Checks are performed to ensure that the XML headers are valid.



Note: The Static XML Headers algorithm is available only in IBM® Security AppScan®. To add static XML headers to calls in other products, expand the **Headers** section on the **Message** tab of the request.

Time Stamp

The time stamp security algorithm adds time stamp information to the XML document in the response. For details on security algorithms, refer to the web service security specification.

Actor / Role name

Specify the name of the recipient of the algorithm header element, if required.

Must understand

Select whether it is mandatory that the algorithm header is processed by the recipient, if required. The recipient is either the Actor name or the server.

Expiration delay

Specify the delay after which the time stamp expires.

Millisecond precision

Select this option to produce a time stamp that uses millisecond precision instead of the default (1/100th second).

User name token

The user name token security algorithm adds a user name token to the XML document in the message. For details on security algorithms, refer to the web service security specification.

Actor / Role name

Specify the name of the recipient of the algorithm header element, if required.

Must understand

Select whether it is mandatory that the algorithm header is processed by the recipient, if required. The recipient is either the Actor name or the server.

Name

Type the name of the user.

Password

Type the password of the user.

Password type

Specify the password type for the security algorithm as defined in the Web Services Security UsernameToken profile.

Use nonce

Select this check box to add the Nonce element to the User Name Token XML code. In most cases, the Nonce ID is required.

Use created

Select this check box to add current timestamp to the Created XML element in the User Name Token XML.

XML Encryption

The XML encryption security algorithm specifies how the XML document is encrypted. For details on security algorithms, refer to the web service security specification.

Actor / Role name

Specify the name of the recipient of the algorithm header element, if required.

Must understand

Select whether it is mandatory that the algorithm header is processed by the recipient, if required. The recipient is either the Actor name or the server.

Identifier type

Select the type of key identifier to be used for the encryption. The following key identifiers are available, as defined in the Web Services Security (WSS) specification X509 profile and the OASIS WSS 1.1 specification:

- ISSUER_SERIAL
- BST_DIRECT_REFERENCE
- X509_KEY_IDENTIFIER
- SKI_KEY_IDENTIFIER
- EMBEDDED_KEYNAME

- THUMBPRINT_IDENTIFIER
- ENCRYPTED_KEY_SHA1_IDENTIFIER

User XPath part selection

This enables you to specify an XPath query that describes parts of the XML document that can be subjects of the algorithm. By default, the body is the subject.

Key

Select the key used for the encryption. The details of each key vary.

- **x509 key**: This specifies the name and password of the x509 key and the keystore where it is located.
- **Raw key**: This specifies the name and the byte value of your SecretKey in hexadecimal.
- **Encrypted key**: This specifies a reference to an encrypted key that was previously defined in the security stack. Click **Insert a new encrypted key** to create a new encrypted key definition block.

Encoding Algorithm Name

Specify the encryption method to be used as defined in the XML Encryption Syntax and Processing specification.

Key Encoding Algorithm

Specify the standard algorithm for encoding the key as defined in the XML Encryption Syntax and Processing specification.

XML Signature

The XML signature security algorithm specifies how the XML document is signed. For details on security algorithms, refer to the web service security specification.

Actor / Role name

Specify the name of the recipient of the algorithm header element, if required.

Must understand

Select whether it is mandatory that the algorithm header is processed by the recipient, if required. The recipient is either the Actor name or the server.

Security token

Select the type of key identifier to be used for the signature. The following key identifiers are available, as defined in the the Web Service Security (WSS) specification X509 profile and OASIS WSS 1.1 specification:

- ISSUER_SERIAL
- BST_DIRECT_REFERENCE
- X509_KEY_IDENTIFIER
- SKI_KEY_IDENTIFIER
- KEY_VALUE
- USER_NAME_TOKEN
- CUSTOM_SYMM_SIGNATURE

In addition, the following identifiers are available when the signature is based on a UsernameToken profile:

- USER_NAME_TOKEN
- CUSTOM_SYMM_SIGNATURE

User XPath part selection

Specify an XPath query that describes parts of the XML document that can be the subjects of the algorithm. By default, the body is the subject. Click the **XPath Helper** button to build the Xpath expression.

Key

Select the key used for the encryption. The details of each key vary.

- **x509 key:** This key specifies the name and password of the x509 key and the keystore where it is located.
- **User name token key:** This specifies a user name and password for the signature.
- **Encrypted key:** This specifies a reference to an encrypted key that was previously defined in the security stack. Click **Insert a new encrypted key** to create a new encrypted key definition block.

Signature algorithm name

Specify the signature method algorithm as described in the XML Signature Syntax and Processing specification.

Canonicalization

Specify the canonicalization method to be used as described in the XML Signature Syntax and Processing specification.

Digest algorithm method

Specify which digest method to be used based on the algorithm method used on the server side.

Inclusive namespaces

Specify whether the canonicalization is exclusive as described in the Exclusive XML Canonicalization specification.

Encrypted Key

This block defines an encrypted key that can be used in an XML signature or XML encryption block. The encrypted key block must be before a block that uses the encrypted key.

Actor / Role name

Specify the name of the recipient of the algorithm header element, if required.

Must understand

Select whether it is mandatory that the algorithm header is processed by the recipient, if required. The recipient is either the Actor name or the server.

Key name

Specify the name of the encrypted key.

Identifier type

Select the type of key identifier to be used for the key. The following key identifiers are available, as defined in the the Web Service Security (WSS) specification X509 profile and OASIS WSS 1.1 specification:

- ISSUER_SERIAL
- BST_DIRECT_REFERENCE
- X509_KEY_IDENTIFIER
- THUMBPRINT_IDENTIFIER
- SKI_KEY_IDENTIFIER

Key size

Specify the size of the key in bits.

Key encoding algorithm name

Specify the algorithm to be used for encoding the key.

Keystore

Select a keystore or click **Edit Security** to define a new keystore or to manage the existing keystores.

Name

Select a key contained in the specified keystore.

Password

Type the password for the selected key name.

Custom Security Algorithm

If you want to use a Java™ class as a custom security algorithm, then use this stack element to apply the custom algorithm to the service.

Java™ Project

If you have not implemented a custom Java™ class, select **Java Project**, type a name for the new project, and click **Generate** to create a new Java™ class with the default structure for custom security implementations.



Note: If you are using IBM® Security AppScan®, this field is not available.

Implementation class

Specify the name of the class that implements the custom security algorithm. Click **Browse Class** to select an existing Java™ class from the workspace.

Properties

Use this table to send any specific properties and associated values to the custom security algorithm.

WS-Addressing Algorithm

Use this block if your service uses either WS-Addressing 2004/08 or the WS-Addressing 1.0 Core standard.

Namespace

Specify the namespace for either WS-Addressing 2004/08 or WS-Addressing 1.0 Core.

Action if request uses WS-Addressing

Select the action to complete if WS-Addressing is already in the request.

Replace anonymous address in Reply-to with:

Select this option to generate the specified address in the Reply-to header instead of an anonymous address.

Remove WS-Addressing from response

Select this option to strip any WS-Addressing headers from the response.

WS-Policy Algorithm

Use this block if your service requires a security policy file compliant with the WS-Policy specification.

Use policy included in WSDL (WS-PolicyAttachment)

Select this option to use the security policy configuration that is attached to the WSDL as in the WS-PolicyAttachment specification.

