

IBM Z Workload Scheduler
Scheduling End-to-end with z-centric Capabilities
Version 10.2 SPE (Revised December 2025)

Note

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

This edition applies to version 10, release 2, modification level 0 of IBM Z Workload Scheduler (program number 5698-T09) and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

List of Figures.....	vi
List of Tables.....	vii
About this publication.....	ix
Conventions.....	ix
How to read syntax diagrams.....	ix
Chapter 1. Configuring.....	13
Overview.....	13
Advantages of job types with advanced options.....	17
Configuring the agent.....	18
Configuring proxy properties [ITA_Env].....	21
Configuring log message properties [JobManager.Logging.cclog].....	22
Configuring trace properties when the agent is stopped [JobManager.Logging.cclog].....	23
Trace configuration for the agent.....	25
Configuring common launchers properties [Launchers].....	28
Configuring properties of the native job launcher [NativeJobLauncher].....	30
Configuring properties of the Java™ job launcher [JavaJobLauncher].....	33
Configuring properties of the Resource advisor agent [ResourceAdvisorAgent].....	34
Configuring properties of the System scanner [SystemScanner].....	36
Configuring properties of event-driven workload automation [EventDrivenWorkload].....	37
Regular agent maintenance.....	38
Configuring Z controller.....	39
Defining destinations with the ROUTOPTS statement.....	39
Chapter 2. Customizing the HTTP connections	42
Customizing HTTP connection with HTTPOPTS statement.....	42
Customizing the http timeout on the Dynamic Workload Console.....	42
Configuring the heartbeat checking interval for z-centric agents and dynamic domain managers.....	43
Configuring the connection between the z-centric agent and controller through a proxy.....	43
Starting and stopping HTTP subtasks.....	44
Chapter 3. Defining user IDs and passwords with the USRREC statement.....	46
USRREC.....	46
Chapter 4. Defining variables and passwords for local resolution.....	49
Creating and managing variables and passwords on the agents.....	49
Specifying variables and passwords in the JOBREC statement.....	52
Specifying local variables and passwords in job types with advanced options.....	53
Chapter 5. Defining and managing IBM Z Workload Scheduler Agent workstations in the database.....	55
Creating.....	55
Modifying.....	56
Browsing.....	56
Deleting.....	57
Listing.....	57
Managing the status of the workstation in the plan.....	57
Automatic and manual changes to the status of IBM Z Workload Scheduler Agent workstation.....	58
WSSTAT.....	59
Browsing system information of IBM Z Workload Scheduler Agent workstations.....	62
Chapter 6. Defining and managing dynamic workstations.....	63
Creating broker workstations.....	64
Defining a backup dynamic domain manager.....	65
Creating pools and dynamic pools.....	66
Submitting jobs to pools or dynamic pools.....	66
Submitting jobs by reference.....	67
Chapter 7. Setting the security features.....	68
Customizing the SSL connection with the Z controller when using your certificates (SAF).....	68
Customizing the SSL connection with the Z controller when using your certificates (USS).....	71
Enabling FIPS compliance over IBM Z Workload Scheduler server SSL secured connection.....	72
Encrypting the user password.....	73
Chapter 8. Defining and managing jobs.....	74
Defining jobs from the Dynamic Workload Console.....	74
Defining a job in the JOBLIB data set.....	74
JOBREC.....	75
Syntax diagram for database job types.....	77
Syntax diagram for file transfer job types.....	85
Syntax diagram for web service job types.....	91
Syntax diagram for Java job type.....	93
Syntax diagram for xajob job type.....	95
Syntax diagram for J2EE jms job type.....	97
Syntax diagram for native job type.....	99
Using variable substitution.....	103
Defining file dependencies to check for file changes.....	103
Maintaining the filewatch log database.....	106
Checking for file changes with filemonitor.....	107

Promoting critical jobs.....	113	result in a core dump or in a resource temporarily unavailable message.....	158
Killing operations.....	113	Appendix A. Configure to schedule job types with advanced options.....	159
ISPF.....	114	Logs for job types with advanced options.....	159
PIF.....	114	Appendix B. Managing return codes.....	161
BCIT.....	114	Appendix C. Modifying the workstation from fault-tolerant agent to z-centric agent.....	163
OCL.....	114	Appendix D. Integrating with SAP and PeopleSoft.....	165
Dynamic Workload Console.....	115	Access method interface.....	166
Chapter 9. Administering IBM i dynamic environment... 116		Method command line syntax.....	166
Scheduling on IBM i.....	116	Method response messages.....	168
The agent joblog and TWSASPOOLS environment variable.....	117	Method options file.....	169
Child job monitoring on IBM i agents.....	117	Running methods.....	172
The agent return code retrieval.....	121	Launch job task (LJ).....	172
Controlling the job environment with the user return code.....	121	Manage job task (MJ).....	173
Alternative method to set the user return code.....	122	Access methods.....	174
Configuring the agent on IBM i systems.....	123	Installing and configuring the access methods.....	174
Configuring log message properties [JobManager.Logging.cclg].....	124	Access method for PeopleSoft.....	186
Configuring trace properties when the agent is stopped [JobManager.Logging.cclg].....	126	Common serviceability for the access methods.....	200
Trace configuration for the agent.....	127	Integration with SAP	210
Configuring common launchers properties [Launchers].....	130	Introducing IBM Workload Scheduler for SAP....	210
Configuring properties of the native job launcher [NativeJobLauncher].....	132	Access method for SAP.....	213
Configuring properties of the Java™ job launcher [JavaJobLauncher].....	136	Scheduling jobs on IBM Workload Scheduler from SAP Solution Manager.....	387
Configuring properties of the Resource advisor agent [ResourceAdvisorAgent].....	136	Appendix E. Collecting job metrics.....	398
Configuring properties of the System scanner [SystemScanner].....	138	Job metrics queries for DB2 for zOS.....	398
Configuring to schedule job types with advanced options.....	139	Job metrics queries for DB2.....	398
Customizing the SSL connection between IBM i agents and the Z controller when using your certificates.....	140	Notices.....	ccxcix
Chapter 10. Monitoring and recovering jobs..... 142		Index.....	403
Job status notifications.....	142		
Setting up and running automatic recovery for jobs.....	142		
Automatic-recover-control statement for IBM Z Workload Scheduler Agent scripts.....	143		
Chapter 11. Running a script when a job completes..... 150			
Chapter 12. Viewing the job logs from the host..... 151			
Customizing the job log header.....	151		
Collecting job logs with Output collector.....	153		
Activating and configuring Output collector.....	155		
Refreshing destinations from Output collector.....	156		
Chapter 13. Troubleshooting..... 158			
On AIX operating systems the concurrent submission of one hundred or more jobs on the same agent can			

List of Figures

Figure 1: Static end-to-end environment with Java™ run time installed in which you can run job types with advanced options.....	13	Figure 22: The Table Criteria panel.....	314
Figure 2: Dynamic end-to-end environment to run also job types with advanced options.....	14	Figure 23: Job definition syntax.....	320
Figure 3: Process flow.....	15	Figure 24: Dynamic Workload Console - Table of results..	325
Figure 4: EQQWCGEP - Creating general information about an IBM Z Workload Scheduler Agent workstation.....	55	Figure 25: Dynamic Workload Console - Details of a process chain job.....	326
Figure 5: EQQWMGEP - Modifying an IBM Z Workload Scheduler Agent workstation description.....	56	Figure 26: Command syntax.....	340
Figure 6: EQQWBGEP - Browsing an IBM Z Workload Scheduler Agent workstation description.....	56	Figure 27: Command syntax.....	345
Figure 7: EQQWDGEP - Deleting an IBM Z Workload Scheduler Agent workstation description.....	57	Figure 28: Managing high priority IDocs overview.....	352
Figure 8: EQQWWSEP - Specifying list selection criteria for IBM Z Workload Scheduler Agent workstations.....	57	Figure 29: A monitor and its MTEs - © SAP AG 2009. All rights reserved.....	363
Figure 9: EQQSWIZC - Browsing an IBM Z Workload Scheduler Agent workstation system information.....	62	Figure 30: Name and description of an MTE - © SAP AG 2009. All rights reserved.....	365
Figure 10: EQQWCGEP - Creating general information about a broker workstation.....	65	Figure 31: Command syntax.....	370
Figure 11: EQQWCEOD - Defining end-to-end options.....	65	Figure 32: Command syntax.....	371
Figure 12: Job log retrieval with Output collector.....	154		
Figure 13: Defining an Extended Agent workstation.....	182		
Figure 14: Defining an Extended Agent job for end-to-end scheduling.....	185		
Figure 15: Command syntax.....	253		
Figure 16: The Variant List panel.....	258		
Figure 17: The Variant Information page of the Variant List panel.....	259		
Figure 18: Job definition syntax.....	264		
Figure 19: The Raise Event panel.....	278		
Figure 20: Job definition syntax.....	284		
Figure 21: The Table Criteria panel.....	311		