Policy

If you are not using the WS-PolicyAttachment specification, specify the XML policy file. Click **Browse** to add a policy file from the workspace or to import a policy file.

Signature configuration

Select this option to specify a keystore for any signature that is specified in the policy. Click **Edit Security** to add a keystore from the workspace or to import a keystore.

Encryption configuration

Select this option to specify a keystore for any encryption that is specified in the policy. Click **Edit Security** to add a keystore from the workspace or to import a keystore.

Decryption configuration

Select this option to specify a keystore for any decryption that is specified in the policy. Click **Edit Security** to add a keystore from the workspace or to import a keystore.

Retrieve token from security token server (WS-Trust and WS-SecureConversation)

Select this option, and click **Configure** to specify a Security Token Server (STS) to use with the policy.

Additional properties

Use this table to specify settings for the advanced properties or specific implementations of the WS-Security specification. Click **Add** to add a property name and to set a value.

Response Stack

Use this page to specify the stack that applies security and addressing parameters to responses after they are received. Stacks are a set of algorithms that are executed in a given order. Use the WSDL security editor to define a stack for each WSDL. The stack will be applied to all requests that use the WSDL.

Override stack

By default, you edit the security algorithm stack attached to a specific WSDL file in the WSDL Security Editor. Select this option to specify a different security algorithm stack only for the current response.

Show response stack

Clear the **Show response stack** option to hide the **Response Stack** page.

Security Algorithm Details

Click **Add**, **Insert**, or **Remove** to add or remove security algorithms in the stack. Click **Up** and **Down** to change the order of a selected algorithm in the security stack. The following security algorithms can be added to the security stack:

XML Encryption

The XML encryption security algorithm specifies how the XML document is encrypted. For details on security algorithms, refer to the web service security specification.

Actor / Role name

Specify the name of the recipient of the algorithm header element, if required.

Must understand

Select whether it is mandatory that the algorithm header is processed by the recipient, if required. The recipient is either the Actor name or the server.

Identifier type

Select the type of key identifier to be used for the encryption. The following key identifiers are available, as defined in the Web Services Security (WSS) specification X509 profile and the OASIS WSS 1.1 specification:

- ISSUER_SERIAL
- BST_DIRECT_REFERENCE
- X509_KEY_IDENTIFIER
- SKI_KEY_IDENTIFIER
- EMBEDDED_KEYNAME
- THUMBPRINT_IDENTIFIER
- ENCRYPTED_KEY_SHA1_IDENTIFIER

User XPath part selection

This enables you to specify an XPath query that describes parts of the XML document that can be subjects of the algorithm. By default, the body is the subject.

Key

Select the key used for the encryption. The details of each key vary.

- **x509 key:** This specifies the name and password of the x509 key and the keystore where it is located.
- **Raw key:** This specifies the name and the byte value of your SecretKey in hexadecimal.
- **Encrypted key:** This specifies a reference to an encrypted key that was previously defined in the security stack. Click **Insert a new encrypted key** to create a new encrypted key definition block.

Encoding Algorithm Name

Specify the encryption method to be used as defined in the XML Encryption Syntax and Processing specification.

Key Encoding Algorithm

Specify the standard algorithm for encoding the key as defined in the XML Encryption Syntax and Processing specification.

Encrypted Key

This block defines an encrypted key that can be used in an XML signature or XML encryption block. The encrypted key block must be before a block that uses the encrypted key.

Actor / Role name

Specify the name of the recipient of the algorithm header element, if required.

Must understand

Select whether it is mandatory that the algorithm header is processed by the recipient, if required. The recipient is either the Actor name or the server.

Key name

Specify the name of the encrypted key.

Identifier type

Select the type of key identifier to be used for the key. The following key identifiers are available, as defined in the the Web Service Security (WSS) specification X509 profile and OASIS WSS 1.1 specification:

- ISSUER_SERIAL
- BST_DIRECT_REFERENCE
- X509_KEY_IDENTIFIER
- THUMBPRINT_IDENTIFIER
- SKI_KEY_IDENTIFIER

Key size

Specify the size of the key in bits.

Key encoding algorithm name

Specify the algorithm to be used for encoding the key.

Keystore

Select a keystore or click **Edit Security** to define a new keystore or to manage the existing keystores.

Name

Select a key contained in the specified keystore.

Password

Type the password for the selected key name.

Custom Security Algorithm

If you want to use a Java™ class as a custom security algorithm, then use this stack element to apply the custom algorithm to the service.

Java™ Project

If you have not implemented a custom Java™ class, select **Java Project**, type a name for the new project, and click **Generate** to create a new Java™ class with the default structure for custom security implementations.



Note: If you are using IBM® Security AppScan®, this field is not available.

Implementation class

Specify the name of the class that implements the custom security algorithm. Click **Browse Class** to select an existing Java™ class from the workspace.

Properties

Use this table to send any specific properties and associated values to the custom security algorithm.

Generic service client binary call details

In the generic service client, binary calls are specialized calls for sending binary messages. The transport information refers to the information that is required to send and receive and answer depending on the selected protocol.

Update node name automatically

Select this option to automatically rename the request in the Test Contents view.

Do not wait for response

Select this option to skip directly to the next request in the test after the current request is sent.

Time Out (ms)

This is the timeout value in milliseconds. If no response is received after the specified time, an error is produced.

Think Time (ms)

This specifies the programmatically calculated time delay that is observed for each user when this test is run with multiple virtual users. Think time is a statistical emulation of the amount of time actual users spend reading or thinking before performing an action.

Update Response

Click this button to invoke the request with the current settings and to use the response to create a binary response element or to update the existing response element.

Message**Source**

This page presents the binary contents of the request and provides access to data correlation. The same contents are presented in Binary and Raw ASCII views.

Attachments

This page lists the MIME or DIME attachments that are attached to the request. The contents of this view conform to the Multipurpose Internet Mail Extensions (MIME) or Direct Internet Message Encapsulation (DIME) specification. You can use this page to add workbench resources as MIME or DIME attachments and change properties.

Transport

This page covers the transport protocol used to send the request. The transport protocol can be either HTTP, Java™ Message Service (JMS), or WebSphere® MQ. You can create several configurations for each protocol so that you can easily switch protocols or variants of protocols.



Note: If you are using IBM® Security AppScan®, only the HTTP transport protocol is available.

Attachments

This page lists the MIME or DIME attachments that are attached to the request. The contents of this view conform to the Multipurpose Internet Mail Extensions (MIME) or Direct Internet Message Encapsulation (DIME) specification. You can use this page to add workbench resources as MIME or DIME attachments and change properties.

The **Content ID** is the identifier that the request uses to refer to the attachments. The method for using this identifier depends on your server requirements.

MIME or DIME

Select whether the attachment conforms to the Multipurpose Internet Mail Extensions (MIME) or Direct Internet Message Encapsulation (DIME) specification

Use MTOM transmission mechanism

By default, the request uses SOAP Messages with Attachments (SwA) to handle attachments. Select this option to handle attachments with the SOAP Message Transmission Optimization Mechanism (MTOM).

Transport

This page covers the transport settings used to send the request. The transport protocol settings apply to a transport configuration, which can be either HTTP, Java™ Message Service (JMS), WebSphere® MQ, or Microsoft .NET. You can create several configurations for each protocol so that you can easily switch protocols or variants of protocols.