List of Tables

Table 1: Installation and configuration steps for setting up an end-to-end with z-centric capabilities environment.....	16
Table 2: Agent configuration parameters.....	38
Table 3: Jobs and their keywords.....	76
Table 4: Configuration files for job types with advanced options.....	139
Table 5: Job status change notifications.....	142
Table 6: The predefined variables of the sample job log header template.....	152
Table 7: Configuration files for job types with advanced options.....	159
Table 8: Method command task options.....	167
Table 9: Launch job task (LJ) messages.....	173
Table 10: How to complete the extended agents definition.....	179
Table 11: Roles and responsibilities in Access method for PeopleSoft.....	187
Table 12: Psagent access method options.....	189
Table 13: Task string parameters for PeopleSoft jobs.....	197
Table 14: Relationship between the run status, the distribution status, and the IBM Workload Scheduler job status.....	199
Table 15: Relationship between the run status and the IBM Workload Scheduler job status.....	200
Table 16: Job states and return codes for the PeopleSoft access method.....	203
Table 17: IBM Workload Scheduler for SAP features.....	211
Table 18: Roles and responsibilities in IBM Workload Scheduler for SAP.....	214
Table 19: Access keywords for activities with SAP scheduling objects.....	216
Table 20: ABAP/4 modules installed	223
Table 21: ABAP/4 modules contents.....	224
Table 22: r3batch global configuration options.....	231
Table 23: r3batch local configuration options.....	234
Table 24: r3batch common configuration options.....	237
Table 25: Placeholders and counters for extended variants.....	260
Table 26: Task string parameters for SAP jobs.....	265
Table 27: Status transitions in IBM Workload Scheduler (internal status) and the corresponding SAP R/3 status...	276
Table 28: Task string parameters for SAP jobs (dynamic definition).....	285
Table 29: Supported attributes for ABAP step definition...	294
Table 30: Supported attributes for external programs and external commands step definition.....	297
Table 31: Placeholders for job interception template files.....	314
Table 32: Task string parameters for SAP jobs.....	320
Table 33: Actions performed when you rerun a process chain job	327
Table 34: Parameters to define an SAP internetwork dependency	343
Table 35: Internetwork dependency definition and possible resolution.....	344
Table 36: History table of the SAP events raised.....	349
Table 37: SAP event matching with the event rule defined.....	349
Table 38: History table of the SAP events raised.....	350
Table 39: SAP events matching with the event rule defined.....	350
Table 40: IBM Workload Scheduler fields used to define event rules based on IDocs.....	353
Table 41: IBM Workload Scheduler fields used to define correlation rules for IDoc events.....	354

Table 42: Parameters of IDOCEventGenerated event type..... 355

Table 43: Standard outbound IDoc statuses..... 357

Table 44: Standard inbound IDoc statuses.....358

Table 45: Mapping between root context MTE name and IBM Workload Scheduler fields..... 366

Table 46: Mapping between summary context MTE name and IBM Workload Scheduler fields..... 367

Table 47: Mapping between object MTE name and IBM Workload Scheduler fields..... 367

Table 48: Mapping between attribute MTE name and IBM Workload Scheduler fields..... 368

Table 49: Alert properties for correlations.....368

Table 50: SAP supported code pages.....375

Table 51: Miscellaneous troubleshooting items..... 376

Table 52: Properties for the `smseadapter.properties` file..... 389

About this publication

Scheduling End-to-end with z-centric Capabilities describes how to set up IBM Z Workload Scheduler to generate a lightweight end-to-end scheduling environment in which to schedule and control workload from the mainframe to distributed systems. While it exhaustively describes the required installation and customization tasks, this first edition provides only miscellaneous notes about the z-centric end-to-end scheduling solution. This publication is therefore to be used in conjunction with the rest of the IBM Workload Automation library; in particular see the following publications:

- *Customization and Tuning* for information about customizing and tuning IBM Z Workload Scheduler.
- *Managing the Workload* for details about organizing your workload and information about managing the workload when it becomes part of a plan and is run.
- *Messages and Codes* for message information.
- *IBM Z Workload Scheduler: Planning and Installation* for details about how to install the IBM Z Workload Scheduler Agent.
- *Memo to Users* for interoperability information among component versions.

Conventions used in this publication

Conventions used in this publication.

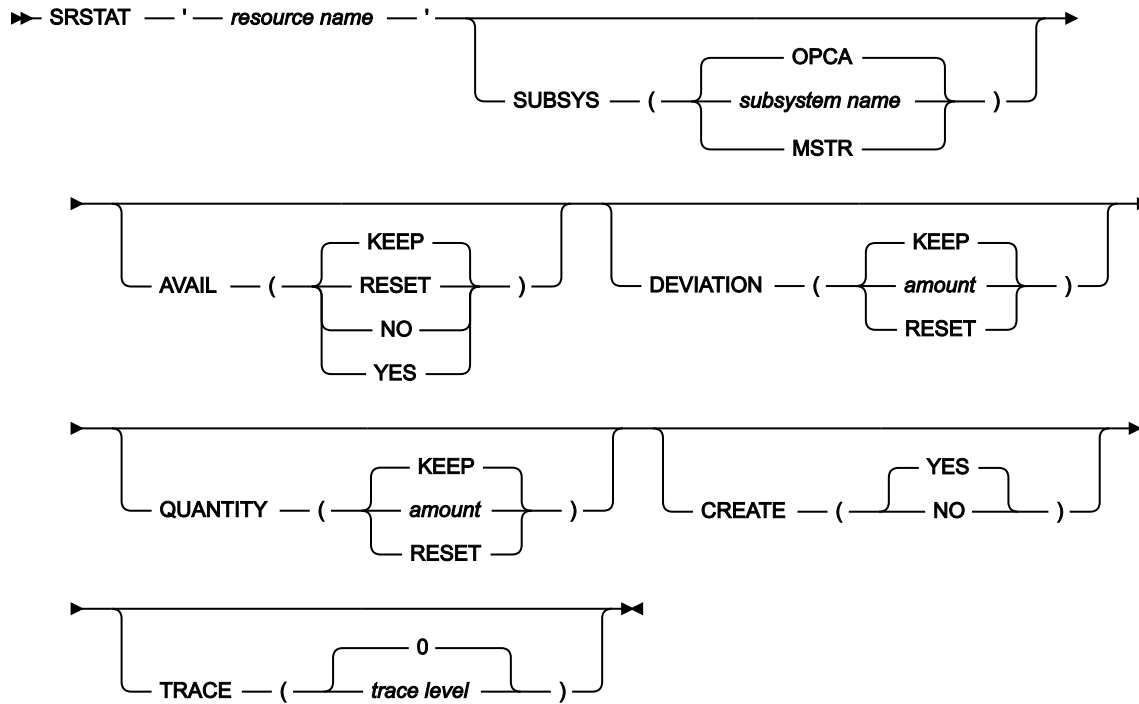
The publication uses several typeface conventions for special terms and actions. Technical changes to the text are indicated by a vertical line to the left of the change. These conventions have the following meanings:

Information type	Style convention	Example
Commands	All capital letters	CREATE
References in the text to fields on panels	All capital letters	QUANTITY
File and directory names, input you should type in panel fields	Monospace	MYAPPLICATION
First time new term introduced, publication titles	Italics	<i>Application</i>

How to read syntax diagrams

Syntax diagrams help to show syntax in a graphical way.

Throughout this publication, syntax is described in diagrams like the one shown here, which describes the SRSTAT TSO command:



The symbols have these meanings:



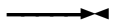
The statement begins here.



The statement is continued on the next line.



The statement is continued from a previous line.



The statement ends here.

Read the syntax diagrams from left to right and from top to bottom, following the path of the line.

These are the conventions used in the diagrams:

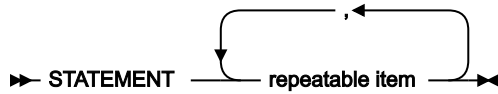
- Required items appear on the horizontal line (main path):

►► **STATEMENT** — **required item** ►►

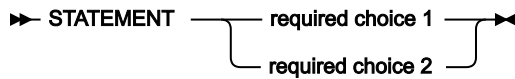
- Optional items appear below the main path:

►► **STATEMENT** — *optional item* ►►

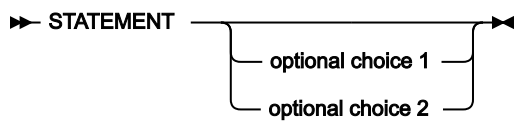
- An arrow returning to the left above the item indicates an item that you can repeat. If a separator is required between items, it is shown on the repeat arrow.



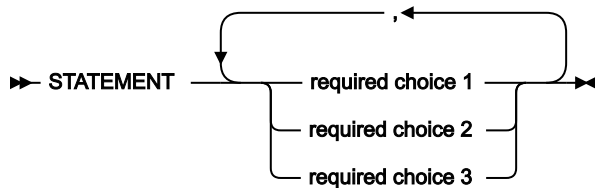
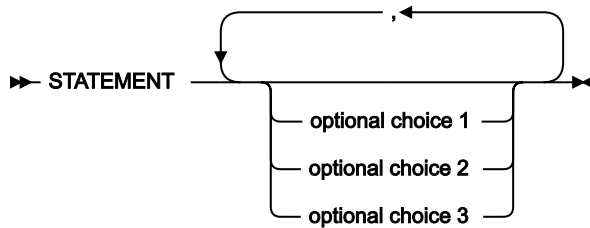
- If you can choose from two or more items, they appear vertically in a stack.
 - If you must choose one of the items, one item of the stack appears on the main path:



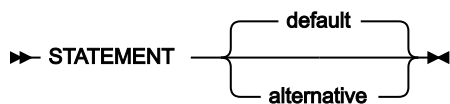
- If choosing one of the items is optional, the entire stack appears below the main path:



- A repeat arrow above a stack indicates that you can make more than one choice from the stacked items:



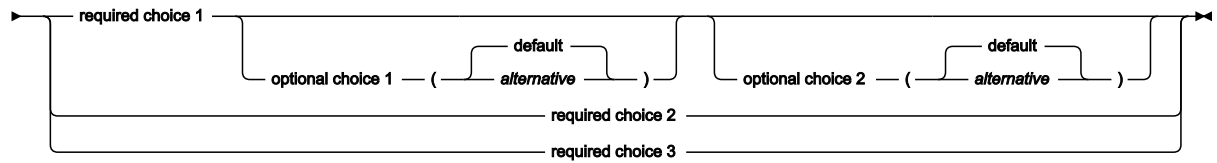
- Parameters that are above the main line are default parameters:



- Keywords appear in uppercase (for example, STATEMENT).
- Parentheses and commas must be entered as part of the command syntax, as shown.

- For complex commands, the item attributes might not fit on one horizontal line. If that line cannot be split, the attributes appear at the bottom of the syntax diagram:

►► STATEMENT ►



Chapter 1. Configuring

This chapter provides a general description of the end-to-end with z-centric capabilities environment and describes the configuration steps required to make it work.

Overview

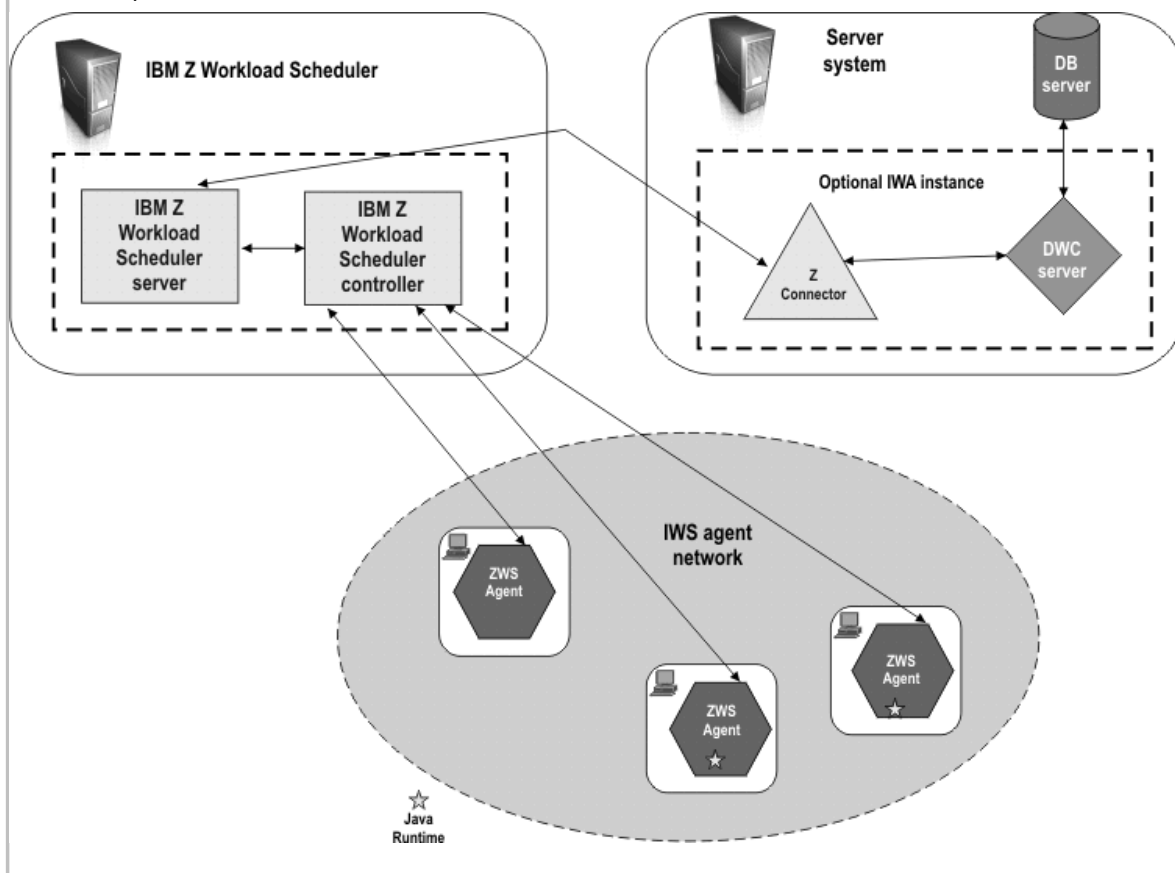
The different types of workload scheduling you can implement in your z-centric end-to-end environment:

Static scheduling

The IBM Z Workload Scheduler Agents are installed on the distributed systems and are connected to the z/OS® system through the IBM Z Workload Scheduler controller.

Static scheduling including job types with advanced options

Figure 1. Static end-to-end environment with Java™ run time installed in which you can run job types with advanced options



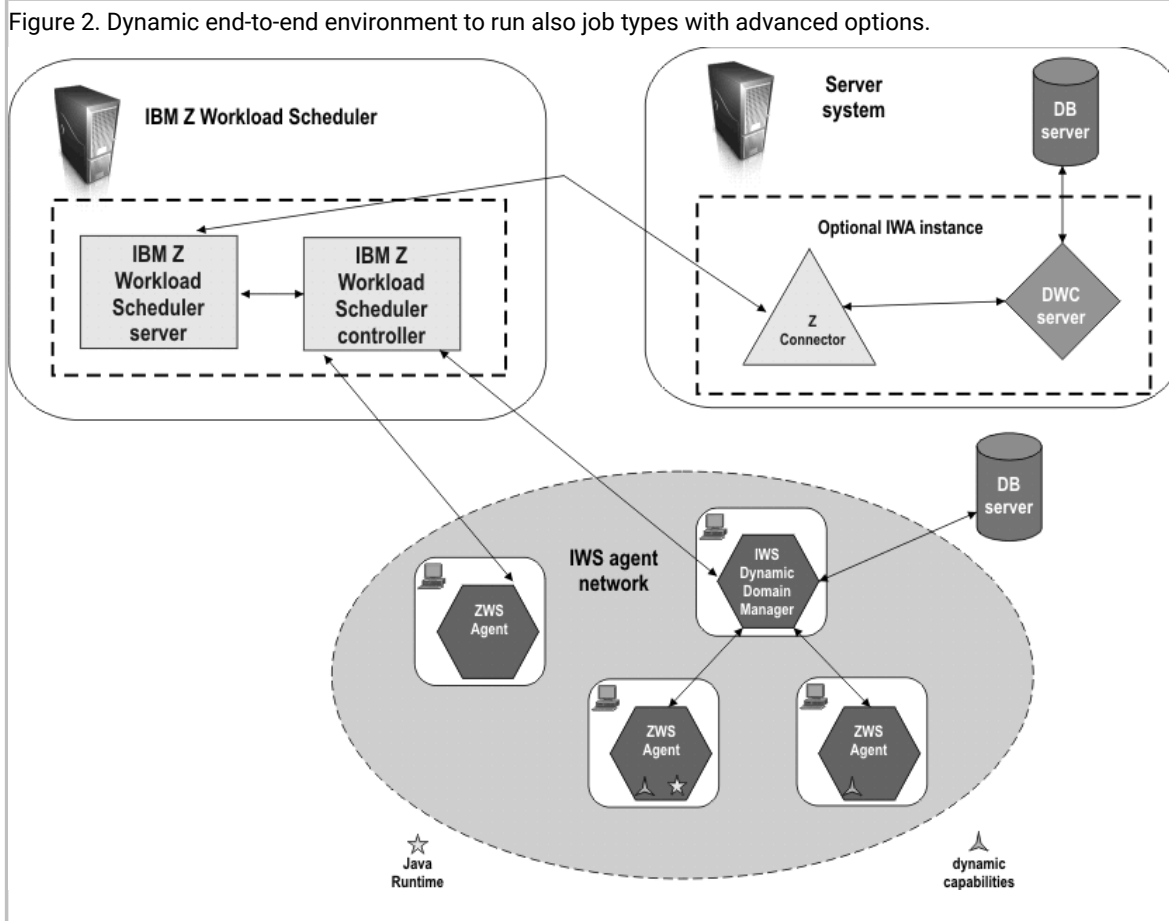
The IBM Z Workload Scheduler Agents are installed on the distributed systems with the Java™ run time, and are connected to the z/OS® system through the IBM Z Workload Scheduler controller. [Figure 1: Static end-to-end environment with Java run time installed in which you can run job types with advanced options on page 13](#) shows a **static** end-to-end environment in which IBM Z Workload Scheduler Agents:

- Without the Java™ run time installed, can run existing job types
- With the Java™ run time installed, can run both existing jobs and job types with advanced options

Dynamic scheduling

The IBM Z Workload Scheduler Agents are installed on the distributed systems with dynamic capabilities, and are connected to the dynamic domain manager. The dynamic domain manager is connected to the z/OS® system through the IBM Z Workload Scheduler controller.

Dynamic scheduling including job types with advanced options



The IBM Z Workload Scheduler Agents are installed on the distributed systems with dynamic capabilities and the Java™ run time, and are connected to the dynamic domain manager. The dynamic domain manager is connected to the z/OS® system through the IBM Z Workload Scheduler controller. [Figure 2: Dynamic end-to-end environment to run also job types with advanced options. on page 14](#) shows a **dynamic** end-to-end environment in which IBM Z Workload Scheduler Agents:

- Without the Java™ run time installed, can run existing job types
- With the Java™ run time installed, can run both existing jobs and job types with advanced options

By using the z-centric end-to-end scheduling environment, you schedule and control both static and dynamic workload from the mainframe to distributed systems with a low cost of ownership. IBM Z Workload Scheduler acts as a single point of control, providing you with all the mainframe capabilities to manage distributed workload.