Note: If you are using IBM® Security AppScan®, only the HTTP transport protocol is available.

HTTP

Select **HTTP** to use the HTTP transport for the request. At the request level, you can update a URL or SOAP action and the reference to the global configuration of a test.

Protocol configuration

Click **Change** to specify a predefined transport configuration or to create a configuration. HTTP transport configurations contain proxy and authentication settings that can be reused.

URL

Specify the URL end point of the service request.

Rest mode

Use this check box to split the REST URL so that it is easy to understand the different parts of REST URL. When you use this option, the main section of URL is placed in the URL field, the resource part is placed in the **Resource** field, and the parameters are placed in the **Parameters** field. Use the **Add** button to manually add more parameters.

Method

Specify the HTTP method to be used to invoke the service request.

Headers

Specify the names and values of any custom HTTP headers that are required by the service. Click **Add**, **Edit** or **Remove** to modify the headers list.

Cookies

Specify the names and values of any cookies that are required by the service. Click **Add**, **Edit** or **Remove** to modify the cookies list.

JMS

Select **JMS** to use the Java™ Messaging Service transport for the request. This page enables you to add string properties that are attached to the request for a JMS configuration. These will be sent as message properties through JMS.

Protocol configuration

Click **Change** to specify a predefined transport configuration or to create a configuration. JMS transport configurations contain generic end point, reception point, and adapter settings that can be reused.

Properties

Specify the names and values of any string properties that are required by the request for the current JMS transport configuration. These are sent as message properties through JMS. Click **Add**, **Edit** or **Remove** to modify the properties list.

WebSphere® MQ

Select **MQ** to use the IBM® WebSphere® MQ transport for the request. This page enables you to specify the SOAP action and override the settings for the WebSphere® MQ configuration selected at the test level.

Protocol configuration

Click **Change** to specify a predefined transport configuration or to create a configuration. WebSphere® MQ transport configurations contain generic queue, header, and SSL settings that can be reused.

SOAP Action

Specifies the SOAP action to be used to invoke the WebSphere® MQ request.

Override MQ protocol configuration values

Select this option to configure the fields of the WebSphere® MQ message. You can replace a subset of an MQ message descriptor with a custom format for use with other server types, specifically when using an XML message request.

Customize message header

Select this option to specify custom headers for the transport for the SOAP over MQ feature that is provided by WebSphere® MQ. This feature uses a predetermined MQ message format (RFH2), therefore, when selected, other **Message Descriptor** options are disabled.

Message descriptor

These settings replace the message descriptor and header settings of the MQ protocol configuration. Refer to WebSphere® MQ documentation for information about message descriptors.

Microsoft™ .NET

Select **Microsoft .NET** to use the Microsoft .NET Framework transport for requests based on Windows™ Communication Foundation (WCF). This page enables you to override the settings for the Microsoft™ .NET configuration selected at the test level.

Item

Click **Add** to specify the name and value of the WCF actions that are required by the service. This table is automatically generated when you import a Microsoft .NET WSDL file. Refer to the Microsoft™ .NET WCF documentation for more information.

Generic service client text call details

In the generic service client, text calls are specialized calls for sending text messages. The transport information refers to the information that is required to send and receive and answer depending on the selected protocol.

Update node name automatically

Select this option to automatically rename the request in the Test Contents view.

Do not wait for response

Select this option to skip directly to the next request in the test after the current request is sent.

Time Out (ms)

This is the timeout value in milliseconds. If no response is received after the specified time, an error is produced.

Think Time (ms)

This specifies the programmatically calculated time delay that is observed for each user when this test is run with multiple virtual users. Think time is a statistical emulation of the amount of time actual users spend reading or thinking before performing an action.

Update Response

Click this button to invoke the request with the current settings and to use the response to create a binary response element or to update the existing response element.

Message

Source

This page presents the binary contents of the request and provides access to data correlation. The same contents are presented in Binary and Raw ASCII views.

Attachments

This page lists the MIME or DIME attachments that are attached to the request. The contents of this view conform to the Multipurpose Internet Mail Extensions (MIME) or Direct Internet Message Encapsulation (DIME) specification. You can use this page to add workbench resources as MIME or DIME attachments and change properties.

Transport

This page covers the transport protocol used to send the request. The transport protocol can be either HTTP, Java™ Message Service (JMS), or WebSphere® MQ. You can create several configurations for each protocol so that you can easily switch protocols or variants of protocols.



Note: If you are using IBM® Security AppScan®, only the HTTP transport protocol is available.

Attachments

This page lists the MIME or DIME attachments that are attached to the request. The contents of this view conform to the Multipurpose Internet Mail Extensions (MIME) or Direct Internet Message Encapsulation (DIME) specification. You can use this page to add workbench resources as MIME or DIME attachments and change properties.

The **Content ID** is the identifier that the request uses to refer to the attachments. The method for using this identifier depends on your server requirements.

MIME or DIME

Select whether the attachment conforms to the Multipurpose Internet Mail Extensions (MIME) or Direct Internet Message Encapsulation (DIME) specification

Use MTOM transmission mechanism

By default, the request uses SOAP Messages with Attachments (SwA) to handle attachments. Select this option to handle attachments with the SOAP Message Transmission Optimization Mechanism (MTOM).

Transport

This page covers the transport settings used to send the request. The transport protocol settings apply to a transport configuration, which can be either HTTP, Java™ Message Service (JMS), WebSphere® MQ, or Microsoft .NET. You can create several configurations for each protocol so that you can easily switch protocols or variants of protocols.



Note: If you are using IBM® Security AppScan®, only the HTTP transport protocol is available.

HTTP

Select **HTTP** to use the HTTP transport for the request. At the request level, you can update a URL or SOAP action and the reference to the global configuration of a test.

Protocol configuration

Click **Change** to specify a predefined transport configuration or to create a configuration. HTTP transport configurations contain proxy and authentication settings that can be reused.

URL

Specify the URL end point of the service request.

Rest mode

Use this check box to split the REST URL so that it is easy to understand the different parts of REST URL. When you use this option, the main section of URL is placed in the URL field, the resource part is placed in the **Resource** field, and the parameters are placed in the **Parameters** field. Use the **Add** button to manually add more parameters.

Method

Specify the HTTP method to be used to invoke the service request.

Headers

Specify the names and values of any custom HTTP headers that are required by the service. Click **Add**, **Edit** or **Remove** to modify the headers list.

Cookies

Specify the names and values of any cookies that are required by the service. Click **Add**, **Edit** or **Remove** to modify the cookies list.

JMS

Select **JMS** to use the Java™ Messaging Service transport for the request. This page enables you to add string properties that are attached to the request for a JMS configuration. These will be sent as message properties through JMS.

Protocol configuration

Click **Change** to specify a predefined transport configuration or to create a configuration. JMS transport configurations contain generic end point, reception point, and adapter settings that can be reused.

Properties

Specify the names and values of any string properties that are required by the request for the current JMS transport configuration. These are sent as message properties through JMS. Click **Add**, **Edit** or **Remove** to modify the properties list.

WebSphere® MQ

Select **MQ** to use the IBM® WebSphere® MQ transport for the request. This page enables you to specify the SOAP action and override the settings for the WebSphere® MQ configuration selected at the test level.

Protocol configuration

Click **Change** to specify a predefined transport configuration or to create a configuration. WebSphere® MQ transport configurations contain generic queue, header, and SSL settings that can be reused.

SOAP Action

Specifies the SOAP action to be used to invoke the WebSphere® MQ request.

Override MQ protocol configuration values

Select this option to configure the fields of the WebSphere® MQ message. You can replace a subset of an MQ message descriptor with a custom format for use with other server types, specifically when using an XML message request.

Customize message header

Select this option to specify custom headers for the transport for the SOAP over MQ feature that is provided by WebSphere® MQ. This feature uses a predetermined MQ message format (RFH2), therefore, when selected, other **Message Descriptor** options are disabled.

Message descriptor

These settings replace the message descriptor and header settings of the MQ protocol configuration. Refer to WebSphere® MQ documentation for information about message descriptors.

Microsoft™ .NET

Select **Microsoft .NET** to use the Microsoft .NET Framework transport for requests based on Windows™ Communication Foundation (WCF). This page enables you to override the settings for the Microsoft™ .NET configuration selected at the test level.

Item

Click **Add** to specify the name and value of the WCF actions that are required by the service. This table is automatically generated when you import a Microsoft .NET WSDL file. Refer to the Microsoft™ .NET WCF documentation for more information.

Security for Request

In this page you can edit the security algorithm stacks that the security protocol uses. Security stacks are a set of algorithms that are executed in a given order.

Override stack

By default, you edit the security algorithm stack attached to a specific WSDL file in the WSDL Security Editor. Select this option to specify a different security algorithm stack only for the current response.

Show response stack

Clear the **Show response stack** option to hide the **Response Stack** page.

Security Algorithm Details

Click **Add**, **Insert**, or **Remove** to add or remove security algorithms in the stack. Click **Up** and **Down** to change the order of a selected algorithm in the security stack. The following security algorithms can be added to the security stack:

XML Encryption

The XML encryption security algorithm specifies how the XML document is encrypted. For details on security algorithms, refer to the web service security specification.

Actor / Role name

Specify the name of the recipient of the algorithm header element, if required.

Must understand

Select whether it is mandatory that the algorithm header is processed by the recipient, if required. The recipient is either the Actor name or the server.

Identifier type

Select the type of key identifier to be used for the encryption. The following key identifiers are available, as defined in the Web Services Security (WSS) specification X509 profile and the OASIS WSS 1.1 specification:

- ISSUER_SERIAL
- BST_DIRECT_REFERENCE
- X509_KEY_IDENTIFIER
- SKI_KEY_IDENTIFIER
- EMBEDDED_KEYNAME
- THUMBPRINT_IDENTIFIER
- ENCRYPTED_KEY_SHA1_IDENTIFIER

User XPath part selection

This enables you to specify an XPath query that describes parts of the XML document that can be subjects of the algorithm. By default, the body is the subject.

Key

Select the key used for the encryption. The details of each key vary.

- **x509 key:** This specifies the name and password of the x509 key and the keystore where it is located.
- **Raw key:** This specifies the name and the byte value of your SecretKey in hexadecimal.
- **Encrypted key:** This specifies a reference to an encrypted key that was previously defined in the security stack. Click **Insert a new encrypted key** to create a new encrypted key definition block.

Encoding Algorithm Name

Specify the encryption method to be used as defined in the XML Encryption Syntax and Processing specification.

Key Encoding Algorithm

Specify the standard algorithm for encoding the key as defined in the XML Encryption Syntax and Processing specification.

Encrypted Key

This block defines an encrypted key that can be used in an XML signature or XML encryption block. The encrypted key block must be before a block that uses the encrypted key.

Actor / Role name

Specify the name of the recipient of the algorithm header element, if required.

Must understand

Select whether it is mandatory that the algorithm header is processed by the recipient, if required. The recipient is either the Actor name or the server.

Key name

Specify the name of the encrypted key.

Identifier type

Select the type of key identifier to be used for the key. The following key identifiers are available, as defined in the the Web Service Security (WSS) specification X509 profile and OASIS WSS 1.1 specification:

- ISSUER_SERIAL
- BST_DIRECT_REFERENCE
- X509_KEY_IDENTIFIER
- THUMBPRINT_IDENTIFIER
- SKI_KEY_IDENTIFIER

Key size

Specify the size of the key in bits.

Key encoding algorithm name

Specify the algorithm to be used for encoding the key.

Keystore

Select a keystore or click **Edit Security** to define a new keystore or to manage the existing keystores.

Name

Select a key contained in the specified keystore.

Password

Type the password for the selected key name.

Custom Security Algorithm

If you want to use a Java™ class as a custom security algorithm, then use this stack element to apply the custom algorithm to the service.

Java™ Project

If you have not implemented a custom Java™ class, select **Java Project**, type a name for the new project, and click **Generate** to create a new Java™ class with the default structure for custom security implementations.



Note: If you are using IBM® Security AppScan®, this field is not available.

Implementation class

Specify the name of the class that implements the custom security algorithm. Click **Browse Class** to select an existing Java™ class from the workspace.

Properties

Use this table to send any specific properties and associated values to the custom security algorithm.

Generic service client message return details

In the generic service client, message returns are generated after a service call is successfully invoked. Message returns display the content returned by the service.

Message

This page shows the XML content of the request and provides access to data correlation. The same content is presented in three different ways.

Form

This view provides a simplified view of the message that focuses on editing the values of the XML content. Use the **Schema** menu to enable assistance with editing XML content so that the XML is valid and complies with the XSD specification.

In the **Form** view, add the XML headers that are required for standard web service calls. On the **Header** bar, click **Add** (+) to create the default XML header structure for WS-Addressing, WS-ReliableMessaging or WS-Coordination requests, or click **More** for other standards. You can enable or disable XML header elements and specify the correct values for each XML element. Checks are performed to ensure that the XML content is valid.



Note: To add XML headers to calls in IBM® Security AppScan®, add a **Static XML Headers** algorithm on the **Request Stack** tab of the request.

Tree

This view provides a hierarchical view of the XML structure of the message, including elements, namespaces, and the associated values. You can use **Add**, **Insert**, **Remove**, **Up**, and **Down** to edit the XML elements and namespaces in the tree.

Use **Skip if Empty** column to select the empty XML elements that you want to skip. This column is visible only if you selected the **Display the 'Skip if Empty' column in XML tree viewer** check box in **Window > Preferences > Test > Test editor > Service test**.


Click **Filter** to hide or show namespace, attribute, or text nodes, depending on your requirements.

Click **Allow only valid modifications** to enable smart editing, based on a specified XML schema document (XSD). To specify a set of XSD documents for the workbench, in the test navigator, right-click the project and select **Properties** and **Schema Catalog**. Disable **Allow only valid modifications** if you do not have an XSD or if you want to bypass the schema.

You can right-click an XML element to convert it to an XML fragment. This enables you to perform data correlation (use datasets and create references) on the entire XML fragment instead of only on the value.

Source

This view displays the source XML content of the message or plain text content. To format XML content, click **Format XML text**. To wrap XML content into a single line, click **Pack XML text to single line**. Similar controls are available for JSON content.

 **Important:** In the Source view, do not edit the tags that start with `SoaTag`. If you delete or change these tags, any references and substitutions in the test will be broken. You cannot recreate these tags after you delete them.