IBM Z Workload Scheduler Agents can be installed on the UNIX™, IBM® i, or Windows™ workstations where you run jobs scheduled from IBM Z Workload Scheduler. After installation, you associate each agent to an IBM Z Workload Scheduler workstation definition, so that it can be used as a target for IBM Z Workload Scheduler jobs. The communication between the IBM Z Workload Scheduler Agent and IBM Z Workload Scheduler controller is direct, through the HTTP or HTTPS protocol. If the connection is interrupted, the HTTP client retries to establish it until it is successful.

Figure 3: Process flow on page 15 shows the process flow to submit and track jobs on IBM Z Workload Scheduler Agent workstations.

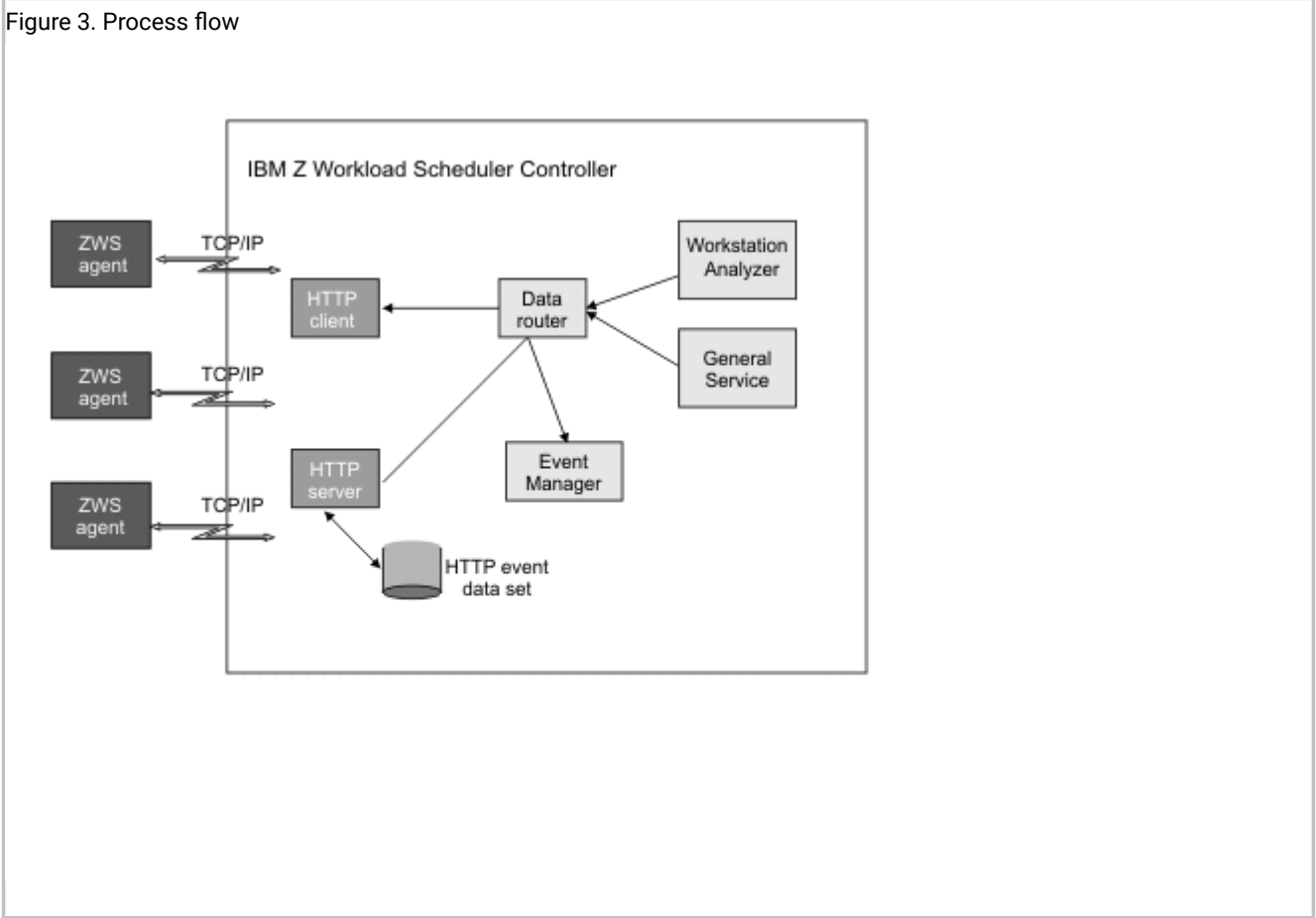


Table 1: Installation and configuration steps for setting up an end-to-end with z-centric capabilities environment on page 16 shows the steps required to set up your end-to-end with z-centric capabilities environment.

Table 1. Installation and configuration steps for setting up an end-to-end with z-centric capabilities environment

To...	Refer to...
Plan your end-to-end environment with z-centric capabilities and install the IBM Z Workload Scheduler Agent.	<i>IBM Z Workload Scheduler: Planning and Installation</i>
Install the dynamic domain manager and agent.	<i>IBM Workload Scheduler: Planning and Installation</i>
Configure the IBM Z Workload Scheduler Agent properties.	Configuring the IBM Z Workload Scheduler Agent settings on page 16
Define the IBM Z Workload Scheduler Agent and dynamic domain manager destinations.	Defining destinations with the ROUTOPTS statement on page 39
Define the IBM Z Workload Scheduler Agent workstations.	Defining and managing IBM Z Workload Scheduler Agent workstations in the database on page 55
Define dynamic workstations.	Defining and managing dynamic workstations on page 63
Configure the SSL connection protocol, if you did not change the default configuration of the IBM Z Workload Scheduler Agent.	Setting the security features on page 68
Customize the connection parameters.	Customizing the HTTP connection with the HTTPOPTS statement on page 42

Configuring the IBM Z Workload Scheduler Agent settings

The configuration settings of the IBM Z Workload Scheduler Agent are contained in the `JobManager.ini` file.

The file is made up of many different sections. Each section name is enclosed between square brackets and each section includes a sequence of `variable = value` statements.

You can customize properties for the following:

- Log and trace
- Native job executor
- Java™ job executor

Not all the properties in the `JobManager.ini` file can be customized. For a list of the configurable properties, see the tables in the following sections.

Advantages of job types with advanced options

While the existing IBM Z Workload Scheduler job is a generic script or command, you can define and schedule job types with advanced options to perform specific tasks, such as database, file transfer, Java™, and web services operations. You can easily define these job types without having specific skills on the applications where the job runs.

The job types with advanced options include both those supplied with the product and the additional types implemented through the custom plug-ins.

Besides, you can deploy or update a plug-in from Automation Hub at the following link: [Automation Hub](#).

The following job types with advanced options are available:

File transfer jobs

Transfer files to and from a server reachable using FTP, SSH, or other protocols.

Web services jobs

Call a web service.

Database jobs

Perform queries, SQL statements, and jobs on a number of databases, including custom databases. You can also create and run stored procedures on DB2, Oracle, and MSSQL databases.

Executable jobs

Run a script or command with advanced options, such as redirecting standard input and standard output to a file.

Java jobs

Run a Java class

MSSQL jobs

Run a Microsoft SQL job.

Access Method jobs

Extend job scheduling functions of IBM Workload Scheduler to other systems and applications using access methods. The access methods communicate with the external system to launch the job and return the status of the job. The following access methods are available:

- Oracle E-Business Suite
- PeopleSoft
- SAP
- MVS
- Custom methods

J2EE jobs

Allow Java applications in the same network to send and receive messages from and to a JMS destination.

IBM i jobs

Run a command on IBM i systems.

Provisioning

Jobs that span physical computers, virtual machines, and private and public cloud environments creating an on-demand environment. This job type integrates with IBM SmartCloud Provisioning.

Remote command

Jobs that can run on remote computers that do not require an agent installation.

You define job types with advanced options by connecting to a z/OS® engine with the Dynamic Workload Console. From the Dynamic Workload Console, you open the Workload Designer and select the job type you want to create. When the job definition is saved, it is stored in the JCL library and is available for scheduling from IBM Z Workload Scheduler

The job types with advanced options run on IBM Z Workload Scheduler Agents in both the static configuration (IBM Z Workload Scheduler Agent connected directly to the Z controller) and in the dynamic configuration (IBM Z Workload Scheduler Agent connected to the dynamic domain manager).

In addition to configuring job types with advanced options using the Dynamic Workload Console, you can use the related configuration files. For more information, see [Configuring to schedule job types with advanced options on page 159](#).

For the procedure about defining job types with advanced options, see the section about creating job types with advanced options in *Dynamic Workload Console User's Guide*.

For details about each job type, see the Dynamic Workload Console online help. For details about how to create jobs using the JOBRECE statement, see [JOBREC on page 75](#).

You can create custom plug-ins to implement your own job types with advanced options for applications not supported by IBM Workload Scheduler. For information about how to create custom plug-ins, see *Extending IBM Workload Automation, SC14-7623*.

The agents with dynamic capabilities can run the jobs you created for the existing IBM Workload Scheduler workstation types. To run these jobs on the dynamic agents, you only have to change the specification of the workstation where you want the job to run. The major advantage is that you can use the workflows you previously created without additional effort.

Configuring the agent

The configuration settings of the agent are stored in the `JobManager.ini` file. Starting with version 10.2.4, if you are integrating with a password vault tool, the related configuration is no longer saved in the `JobManager.ini` file. Instead, it is now stored in a profile and, optionally, in a configuration file. For more information, see [Configuring the agent to work with a password vault](#).

In a distributed environment, if a gateway is configured to allow the master domain manager or dynamic domain manager to communicate with a dynamic agent located behind a network boundary, then the gateway configuration settings of the agent are contained in the `JobManagerGW.ini` file. This file is almost identical to the `JobManager.ini` file, however, only parameters

in the [ITA], [Env], and [ResourceAdvisorAgent] sections require configuration. For these parameters, definitions are given for both the `JobManager.ini` and `JobManagerGW.ini` files.

To find out where these files are located, see the section about installation paths in *IBM Workload Scheduler: Planning and Installation*.

Only a subset of the available parameters is documented, because some parameters are reserved for internal use.

These files are made up of many different sections. Each section name is enclosed between square brackets and each section includes a sequence of `variable = value` statements.

You can customize properties for the following:

- Event-driven workload automation properties
- Log properties
- Trace properties when the agent is stopped. You can also customize traces when the agent is running using the procedure described in [Configuring trace properties when the agent is running on page 25](#).
- Native job executor
- Java™ job executor
- Resource advisor agent
- System scanner

The log messages are written in the following file:

On Windows operating systems:

`<TWA_home>\TWS\stdlist\JM\JobManager_message.log`

On UNIX and Linux operating systems:

`<TWA_DATA_DIR>/stdlist/JM/JobManager_message.log`

The trace messages are written in the following file:

On Windows operating systems:

- `<TWA_home>\TWS\stdlist\JM\ITA_trace.log`
- `<TWA_home>\TWS\stdlist\JM\JobManager_trace.log`
- `<TWA_home>\TWS\JavaExt\logs\javaExecutor0.log`

On UNIX and Linux operating systems:

- `<TWA_DATA_DIR>/stdlist/JM/ITA_trace.log`
- `<TWA_DATA_DIR>/stdlist/JM/JobManager_trace.log`
- `<TWA_DATA_DIR>/JavaExt/logs/javaExecutor0.log`

Logging information about job types with advanced options

You can use the `logging.properties` file to configure the logging process for job types with advanced options, with the exception of the Executable and Access Method job types.

The `logging.properties` file is located on the IBM Z Workload Scheduler Agent, located in the following path:

On Windows operating systems:

`<TWA_home>/TWS/JavaExt/cfg/logging.properties`

On UNIX and Linux operating systems:

`<TWA_DATA_DIR>/JavaExt/cfg/logging.properties`

After installation, this file is as follows:

```
# Specify the handlers to create in the root logger
# (all loggers are children of the root logger)
# The following creates two handlers
handlers = java.util.logging.ConsoleHandler,
           java.util.logging.FileHandler

# Set the default logging level for the root logger
.level = INFO

# Set the default logging level for new ConsoleHandler instances
java.util.logging.ConsoleHandler.level = INFO

# Set the default logging level for new FileHandler instances
java.util.logging.FileHandler.level
    = ALL
java.util.logging.FileHandler.pattern
    = C:\TWA_home\TWS\JavaExt\logs\javaExecutor%g.log
java.util.logging.FileHandler.limit
    = 1000000
java.util.logging.FileHandler.count
    = 10

# Set the default formatter for new ConsoleHandler instances
java.util.logging.ConsoleHandler.formatter =
    java.util.logging.SimpleFormatter
java.util.logging.FileHandler.formatter =
    java.util.logging.SimpleFormatter

# Set the default logging level for the logger named com.mycompany
com.ibm.scheduling = INFO
```

You can customize:

- The logging level (from INFO to WARNING, ERROR, or ALL) in the following keywords:

`.level`

Defines the logging level for the internal logger.

com.ibm.scheduling

Defines the logging level for the job types with advanced options. To log information about job types with advanced options, set this keyword to ALL.

- The path where the logs are written, specified by the following keyword:

```
java.util.logging.FileHandler.pattern
```

Not all the properties in the `JobManager.ini` and `JobManagerGW.ini` files can be customized. For a list of the configurable properties, see the following sections:

- [Configuring log message properties \[JobManager.Logging.cclog\] on page 22](#)
- [Configuring trace properties when the agent is stopped \[JobManager.Logging.cclog\] on page 23.](#)
- [Configuring common launchers properties \[Launchers\] on page 28](#)
- [Configuring properties of the native job launcher \[NativeJobLauncher\] on page 30.](#)
- [Configuring properties of the Java job launcher \[JavaJobLauncher\] on page 33.](#)
- [Configuring properties of the Resource advisor agent \[ResourceAdvisorAgent\] on page 34.](#)
- [Configuring properties of the System scanner \[SystemScanner\] on page 36](#)
- [Configuring environment variables \[Env\]](#)
- The section about configuring properties of event-driven workload automation [EventDrivenWorkload] in *IBM Z Workload Scheduler: Scheduling End-to-end with z-centric Capabilities*

Configuring proxy properties [ITA_Env]

About this task

To have the IBM Z Workload Scheduler Agent communicate with the Z controller through a proxy, edit the `[ITA_Env]` section in the `JobManager.ini` file as follows.



Note: On the controller, ensure that you have set the PROXY keyword in the ROUTOPTS statement.

Depending on the type of connection you are using, add or modify the following property:

SSL-secured connection

https_proxy

The URL of the proxy through which the agent communicates to the controller. The value is

```
https_proxy =http://<proxy_wks>:<proxy_wks_port>
```

where:

- `<proxy_wks>` is the fully qualified host name of the workstation where the proxy is configured.
- `<proxy_wks_port>` is the port number of the workstation where the proxy is configured.

Non SSL-secured connection**http_proxy**

The URL of the proxy through which the agent communicates to the controller. The value is

```
http_proxy =http://<proxy_wks>:<proxy_wks_port>
```

where:

- <proxy_wks> is the fully qualified host name of the workstation where the proxy is configured.
- <proxy_wks_port> is the port number of the workstation where the proxy is configured.

Restart the agent after the property change.

Configuring log message properties [JobManager.Logging.cclog]**About this task**

To configure the logs, edit the [JobManager.Logging.cclog] section in the `JobManager.ini` file. This procedure requires that you stop and restart the IBM Workload Scheduler agent

The section containing the log properties is named:

```
[JobManager.Logging.cclog]
```

You can change the following properties:

JobManager.loggerhd.fileName

The name of the file where messages are to be logged. the default value is

On Windows operating systems

```
TWA_home\stdlist\JM\JobManager_message.log
```

On UNIX operating systems

```
$(TWA_DATA_DIR)/stdlist/JM/JobManager_message.log
```

JobManager.loggerhd.maxFileBytes

The maximum size that the log file can reach. The default is **1024000** bytes.

JobManager.loggerhd.maxFiles

The maximum number of log files that can be stored. The default is **3**.

JobManager.loggerhd.fileEncoding

By default, log files for the agent are coded in UTF-8 format. If you want to produce the log in a different format, add this property and specify the required codepage.

JobManager.loggerfl.level

The amount of information to be provided in the logs. The value ranges from 3000 to 7000. Smaller numbers correspond to more detailed logs. The default is **3000**.

JobManager.ffdc.maxDiskSpace

Exceeding this maximum disk space, log files collected by the first failure data capture mechanism are removed, beginning with the oldest files first.

JobManager.ffdc.baseDir

The directory to which log and trace files collected by the ffdc tool are copied. The default directory is

On Windows operating systems

<TWA_home>\stdlist\JM\JOBMANAGER-FFDC

On UNIX operating systems

\$(TWA_DATA_DIR)/stdlist/JM/JOBMANAGER-FFDC

JobManager.ffdc.filesToCopy

Log and trace files (JobManager_message.log and JobManager_trace.log) collected by the ffdc tool located in <TWA_home>\TWS\stdlist\JM. The files are available in the following paths:

On Windows operating systems

- TWA_home\TWS\stdlist\JM\JobManager_message.log
- TWA_home\TWS\stdlist\JM\JobManager_trace.log

On UNIX operating systems

- \$(TWA_DATA_DIR)/stdlist/JM/JobManager_message.log
- \$(TWA_DATA_DIR)/stdlist/JM/JobManager_trace.log

When a message is logged (JobManager.ffdc.triggerFilter = JobManager.msgIdFilter) that has an ID that matches the pattern "AWSITA*E" (JobManager.msgIdFilter.msgIds = AWSITA*E), which corresponds to all error messages, then the log and trace files (JobManager.ffdc.filesToCopy = "/opt/IBM/TWA_<TWS_user>/TWS/stdlist/JM/JobManager_message.log" "/opt/IBM/TWA_<TWS_user>/TWS/stdlist/JM/JobManager_trace.log") are copied (JobManager.ffdc.className = ccg_ffdc_filecopy_handler) to the directory JOBMANAGER-FFDC (JobManager.ffdc.baseDir = /opt/IBM/TWA_<TWS_user>/TWS/stdlist/JM/JOBMANAGER-FFDC). If the files copied exceed 10 MB (JobManager.ffdc.maxDiskSpace = 10000000), then the oldest files are removed first (JobManager.ffdc.quotaPolicy = QUOTA_AUTODELETE).

Configuring trace properties when the agent is stopped [JobManager.Logging.cclog]

How to configure the trace properties when the agent is stopped.

To configure the trace properties when the agent is stopped, edit the [JobManager.Logging] section in the `JobManager.ini` file and then restart the IBM Workload Scheduler agent.