Attachments

This page lists the MIME or DIME attachments that are attached to the request. The contents of this view conform to the Multipurpose Internet Mail Extensions (MIME) or Direct Internet Message Encapsulation (DIME) specification. You can use this page to add workbench resources as MIME or DIME attachments and change properties.

The **Content ID** is the identifier that the request uses to refer to the attachments. The method for using this identifier depends on your server requirements.

MIME or DIME

Select whether the attachment conforms to the Multipurpose Internet Mail Extensions (MIME) or Direct Internet Message Encapsulation (DIME) specification

Use MTOM transmission mechanism

By default, the request uses SOAP Messages with Attachments (SwA) to handle attachments. Select this option to handle attachments with the SOAP Message Transmission Optimization Mechanism (MTOM).

Response Properties

This page lists the names and values of properties of the response.

WSDL security editor reference

With the Web Service Description Language (WSDL) security editor you can create and edit security configurations for a WSDL file.

Keystores

In this page, you can edit the keystores that are used for the WSDL file. The keystore contains the public and private keys that are required for the specified security protocol.

Defined Keystores

Click Add or Remove to add or remove keystore files from the workbench.

Keystore Details

This specifies the location and file name of the selected keystore. Click **Browse** to select a different file.

Name

This specifies the name of the keystore. This name is used throughout the test instead of the file name.

File

Click **Browse** to specify a keystore file containing a valid server certificate. The following formats are checkmark_perf.jpg:

- KS
- JKS
- JCEKS
- PKCS12 (p12 or PFX)
- PEM

Password

If the keystore file is encrypted, type the required password.

Security Stacks

In this page you can edit the security algorithm stacks that the security protocol uses. Security stacks are a set of algorithms that are executed in a given order.

Security Stacks

Click **Add**, **Remove**, or **Rename** to add, remove, or rename the security stacks that are associated with the WSDL file.

Security Algorithm Details

Click **Add**, **Insert**, or **Remove** to add or remove security algorithms in the stack. Click **Up** and **Down** to change the order of a selected algorithm in the security stack. The following security algorithms can be added to the security stack:

Time Stamp

The time stamp security algorithm adds time stamp information to the XML document in the response. For details on security algorithms, refer to the web service security specification.

Actor / Role name

Specify the name of the recipient of the algorithm header element, if required.

Must understand

Select whether it is mandatory that the algorithm header is processed by the recipient, if required. The recipient is either the Actor name or the server.

Expiration delay

Specify the delay after which the time stamp expires.

Millisecond precision

Select this option to produce a time stamp that uses millisecond precision instead of the default (1/100th second).

User name token

The user name token security algorithm adds a user name token to the XML document in the message. For details on security algorithms, refer to the web service security specification.

Actor / Role name

Specify the name of the recipient of the algorithm header element, if required.

Must understand

Select whether it is mandatory that the algorithm header is processed by the recipient, if required. The recipient is either the Actor name or the server.

Name

Type the name of the user.

Password

Type the password of the user.

Password type

Specify the password type for the security algorithm as defined in the Web Services Security UsernameToken profile.

Use nonce

Select this check box to add the Nonce element to the User Name Token XML code. In most cases, the Nonce ID is required.

Use created

Select this check box to add current timestamp to the Created XML element in the User Name Token XML.

XML Encryption

The XML encryption security algorithm specifies how the XML document is encrypted. For details on security algorithms, refer to the web service security specification.

Actor / Role name

Specify the name of the recipient of the algorithm header element, if required.

Must understand

Select whether it is mandatory that the algorithm header is processed by the recipient, if required. The recipient is either the Actor name or the server.

Identifier type

Select the type of key identifier to be used for the encryption. The following key identifiers are available, as defined in the Web Services Security (WSS) specification X509 profile and the OASIS WSS 1.1 specification:

- ISSUER_SERIAL
- BST_DIRECT_REFERENCE
- X509_KEY_IDENTIFIER
- SKI_KEY_IDENTIFIER
- EMBEDDED_KEYNAME
- THUMBPRINT_IDENTIFIER
- ENCRYPTED_KEY_SHA1_IDENTIFIER

User XPath part selection

This enables you to specify an XPath query that describes parts of the XML document that can be subjects of the algorithm. By default, the body is the subject.

Key

Select the key used for the encryption. The details of each key vary.

- **x509 key**: This specifies the name and password of the x509 key and the keystore where it is located.
- **Raw key**: This specifies the name and the byte value of your SecretKey in hexadecimal.
- **Encrypted key**: This specifies a reference to an encrypted key that was previously defined in the security stack. Click **Insert a new encrypted key** to create a new encrypted key definition block.

Encoding Algorithm Name

Specify the encryption method to be used as defined in the XML Encryption Syntax and Processing specification.

Key Encoding Algorithm

Specify the standard algorithm for encoding the key as defined in the XML Encryption Syntax and Processing specification.

XML Signature

The XML signature security algorithm specifies how the XML document is signed. For details on security algorithms, refer to the web service security specification.

Actor / Role name

Specify the name of the recipient of the algorithm header element, if required.

Must understand

Select whether it is mandatory that the algorithm header is processed by the recipient, if required. The recipient is either the Actor name or the server.

Security token

Select the type of key identifier to be used for the signature. The following key identifiers are available, as defined in the the Web Service Security (WSS) specification X509 profile and OASIS WSS 1.1 specification:

- ISSUER_SERIAL
- BST_DIRECT_REFERENCE
- X509_KEY_IDENTIFIER
- SKI_KEY_IDENTIFIER
- KEY_VALUE
- USER_NAME_TOKEN
- CUSTOM_SYMM_SIGNATURE

In addition, the following identifiers are available when the signature is based on a UsernameToken profile:

- USER_NAME_TOKEN
- CUSTOM_SYMM_SIGNATURE

User XPath part selection

Specify an XPath query that describes parts of the XML document that can be the subjects of the algorithm. By default, the body is the subject. Click the **XPath Helper** button to build the Xpath expression.

Key

Select the key used for the encryption. The details of each key vary.

- **x509 key:** This key specifies the name and password of the x509 key and the keystore where it is located.
- **User name token key:** This specifies a user name and password for the signature.
- **Encrypted key:** This specifies a reference to an encrypted key that was previously defined in the security stack. Click **Insert a new encrypted key** to create a new encrypted key definition block.

Signature algorithm name

Specify the signature method algorithm as described in the XML Signature Syntax and Processing specification.

Canonicalization

Specify the canonicalization method to be used as described in the XML Signature Syntax and Processing specification.

Digest algorithm method

Specify which digest method to be used based on the algorithm method used on the server side.

Inclusive namespaces

Specify whether the canonicalization is exclusive as described in the Exclusive XML Canonicalization specification.

Custom Security Algorithm

If you want to use a Java™ class as a custom security algorithm, then use this stack element to apply the custom algorithm to the service.

Java™ Project

If you have not implemented a custom Java™ class, select **Java Project**, type a name for the new project, and click **Generate** to create a new Java™ class with the default structure for custom security implementations.



Note: If you are using IBM® Security AppScan®, this field is not available.

Implementation class

Specify the name of the class that implements the custom security algorithm. Click **Browse Class** to select an existing Java™ class from the workspace.

Properties

Use this table to send any specific properties and associated values to the custom security algorithm.

Security Considerations

This document describes the actions that you can take to ensure that your installation is secure, customize your security settings, and set up user access controls.

- [Enabling secure communication between multiple applications on page cmvi](#)
- [Ports, protocols, and services on page cmvi](#)
- [Customizing your security settings on page cmvi](#)
- [Privacy policy considerations on page cmvi](#)
- [Security limitations on page cmvi](#)

Enabling secure communication between multiple applications

The workbench computer that controls the execution of the test communicates with the remote agent computers. The agents apply load for IBM® Rational® Service Tester for SOA Quality. The communication can be secure or nonsecure. By default, the tool use nonsecure communication. Also, if a workbench computer uses a self-signed certificate, it cannot be changed. Agent computers are trusting.

- Configuring port numbers for agents

Ports, protocols, and services

The Majordomo service running on remote agents must run with administrator or super user credentials, which means that the test execution it supports has full privileges on the test computer where it resides. Product communication uses HTTP and HTTPS. Ports are configurable.

- Configuring port numbers for agents

Customizing your security settings

Datasets can be encrypted and access controlled by password that is difficult, but not impossible, to break.

- [Encrypted datasets overview on page 305](#)

Privacy policy considerations

This software offering does not use cookies or other technologies to collect personally identifiable information. For additional information on cookies, see the [Notices on page cmvii](#) topic.

Security limitations

Passwords are stored using Eclipse mechanisms that are difficult but not impossible to break.

Workbench and agent communication is encrypted but not absolutely safe from impersonation attack.

Notices

This document provides information about copyright, trademarks, terms and conditions for the product documentation.

© Copyright IBM Corporation 2001, 2016 / © Copyright HCL Technologies Limited 2016, 2023

This information was developed for products and services offered in the US.

IBM® may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM® representative for information on the products and services currently available in your area. Any reference to an IBM® product, program, or service is not intended to state or imply that only that IBM® product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM® intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM® product, program, or service.

IBM® may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM® Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM® may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM® websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM® product and use of those websites is at your own risk.

IBM® may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM® under terms of the IBM® Customer Agreement, IBM® International Program License Agreement or any equivalent agreement between us.

The performance data discussed herein is presented as derived under specific operating conditions. Actual results may vary.

Information concerning non-IBM® products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM® has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM® products. Questions on the capabilities of non-IBM® products should be addressed to the suppliers of those products.

Statements regarding IBM®'s future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM®, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM®, therefore, cannot guarantee or imply reliability,

serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM® shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work must include a copyright notice as follows:

© (your company name) (year).

Portions of this code are derived from IBM Corp. Sample Programs.

© Copyright IBM Corp. 2000, 2017.

Trademarks

IBM®, the IBM® logo, and ibm.com® are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM® or other companies. A current list of IBM® trademarks is available on the web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

Terms and conditions for product documentation

Permissions for the use of these publications are granted subject to the following terms and conditions.

Applicability

These terms and conditions are in addition to any terms of use for the IBM® website.

Personal use

You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of IBM®.

Commercial use

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM®.

Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM® reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM®, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM® MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

IBM Online Privacy Statement

IBM Software products, including software as a service solutions, (Software Offerings) may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user, or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering use of cookies is set forth below.

This Software Offering does not use cookies or other technologies to collect personally identifiable information.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, see IBM's Privacy Policy at <http://www.ibm.com/privacy> and IBM's Online Privacy Statement at <http://www.ibm.com/privacy/details> in the section entitled Cookies, Web Beacons and Other Technologies, and the IBM Software Products and Software-as-a-Service Privacy Statement at <http://www.ibm.com/software/info/product-privacy>.

Index

A

- Access web reports remotely 468
- accessibility
 - keyboard shortcuts (testing) 856
- activation kits 66
 - purchasing 65
- adapters
 - Rational Quality Manager 114, 117, 119, 120, 122
- agent computers
 - deleting locations 286
 - renaming locations 284
- assets
 - testing shared assets 123
- asynchronous calls
 - adding to web service tests 275
- asynchronous requests
 - adding to web service tests 273
- asynchronous service 271
- attachment verification points
 - adding 248, 251
- authentication
 - SSL 348

B

- binary content
 - viewing 362
- byte counters
 - types 522

C

- call counters
 - services 558
- call details
 - generic service client 871, 888, 892
- callback
 - adding to web service tests 274
- calls
 - web service tests 252
- Citrix
 - counters
 - image synchronization 550
 - overview 548
 - user action 555
 - window 555
 - window synchronization 548
 - reports 507
 - timer counters 553
- Citrix Performance Report 507
- ClearCookies class 416
- clients
 - generated clients in service tests 211
- clock skew
 - correcting 454
- coexistence
 - product installation 49
- command line
 - create config file 370
 - schedule runs 395
- ComputerSpecific class 417, 418
- conditions
 - error handling 860
- configuration
 - SSL authentication 348
 - WebSphere MQ transport 343
- configurations
 - service testing
 - SOAP security 208, 341

- service tests 205
- web service
 - Custom security algorithms 268
 - SOAP security 257
 - WS-Addressing 270
 - web service security 256
- web services
 - WS-Policy 259
- contain verification points
 - web service 244
- conventions
 - installation 47
- cookies
 - setting and clearing for virtual users 416
- CountAllIterations class 414
- counter
 - manage counters 453
- counters
 - Citrix 548
 - image synchronization 550
 - user action 555
 - window 555
 - window synchronization 548
 - Citrix timers 553
 - HTTP 522
 - byte 522
 - page performance 523
 - run 534
 - test 536
 - SAP 541
 - screen (SAP) 541
 - services
 - call 558
 - overview 558
 - verification points 569
 - socket API 579, 579
 - transaction 537
- CountUserIterations class 414
- creating
 - Service stubs 334
- CSV format
 - exporting report counters
 - automatically 372
 - exporting results 457
- Custom code
 - debug 429
- custom counters
 - test execution services 424
- custom Java code
 - code execution counts 414
 - controlling loops 408
 - creating 402
 - custom counters 424
 - deleting 286
 - determining where a test is running 417, 418
 - extracting strings 420
 - interfaces and classes 405
 - migrating 436
 - overview 402
 - performance 407
 - printing input arguments to a file 413
 - renaming 284
 - retrieving the maximum JVM heap size 421
 - retrieving virtual user IP address 413
 - running a program with a test 422

- setting and clearing cookies for virtual users 416
- statistics 426
- transactions 426
- verification points 428
- Custom security algorithms
 - web service 268

D

- data correlation
 - custom code example 420
 - description 315
 - disabling 332
 - manual 320
 - multiple fields 327
 - overview 315, 318
 - problem identification 329
 - regenerating tests 332
 - rules 858
 - substituting 321
 - tests
 - data correlation 315
 - multiple-field data correlation 327
 - troubleshooting 331
 - viewing 317
- data sources
 - resource monitoring 861
- DataAreaLockException (test execution services) 405
- datasets
 - creating in test 288
 - creating in workspace 292
 - deleting 286
 - editing 300
 - encrypting 306
 - encryption 305
 - navigating to tests 307
 - options 296
 - organizing 241
 - overview 288
 - removing encryption 307
 - renaming 284
 - segmented
 - row assignment 296
 - test references 297
 - typical 296
 - viewing in tests 299
- Debug custom code 429
- default reports
 - changing 453
- defects
 - submitting 126
- digital certificates
 - authentication in tests 237
 - creating 232, 234
 - overview 229
 - types 231
 - using in tests 237