The section containing the trace properties is named:

```
[JobManager.Logging.cclog]
```

You can change the following properties:

JobManager.trhd.fileName

The name of the trace file. the default path is as follows:

On Windows operating systems

`TWA_home/TWS/stdlist/JM/JobManager_trace.log`

On UNIX operating systems

`$(TWA_DATA_DIR)/stdlist/JM/JobManager_trace.log`

JobManager.trhd.maxFileBytes

The maximum size that the trace file can reach. The default is 10240000 bytes.

JobManager.trhd.maxFiles

The maximum number of trace files that can be stored. The default is 5.

JobManager.trfl.level

Determines the type of trace messages that are logged. Change this value to trace more or fewer events, as appropriate, or on request from IBM Software Support. Valid values are:

DEBUG_MAX

Maximum tracing. Every trace message in the code is written to the trace logs.

INFO

All *informational*, *warning*, *error* and *critical* trace messages are written to the trace. The default value.

WARNING

All *warning*, *error* and *critical* trace messages are written to the trace.

ERROR

All *error* and *critical* trace messages are written to the trace.

CRITICAL

Only messages which cause the agent to stop are written to the trace.

The output trace (`JobManager_trace.log`) is provided in XML format.

After installing the z-centric agent or dynamic agent on Windows 2012, the `JobManager_trace.log` might not be created. In this case, perform the following procedure:

1. Stop the agent.
2. Create a backup copy of `JobManager.ini`, and edit the original file by changing the row:

```
JobManager.trhd.className = ccg_multiproc_filehandler
```

to

```
JobManager.trhd.className = ccg_filehandler
```

3. Restart the agent.

Configuring trace properties when the agent is running

Use the **twstrace** command to set the trace on the agent when it is running.

Using the **twstrace** command, you can perform the following actions on the agent when it is running:

- [See command usage and verify version on page 25.](#)
- [Enable or disable trace on page 26.](#)
- Set the traces to a specific level, specify the number of trace files you want to create, and the maximum size of each trace file. See [Set trace information on page 26.](#)
- [Show trace information on page 26.](#)
- Collect trace files, message files, and configuration files in a compressed file using the command line. See [Collect trace information on page 27.](#)
- Collect trace files, message files, and configuration files in a compressed file using the Dynamic Workload Console. See the section about retrieving agent traces from the Dynamic Workload Console in *Troubleshooting Guide*.

You can also configure the traces when the agent is not running by editing the [JobManager.Logging] section in the `JobManager.ini` file as described in [Configuring the agent on page 18](#). This procedure requires that you stop and restart the agent.

twstrace command

Use the **twstrace** command to configure traces, and collect logs, traces, and configuration files (`ita.ini` and `jobManager.ini`) for agents. You collect all the information in a compressed file when it is running without stopping and restarting it.

See command usage and verify version

To see the command usage and options, use the following syntax.

Syntax

```
twstrace -u | -v
```

Parameters

-u

Shows the command usage.

-v

Shows the command version.

Enable or disable trace

To set the trace to the maximum or minimum level, use the following syntax.

Syntax

twstrace -enable | -disable

Parameters

-enable

Sets the trace to the maximum level. The maximum level is **1000**.

-disable

Sets the trace to the minimum level. The minimum level is **3000**.

Set trace information

To set the trace to a specific level, specify the number of trace files you want to create, and the maximum size the trace files can reach, use the following syntax.

Syntax

twstrace [-level <level_number>] [-maxFiles <files_number>] [-maxFileBytes <bytes_number>]

Parameters

-level <level_number>

Sets the trace level. Specify a value in the range from 1000 to 3000, which is also the default value. Note that if you set this parameter to 3000, you have the lowest verbosity level and the fewest trace messages. To have a better trace level, with the most verbose trace messages and the maximum trace level, set it to **1000**.

-maxFiles <files_number>

Specify the number of trace files you want to create.

-maxFileBytes <bytes_number>

Set the maximum size in bytes that the trace files can reach. The default is **1024000** bytes.

Show trace information

To display the current trace level, the number of trace files, and the maximum size the trace files can reach, use the following syntax.

Syntax

twstrace -level | -maxFiles | -maxFileBytes

Parameters

-level

See the trace level you set.

-maxFiles

See the number of trace files you create.

-maxFileBytes

See the maximum size you set for each trace file

Example

Sample

The example shows the information you receive when you run the following command:

```
twstrace -level -maxFiles -maxFileBytes
AWSITA176I The trace properties are: level="1000",
max files="3", file size="1024000".
```

Collect trace information

To collect the trace files, the message files, and the configuration files in a compressed file, use the following syntax.

Syntax

```
twstrace -getLogs [ -zipFile <compressed_file_name> ] [ -host <host_name> ] [ -protocol {http | https} [ -port <port_number> ] [ -iniFile <ini_file_name> ]
```

Parameters

-zipFile <compressed_file_name>

Specify the name of the compressed file that contains all the information, that is logs, traces, and configuration files (ita.ini and jobManager.ini) for the agent. The default is **logs.zip**.

-host <host_name>

Specify the host name or the IP address of the agent for which you want to collect the trace. The default is **localhost**.

-protocol http|https

Specify the protocol of the agent for which you are collecting the trace. The default is the protocol specified in the .ini file of the agent.

-port <port_number>

Specify the port of the agent. The default is the port number of the agent where you are running the command line.

-iniFile <ini_file_name>

Specify the name of the **.ini** file that contains the SSL configuration of the agent for which you want to collect the traces. If you are collecting the traces for a remote agent for which you customized the security certificates, you must import the certificate on the local agent and specify the name of the **.ini** file that contains this configuration. To do this, perform the following actions:

1. Extract the certificate from the keystore of the remote agent.
2. Import the certificate in a local agent keystore. You can create an ad hoc keystore whose name must be **TWSClientKeyStore.kdb**.
3. Create an **.ini** file in which you specify:

- **0** in the **tcp_port** property as follows:

```
tcp_port=0
```

- The port of the remote agent in the **ssl_port** property as follows:

```
ssl_port=<ssl_port>
```

- The path to the keystore you created in Step 2 on page 28 in the **key_repository_path** property as follows:

```
key_repository_path=<local_agent_keystore_path>
```

Configuring common launchers properties [Launchers]

About this task

In the `JobManager.ini` file, the section containing the properties common to the different launchers (or executors) is named:

```
[Launchers]
```

The following properties are available:

BaseDir

The installation path of the IBM Workload Scheduler agent. Do not modify this value.

CommandHandlerMinThreads

Indicates the maximum number of commands that can be run on the agent concurrently. Limits to the number of jobs vary depending on the resources of your workstation, however consider that operations on comdhandler are usually short. The default is **20**. Usually, there is no need to modify this setting, even if you plan a very high workload on the agent. You might want to change it if many commands are run concurrently on the agent, for example, many concurrent requests to retrieve job logs.

CommandHandlerMaxThreads

Indicates the maximum number of commands that can be run on the agent concurrently. Limits to the number of jobs vary depending on the resources of your workstation, however consider that operations on comdhandler are usually short. The default is **100**. Usually, there is no need to modify this setting, even if you plan a very high workload on the agent. You might want to change it if many commands are run concurrently on the agent, for example, many concurrent requests to retrieve job logs.

CpaHeartBeatTimeSeconds

The polling interval in seconds used to verify if the **agent** process is still up and running. If the agent process is inactive the product stops also the **JobManager** process. The default is **30**. Modify only if you use dynamic pools with CPU-based requirements or optimization policies. With a lower value, the agent reacts quickly to CPU modifications, but this might cause unstable values in case of CPU spikes. Lower values causes a higher use of resources on the agent.

DirectoryPermissions

The access rights assigned to the agent for creating directories when running jobs. The default is **0755**. Supported values are UNIX-format entries in hexadecimal notation.

DownloadDir

The name of the directory where the fix pack installation package or upgrade elmage for IBM Workload Scheduler fault-tolerant agents or dynamic agents is downloaded during the centralized agent update process. If not specified, the following default directory is used:

On Windows operating systems:

`TWA_home\TWS\stdlist\JM\download`

On UNIX operating systems:

`TWA_DATA_DIR/TWS/stdlist/JM/download`

The centralized agent update process does not apply to z-centric agents.

ExecutorsMaxThreads

Specifies the maximum number of jobs the dynamic agent can run concurrently. For example, to allow the dynamic agent to run a maximum of 500 jobs concurrently, set this parameter to **500**. The default is **400**.

ExecutorsMinThreads

Specifies the minimum number of jobs the dynamic agent can run concurrently. For example, to allow the dynamic agent to run a minimum of 500 jobs concurrently, set this parameter to **500**. The default is **38**. Modify if the number of expected concurrent jobs is much higher than 38. The agent dynamically allocates more threads if necessary, until it reaches the value specified in **ExecutorsMaxThreads**.

FilePermissions

The access rights assigned to the agent for creating files when running jobs. The default is **0755**. Supported values are UNIX-format entries in hexadecimal notation.

MaxAge

The number of days that job logs are kept (in path `TWA_home/TWS/stdlist/JM`) before being deleted. The default is **30**. Possible values range from a minimum of 1 day.

MaxMetricsSizeKBytes

The maximum combined size in kilobytes of both metrics and output properties. This property is not present by default in the `JobManager.ini` file, but can be manually added to help optimize CPU and disk space usage. If you encounter a message similar to the one below, configuring this property might be beneficial:

```
AWSITA447E Failed to add the value for the specified metric or property:
storage capacity for metrics and properties is insufficient.
```

If this property is not present in the `JobManager.ini` file, the default value is 16,384 KB.

NotifierMaxThreads

Notifier threads are in charge of notifying the dynamic workload broker of each status change in a job. This parameter specifies the maximum number of job status changes that can be notified to the dynamic workload broker.