E

- Eclipse
 - fully-enabled
 - start-up 36
 - installing with Eclipse instance 49
 - streamlined 36
- encrypting
 - recording session data 239
- Entrust

- overview 237
 - equal verification points
 - web service 243
 - error handling
 - conditions 860
 - ExecTest class 422
 - Export
 - Event Console Output 469
 - exporting
 - performance test assets 279
 - performance test projects 280
 - report counters
 - to CSV format automatically 372
 - reports
 - to .view file 463
 - to HTML format 456
 - results
 - to CSV format 457
- F**
- fields
 - data correlation 327
 - file attachments
 - opening 365
 - files
 - printing input arguments to 413
 - floating licenses
 - purchasing 65
 - functional reports
 - generating 439
- G**
- generated clients
 - recording service tests 211
 - generic service client 338
 - call details 871, 888, 892
 - message return details 898
 - overview 31
 - graphs
 - customizing 451
- H**
- HTML format
 - exporting reports 456
 - HTTP
 - counters overview 522
 - run counters 534
 - HTTP endpoint
 - service call 352
 - HTTP proxies
 - recording web service tests 208
 - HTTP transport
 - services 338
- I**
- IARM (test execution services) 405
 - ICustomCode2 (test execution services) 405
 - IDataArea (test execution services) 405
 - IEngineInfo (test execution services) 405
 - ILoopControl (test execution services) 405
 - image synchronization
 - counters 550
 - importing
 - reports
 - to .view file 463
 - installation
 - extending an Eclipse instance 49
 - launchpad program 51
 - locations 48
 - terminology 47
 - using the launchpad program 51
- Installation Manager**
- overview 48
- installing packages**
- Installation Manager 48
- installing products**
- coexistence 49
- integrating**
- Rational Team Concert 124
- invoke**
- HTTP service call 352
 - JMS service call 354
 - service call with a WSDL file 361
- IP addresses**
- retrieving from virtual user 413
- IPDLogManager (test execution services) 405**
- IScalar (test execution services) 405**
- IStat (test execution services) 405**
- IStatistics (test execution services) 405**
- IStatisticsManager (test execution services) 405**
- IStatTree (test execution services) 405**
- ITCAM**
- response time breakdown 863
- ITestExecutionServices (test execution services) 405**
- ITestInfo (test execution services) 405**
- ITestLogManager (test execution services) 405**
- IText (test execution services) 405**
- ITime (test execution services) 405**
- ITransaction (test execution services) 405**
- IVirtualUserInfo (test execution services) 405**
- J**
- Jaeger
 - 155
 - Jaeger logs
 - testlogs 156
 - Java
 - test execution services 402
 - JMS endpoint
 - service call 354
 - JMS transport
 - services 342
 - JVM heap size
 - retrieving maximum 421
 - JVM_Info class 421
- K**
- keyboard shortcuts
 - testing 856
- L**
- launch configurations
 - default 366
 - test settings 367
 - launchpad program
 - installing the product 51
 - starting installations 51
 - license expiration terms
 - viewing 67
 - license types
 - viewing 67
 - licenses
 - managing 63
 - purchasing 65
 - testing 65
 - viewing information on packages 67
 - Licenses
 - product enablement 66
- locations**
- deleting 286
 - organizing 241
 - renaming 284
 - user groups 870
- logs 465**
- exporting test events 469
 - levels
 - services performance tests 205
 - stub server activity 336
 - viewing test events 467
- loops**
- controlling 408
 - searching tests 277
- M**
- manage counters 453
 - managing licenses
 - overview 63
 - memory
 - increasing 401
 - message content
 - viewing 362
 - message return details
 - generic service client 898
 - message returns
 - web services
 - comparing contents 465
 - Microsoft .NET transport
 - services 346
 - migration
 - custom Java code 436
 - performance testing assets 67
 - modifying packages
 - installation manager 48
- N**
- notification-based services
 - testing 271
- O**
- obfuscating
 - recording session data 239
 - OpenSSL
 - digital certificates 232
 - OutOfScopeException (test execution services) 405
- P**
- package groups
 - coexistence 49
 - installation locations 48
 - page counters
 - types 523
 - page percentile reports
 - performance testing 502
 - pages
 - searching tests 277
 - test report elements 499
 - viewing test data 299, 307
 - viewing test request data 307
 - ParseResponse class 420
 - percentile reports
 - performance testing 511
 - Performance report 482
 - performance testing
 - reports
 - overview (SAP) 503
 - socket API 33
 - TN3270 applications 33
 - web service
 - overview 29, 333

- performance tests
 - socket reports overview 520
- ports
 - configuring for different locations 369
- preferences
 - changing recording (HTTP) 540
 - test generation
 - changing web service preferences 229
- PrintArgs class 413
- problem identification
 - data correlation 329
- problems
 - troubleshooting 583
- product enablement
 - overview 66
- projects
 - creating 204
 - deleting 286
 - renaming 284
- properties
 - schedules 864

R

- Rational Common Licensing 66
- Rational
 - Performance Tester
 - licenses 65
 - migrating from earlier releases 67
 - Quality Manager
 - adapter 120, 122
 - configuring the adapter 114
 - running the adapter 119
 - starting the adapter 117
 - testing shared assets 123
 - Service Tester for SOA Quality
 - migrating from earlier releases 67
 - Team Concert
 - defect tracking 126
 - integrating 124
- raw transaction data
 - viewing 362
- recording
 - changing preferences
 - HTTP 540
 - sensitive data 239
 - service tests
 - generated clients 211
 - overview 204
 - web service tests
 - BPEL resources 222
 - creating manually 224
 - HTTP proxies 208
 - WebSphere MQ tests
 - creating manually 225
 - XML call tests
 - creating manually 228
- recordings
 - regenerating tests 240
- references
 - test datasets 307
- Reliable messaging 364
- remote locations
 - configuring differing ports for tests 369
 - memory allocation increases 401
- remote WSDL files
 - synchronization 363
- replacing text
 - searches 277
- reports
 - .view format exports 463

- changing default 453
- Citrix
 - performance report 507
 - types 507
- comparisons 437, 437
- counters
 - CSV format automatic exports 372
 - customizing graphs 451
 - displaying 467
 - filtering results 449
 - functional test 439
 - HTML format exports 456
 - HTTP 482
 - migration 67
 - page elements 499
 - performance (SAP) 503
 - performance testing percentiles 502, 511
 - reports
 - HTTP performance report 482
 - SAP Performance 504
 - service
 - performance report 512
 - socket performance 520
 - socket performance tests 520
 - synchronization point 473
 - Transaction 475, 477
 - Transaction Net Server Time 478
 - verification points
 - citrix 510
 - HTTP 503
 - SAP 506
 - web service 517
 - web service
 - types 512
- requirements
 - software installation 46
- resource monitoring
 - data sources 861
- resources
 - organizing 241
- response time breakdown
 - data sources 863
- response times
 - reports 478
- responses
 - adding to service tests 254
 - searching tests 277
- results
 - CSV format exports 457
 - deleting 286
 - filtering 449
 - organizing 241
 - renaming 284
- rules
 - data correlation 858
- run counters
 - types 534
- runs
 - displaying reports after 467

S

- SAP
 - Performance reports 504
 - screen counters 541, 541
- schedule runs
 - configuring differing ports for tests 369
 - launch configurations 366
 - out-of-memory errors 401
 - overview 366
 - random order test elements 283

- schedules
 - command-line starts 395
 - deleting 286
 - disabling portions of 281
 - keyboard shortcuts 856
 - launch configuration settings for tests 367
 - migration 67
 - organizing 241
 - properties 864
 - renaming 284
 - run configurations 369
- screen counters
 - SAP 541
- security
 - keystores 900
 - services 900
 - stacks 900
 - Web Service Description Language editor 900
- selectors
 - adding to tests 283
- sending
 - service request with a WSDL file 350
 - WebSphere MQ service request 355
- service calls 31
- service requests
 - viewing content 362
- service responses
 - adding to service tests 253
- services
 - configuring environment 208, 341
 - counters
 - call 558
 - overview 558
 - verification points 569
 - file attachments 365
 - HTTP call 352
 - HTTP transport configuration 338
 - invoking calls 338
 - JMS call 354
 - JMS transport configuration 342
 - Microsoft .NET transport configuration 346
 - recording tests
 - generated clients 211
 - security editor reference 900
 - WebSphere MQ request 355
 - WSDL file 350, 361
 - SetCookieFixedValue class 416
- shared assets
 - testing 123
- shared resources directories
 - installation locations 48
- shortcuts
 - keyboard
 - testing 856
- simulating
 - services 334
- SOAP security
 - creating configurations 257
- socket API
 - counters 579
 - performance testing 33
 - screen counters 579
- socket API tests
 - performance reports 520
- software installation
 - requirements 46
- split points
 - inserting during recording 240

- SSL authentication
 - configuration 348
- stages 437
- statistics
 - CSV format exports 457
- statistics sample interval 205
- StatType (test execution services) 405
- strings
 - extracting from input arguments 420
- stub servers
 - logging activity 336
- stubbing
 - services 334
- substitutions
 - data correlation 321
- Synchronization Point report
 - performance testing 473
- synchronizing
 - remote WSDL files 363

T

- terminology
 - product installation 47
- test assets
 - importing into Rational Quality Manager 122
 - organizing 241
- test counters
 - types 536
- test data
 - recorrelating 332
 - sources 318
- test elements
 - running in random order 283
 - selecting multiple types 276
- test execution services
 - code execution counts 414
 - custom counters 424
 - determining where a test is running 417, 418
 - extracting strings 420
 - interfaces and classes 405
 - migrating Java code 436
 - overview 402, 402
 - printing input arguments to a file 413
 - retrieving the maximum JVM heap size 421
 - retrieving virtual user IP address 413
 - running a program with a test 422
 - setting and clearing cookies for virtual users 416
 - statistics 426
 - transactions 426
 - verification points 428
- test generation
 - changing preferences
 - web service 229
- testing
 - keyboard shortcuts 856
 - overview 271
 - services
 - guidelines 205
- tests
 - adding custom Java code 402
 - annotating during recording 238
 - annotations
 - adding during recording 238
 - automating 370, 370
 - configuring different ports for 369
 - customizing 405
 - dataset references 297

- datasets 305, 306, 307
- declaring variables 310
- deleting 286
- disabling portions of 281
- editing
 - overview 242
 - service overview 242
 - web service 242
 - web service security 256
- extending
 - controlling loops 408
 - custom Java code 402
- failures
 - virtual user memory allocation 401
- keyboard shortcuts 856
- logs 205
 - exporting 469
 - viewing 467
- manual data correlation 320
- migrating custom Java code 436
- migration 67
- organizing 241
- page searches 277
- regenerating from recordings 240
- renaming 284
- results
 - settings 367
- running locally 366
- searching overview 276
- services
 - recording with generated clients 211
- splitting during recording 240
- test execution services
 - code execution counts 414
 - custom counters 424
 - determining where a test is running 417, 418
 - extracting strings 420
 - printing input arguments to a file 413
 - retrieving the maximum JVM heap size 421
 - retrieving virtual user IP address 413
 - running a program with a test 422
 - setting and clearing cookies for virtual users 416
 - statistics 426
 - transactions 426
- variables 308
- web service
 - adding calls 252
 - adding responses 254
 - creating manually 224
 - recording 222
- web services
 - adding asynchronous calls 275
 - adding asynchronous requests 273
 - adding callbacks 274
 - adding responses 253
 - recording with HTTP proxies 208
- WebSphere MQ
 - creating manually 225
- XML call
 - creating manually 228
- time offset
 - correcting 454
- timer counters
 - Citrix 553
- Tivoli Composite Application Manager
 - response time breakdown 863
- Tivoli Monitoring for Transaction Performance
 - response time breakdown 863

- response time breakdown 863
- TN3270 applications
 - performance testing 33
- tokens
 - extracting from input arguments 420
- transaction counters
 - types 537
- Transaction Net Server Time Percentile report
 - performance testing 478
- Transaction Percentile report
 - performance testing 477
- Transaction report
 - performance testing 475
- TransactionException
 - test execution services 405
- transactions
 - searching tests 277
- troubleshooting
 - data correlation 331
 - performance testing 583

U

- updating packages
 - Installation Manager 48
- user groups
 - locations 870
 - properties 870

V

- variable
 - initialize variable from XML 312
- variables
 - assigning 310
 - in tests 308
 - initializing 310
- verification point counters
 - services 569
- verification points
 - Citrix
 - reports 510
 - counters in socket API tests 579
 - HTTP
 - reports 503
 - SAP
 - counters 541
 - reports 506
 - searching tests 277
 - web service
 - adding attachment verification points 248, 251
 - adding Xpath query 247
 - checking returned messages 243, 244
 - overview 243
 - reports 517
- view file
 - exporting reports 463
 - importing reports 463
- virtual users
 - counting code runs 414
 - datasets 297
 - retrieving IP addresses 413
 - setting and clearing cookies 416

W

- WCF transport
 - services 346
- Web reports
 - access remotely 468
- web service
 - adding calls to tests 252

- attachment verification points 248, 251
- configuration 205
- creating tests manually 224
- Custom security algorithms 268
- performance testing 29, 333
- reports 512
- security configurations 256
- security editor overview 256
- SOAP security configurations 257
- test editor overview 242
- verification points
 - checking returned messages 243, 244
 - overview 243
- Xpath query verification points 247
- web Service
 - WSDL syntax compliance 207
- Web Service Description Language
 - security editor 900
 - syntax compliance for JMS web services 207
- Web Service Performance Report 512
- web service responses 247
- web services
 - asynchronous service testing 271
 - message returns
 - comparing contents 465
 - recording tests
 - HTTP proxies 208
 - WS-Addressing 270
 - WS-Policy 259
- WebSphere MQ
 - creating tests manually 225
- WebSphere MQ endpoint
 - service request 355
- WebSphere MQ transport
 - services 343
- weighted blocks
 - renaming 284
- window counters
 - Citrix 555
- window synchronization
 - counters 548
- workspaces
 - copying projects 280
 - copying test assets 279
- WS-Addressing 364
 - creating configurations 270
- WS-Coordination 364
- WS-Policy
 - creating configurations 259
- WS-RM 364
- WSDL
 - sending a service request 350
- WSDL files
 - remote WSDL files 363
- WSDLfile
 - invoking a service call 361

X

- XML call
 - creating tests manually 228
- XML headers 364
- Xpath query verification points 247