NotifierMinThreads

Notifier threads are in charge of notifying the dynamic workload broker of each status change in a job. This parameter specifies the minimum number of job status changes that can be notified to the dynamic workload broker. The default value is **3**. Modify this parameters only in case of unexpected errors and after consulting with software support team.

SpoolDir

The path to the folder containing the jobstore and outputs. The default is:

On Windows operating systems

`TWA_home/TWS/stdlist/JM`

On UNIX operating systems

`$(TWA_DATA_DIR/stdlist/JM`

StackSizeBytes

The size of the operating system stack in bytes. The default is **DEFAULT**, meaning that the **agent** uses the default value for the operating system. Do not modify this parameter unless instructed to do so by the software support team. Incorrect values can cause the agent to crash.

Configuring properties of the native job launcher [NativeJobLauncher]

About this task

In the `JobManager.ini` file, the section containing the properties of the native job launcher is named:

```
[NativeJobLauncher]
```

You can change the following properties:

AllowRoot

Applies to UNIX™ systems only. Specifies if the root user can run jobs on the agent. It can be `true` or `false`. The default is `false`. This property does not apply to IBM i, use the `AllowQSECOFR` option instead

AllowQSECOFR

Applies to IBM i systems only. Specifies if QSECOFR user can run jobs on the agent. It can be `true` or `false`. The default is `true`. Add a line like `AllowQSECOFR = false` to the `JobManager.ini` file to deny job execution to QSECOFR.

CheckExec

If `true`, before launching the job, the agent checks both the availability and the execution rights of the binary file. The default is `true`.

DefaultWorkingDir

Specifies the working directory of native jobs. You can also specify the value for the working directory when creating or editing the job definition in the Graphical Designer. When specified in the Graphical Designer, this value overrides the value specified for the **DefaultWorkingDir** property. If you do not specify any working directories, the `<TWS_home>\bin` directory is used.

JobUnspecifiedInteractive

Applies to Windows™ operating systems only. Specifies if native jobs are to be launched in interactive mode. It can be `true` or `false`. The default is `false`.

KeepCommandTraces

Set to `true` to store the traces of the method invocation for actions performed on a job definition, for example, when selecting from a picklist. These files are stored in the path `/opt/IBM/TWA_<TWS_user>/TWS/stdlist/JM/r3batch_cmd_exec`. The default setting is `false`.

KeepJobCommandTraces

Set to `true` to store the traces of the method invocation for actions performed on a job instance, for example, viewing a spool list. These files are stored in the .zip file of the job instance. The default setting is `true`.

LoadProfile

Applies to agents on Windows servers only. Specifies if the user profile is to be loaded. It can be `true` or `false`. The default is `true`.

MonitorQueueName

Specifies the name of the queue where the IBM i jobs are monitored. If you do not specify this property, the default queue (QBATCH) is used.

PortMax

The maximum range of the port numbers used by the task launcher to communicate with the Job Manager. The default is 0, meaning that the operating system assigns the port automatically.

PortMin

The minimum range of the port numbers used by the task launcher to communicate with the Job Manager. The default is 0, meaning that the operating system assigns the port automatically.

PostJobExecScriptPathName

The fully qualified path of the script file that you want to run when the job completes. By default, this property is not present in the `JobManager.ini` file. If you do not specify any file path or the script file doesn't exist, no action is taken.

This property applies to dynamic agent and z/OS agent. For details about running a script when a job completes, see *User's Guide and Reference*.

PromotedNice

Used in workload service assurance. This property is not supported on the Agent for z/OS.

For UNIX and Linux operating systems only, assigns the priority value to a critical job that needs to be promoted so that the operating system processes it before others. Applies to critical jobs or predecessors that need to be promoted so that they can start at their critical start time.

Boundary values vary depending upon each specific platform, but generally lower values correspond to higher priority levels and vice versa. The default is -1.

Be aware that:

- The promotion process is effective with negative values only. If you set a positive value, the system runs it with the -1 default value.
- An out of range value (for example -200), prompts the operating system to automatically promote the jobs with the lowest allowed nice value.
- Overusing the promotion mechanism (that is, defining an exceedingly high number of jobs as mission critical and setting the highest priority value here) might overload the operating system, negatively impacting the overall performance of the workstation.

PromotedPriority

Used in workload service assurance. This property is not supported on the Agent for z/OS.

For Windows operating systems only, sets to this value the priority by which the operating system processes a critical job when it is promoted. Applies to critical jobs or predecessors that need to be promoted so that they can start at their critical start time. Valid values are:

- High
- AboveNormal (the default)
- Normal
- BelowNormal
- Low OR Idle

Note that if you set a lower priority value than the one non-critical jobs might be assigned, no warning is given.

RequireUserName

When `true`, requires that you add the user name in the JSDL job definition.

When `false`, runs with the user name used by job manager, that is:

- `TWS_user` on UNIX™ and Linux™ systems
- The local system account on Windows™ systems

The default is `false`.

RunExecutablesAsIBMiJobs

If you set this property to `true`, you can define IBM i jobs as generic jobs without using the XML definition. Generic jobs are automatically converted to IBM i jobs. As a side effect, generic jobs cannot be run when this parameter is enabled (`RunExecutablesAsIBMiJobs=true`). There is no default value because this property is not listed in the `JobManager.ini` file after the agent installation.

If you set this property to `true`, ensure that the user you used to install the agent has been granted the `*ALLOBJ` special authority.

RunInteractiveJobOnInvalidSession

Applies only to native and executable jobs starting interactive programs running on dynamic agents installed on Windows operating systems. Interactive programs run only if the job user has an active session open when the job runs. If there is no active session for the job user, the job behavior is defined by this property, as follows. Set the property to `true` to enable jobs to start interactive programs even if there is no active session for the job user. Set the property to `false` to prevent jobs from starting interactive programs if there is no active session for the job user.

ScriptSuffix

The suffix to be used when creating the script files. It is:

`.cmd`

For Windows™

`.sh`

For UNIX™

VerboseTracing

Enables verbose tracing. It is set to `true` by default.

Configuring properties of the Java™ job launcher [JavaJobLauncher]

About this task

In the `JobManager.ini` file, the section containing the properties of the Java™ job launcher is named:

```
[JavaJobLauncher]
```

You can change the following property:

JVMOptions

The options to provide to the Java™ Virtual Machine used to start job types with advanced options. Supported keywords for establishing a secure connection are:

- https.proxyHost
- https.proxyPort

Supported keywords for establishing a non-secure connection are:

- Dhttp.proxyHost
- Dhttp.proxyPort

For example, to set job types with advanced options, based on the default JVM http protocol handler, to the unauthenticated proxy server called with name myproxyserver.mycompany.com, define the following option:

```
JVMOptions = -Dhttp.proxyHost=myproxyserver.mycompany.com -Dhttp.proxyPort=80
```

Configuring properties of the Resource advisor agent [ResourceAdvisorAgent]

About this task

In the `JobManager.ini` and `JobManagerGW.ini` files, the section containing the properties of the Resource advisor agent is named:

```
[ResourceAdvisorAgent]
```

You can change the following properties:

BackupResourceAdvisorUrls

The list of URLs returned by the IBM Workload Scheduler master in a distributed environment or by the dynamic domain manager either in a z/OS or in a distributed environment. The agent uses this list to connect to the master or dynamic domain manager.

CPUScannerPeriodSeconds

The time interval that the Resource advisor agent collects resource information about the local CPU. The default value is every 10 seconds.

FullyQualifiedHostname

The fully qualified host name of the agent. It is configured automatically at installation time and is used to connect with the master in a distributed environment or with the dynamic domain manager in a z/OS or in a distributed environment. Edit only if the host name is changed after installation.

NotifyToResourceAdvisorPeriodSeconds

The time interval that the Resource advisor agent forwards the collected resource information to the Resource advisor. The default value is every 119 seconds.

ResourceAdvisorUrl

JobManager.ini

The URL of the master in a distributed environment, or of the dynamic domain manager in a z/OS or in a distributed environment, that is hosting the agent.

This URL is used until the server replies with the list of its URLs. The value is

`https://$(tdwb_server):$(tdwb_port)/JobManagerRESTWeb/JobScheduler/resource`, where:

`$(tdwb_server)`

is the fully qualified host name of the master in a distributed environment or of the dynamic domain manager either in a z/OS or in a distributed environment.

`$(tdwb_port)`

is the port number of the master in a distributed environment or of the dynamic domain manager either in a z/OS or in a distributed environment.

It is configured automatically at installation time. Edit only if the host name or the port number are changed after installation, or if you do not use secure connection (set to `http`). If you set the port number to zero, the resource advisor agent does not start. The port is set to zero if at installation time you specify that you will not be using the master in a distributed environment or the dynamic domain manager either in a z/OS or in a distributed environment.

In a distributed environment, if **-gateway** is set to either `local` or `remote`, then this is the URL of the dynamic agent workstation where the gateway resides and through which the dynamic agents communicate. The value is

`https://$(tdwb_server):$(tdwb_port)/ita/JobManagerGW/JobManagerRESTWeb/JobScheduler/resource`, where:

`$(tdwb_server)`

The fully qualified host name of the dynamic agent workstation where the gateway resides and through which the dynamic agent communicates with the dynamic workload broker.

`$(tdwb_port)`

The port number of the dynamic agent workstation where the gateway resides and through which the dynamic agent communicates with the dynamic workload broker.

JobManagerGW.ini

In a distributed environment, if **-gateway** is set to `local`, then **ResourceAdvisorUrl** is the URL of the master or dynamic domain manager. The value is

`https://$(tdwb_server):$(tdwb_port)/JobManagerRESTWeb/JobScheduler/resource`, where:

`$(tdwb_server)`

The fully qualified host name of the master or dynamic domain manager.

`$(tdwb_port)`

The port number of the master or dynamic domain manager.

ScannerPeriodSeconds

The time interval that the Resource advisor agent collects information about all the resources in the local system other than CPU resources. The default value is every 120 seconds.

The resource advisor agent, intermittently scans the resources of the machine (computer system, operating system, file systems and networks) and periodically sends an update of their status to the master or dynamic domain manager either in a z/OS or in a distributed environment.

The CPU is scanned every `CPUScannerPeriodSeconds` seconds, while all the other resources are scanned every `ScannerPeriodSeconds` seconds. As soon as one of the scans shows a significant change in the status of a resource, the resources are synchronized with the master in a distributed environment or the dynamic domain manager either in a z/OS or in a distributed environment. The following is the policy followed by the agent to tell if a resource attribute has significantly changed:

- A resource is added or deleted
- A string attribute changes its value
- A CPU value changes by more than `DeltaForCPU`
- A file system value changes by more than `DeltaForDiskMB` megabytes
- A Memory value changes by more than `DeltaForMemoryMB` megabytes

If there are no significant changes, the resources are synchronized with the IBM Workload Scheduler master in a distributed environment or with the dynamic domain manager either in a z/OS or in a distributed environment every `NotifyToResourceAdvisorPeriodSeconds` seconds.

Configuring properties of the System scanner [SystemScanner]

About this task

In the `JobManager.ini` file, the section containing the properties of the System scanner is named:

```
[SystemScanner]
```

You can change the following properties:

CPU Samples

The number of samples used to calculate the average CPU usage. The default value is 3.

DeltaForCPU

The change in CPU usage considered to be significant when it becomes higher than this percentage (for example, `DeltaForCPU` is 20 if the CPU usage changes from 10 percent to 30 percent). The default value is 20 percent.

DeltaForDiskMB

The change in use of all file system resources that is considered significant when it becomes higher than this value. The default value is 100 MB.

DeltaForMemoryMB

The change in use of all system memory that is considered significant when it becomes higher than this value.
The default value is 100 MB.

Configuring properties of event-driven workload automation [EventDrivenWorkload]**About this task**

In the `JobManager.ini` file, the section containing the properties of event-driven workload automation is named:

```
[EventDrivenWorkload]
```



Important: Do not modify the values of these properties, however, you can consult them for troubleshooting purposes.

The following properties are contained in this section:

EventProcessorHostname

The host name where the event processor is located.

ThisWorkstation

The name of the agent workstation where the `JobManager.ini` file, containing these properties, is located.

EventProcessorElfSslPort

The Event Integration Facility (EIF) port number where the event processing server receives events when it is SSL-protected.

EventProcessorElfPort

The EIF port number where the event processing server receives events.

EdwaDesiredStatusRunning

To enable event-driven workload automation set to `true`, otherwise, `false`. The default value is `true`.

CRC

The current cyclic redundancy check (CRC) of the event rules of this workstation as reported by the event processor.

DeployedCRC

The CRC of the event rules already deployed.

EventProcessorWorkstation

The name of the workstation on which the event processor is located.

EventProcessorPort

The port number used to contact the event processor.

EventProcessorContext

The context part of the unified resource identifier (URI) of the event processor.

EventProcessorProtocol

The protocol used to contact the event processor. Valid values: `http`, `https`. The default value is `https`.

Regular maintenance

Regular maintenance refers to the mechanisms that are used on your dynamic workstation agents to free up storage space and improve performance.

Unlike IBM Workload Scheduler fault-tolerant agents, where maintenance tasks must be performed manually using the `rmstdlist` utility command, you can have regular maintenance performed on your dynamic agent workstations to keep disk space under control by configuring the following parameters as appropriate.

Table 2. Agent configuration parameters

File	Parameter	Description
JobManager.ini located in the path	MaxAge	The number of days that job logs are kept before being deleted. The default is 2. Possible values range from a minimum of 1 day.
On UNIX™ operating systems		
<code>TWA_DATA_DIR/ITA/cpa/config/JobManager.ini</code>	JobManager.log gerhd.maxFileBytes	The maximum size that the log file can reach. The default is 1024000 bytes.
On Windows™ operating systems		
<code>TWA_home\TWS\ITA\cpa\config\JobManager.ini</code>	JobManager.log gerhd.maxFiles	The maximum number of log files that can be stored in the <code>stdlist/JM</code> directory. The default is 3.
	JobManager.ffdc .maxDiskSpace	The maximum disk space reached, by the log files collected by the First Failure Data Capture tool, after which the oldest files are removed.
	JobManager.trhd .maxFileBytes	The maximum size that the log file can reach. The default is 10240000 bytes.
	JobManager.trhd .maxFiles	The maximum number of log files that can be stored. The default is 5.
logging.properties located in the path	java.util.logging. FileHandler.limit	The maximum amount to write log messages to a file. Default value is 1000000 (bytes)
On UNIX™ operating systems		
<code>TWA_DATA_DIR/TWS/JavaExt/cfg/</code>	java.util.logging. FileHandler.count	The number of output files to cycle through. Default value is 10.

Table 2. Agent configuration parameters

(continued)

File	Parameter	Description
On Windows™ operating systems		
<code>TWA_home\TWS\Ja</code>		
<code>vaExt\cfg</code>		
		Logs related to jobs with advanced options.

Configuring the Z controller

This section describes the configuration steps required on the Z controller.

Defining destinations with the ROUTOPTS statement

IBM Z Workload Scheduler Agents and dynamic domain managers are connected to the Z controller. This means that you use the ROUTOPTS statement to define the connection details of an IBM Z Workload Scheduler Agent and dynamic domain manager to the controller or standby controller.

Set the HTTP or HTTPS keyword to define all your IBM Z Workload Scheduler Agent destinations. For IBM Z Workload Scheduler Agents and dynamic domain managers, a destination is the combination of a name and the fully qualified host name, or IP address, and port number of the IBM Z Workload Scheduler Agent or dynamic domain manager.

HTTP and HTTPS have the same syntax. The difference is that HTTPS defines the connection as SSL-secure. ROUTOPTS is defined in the member of the EQPPARM library specified by the PARM parameter in the JCL EXEC statement of the controller.

The syntax for defining IBM Z Workload Scheduler Agent and dynamic domain managers destinations is the following:

```
ROUTOPTS HTTP|HTTPS(destination,...,destination)
```

where *destination* is indicated as follows:

For IBM Z Workload Scheduler Agents

```
dest name: 'hostname or IPaddr of agent' /port of agent[/type] [/pulseivl] [/proxyname]
```

For more information, see [Figure 1: Static end-to-end environment with Java run time installed in which you can run job types with advanced options on page 13](#).

For dynamic domain managers

```
hostname or IPaddress of DDM/port of DDM/B[/pulseivl]
```

For more information, see [Figure 2: Dynamic end-to-end environment to run also job types with advanced options. on page 14](#).

For master domain managers

```
hostname or IPaddress of MDM/port of MDM/B
```

The destination name is any alphanumeric sequence of up to 8 characters. You can include as many destinations as you want within the parentheses. They must be separated by commas. Because you cannot specify more than 455 lines for each ROUTOPTS statement, you can add further destinations by specifying more than one ROUTOPTS statement.

The `pulseivl` value is optional and defines the frequency, expressed in minutes, with which the IBM Z Workload Scheduler controller checks that the agent or dynamic domain manager is running. Values range from 0 to 1440 minutes; the value 0 means that the heartbeat checking:

- Is not run for z-centric agents.
- Is run for dynamic domain managers according to the policy established by the Z controller.

For details, see [Configuring the heartbeat checking interval for z-centric agents and dynamic domain managers on page 43](#).



Note: Any `pulseivl` value defined in a `ROUTOPTS HTTP|HTTPS destination` statement overrides the global value eventually defined with the `HTTPOPTS PULSEIVL` keyword.

The following example specifies the destinations to two IBM Z Workload Scheduler Agents:

```
ROUTOPTS HTTPS(ITAAC1:'192.27.144.12'/44112, ITAAC2:'192.14.55.231'/61424//10)
```

where a heartbeat checking interval of 10 minutes is set for ITAAC2. Note that if you use `pulseivl`, there must be two slashes (/) before it because the `type` parameter for z-centric agents is null.

The following example specifies the destination to one dynamic domain manager:

```
ROUTOPTS HTTPS(BRKS:'9.168.125.59'/8081/B/5)
```

where a heartbeat checking interval of 5 minutes is set for the dynamic domain manager. For more information about the ROUTOPTS statement, see *Customization and Tuning*.

The `proxyname` value is optional: you specify it when one or more z-centric agents are connected to the controller through a proxy server. You must have defined the `proxyname` in the PROXY parameter of ROUTOPTS.

The following example defines that the z-centric agents `ZCENT1` and `ZCENT2` are connected to the controller through the proxy server named `PRX1` (you defined `PRX1` in the PROXY parameter of ROUTOPTS):

```
ROUTOPTS HTTPS(ZCENT1:'192.27.144.12'/8081///PRX1, ZCENT2:'192.14.55.231'/16789//5/PRX1)
PROXY(PRX1:'195.34.134.10'/3741)
```

You can modify, add, or delete an HTTP or HTTPS destination while IBM Z Workload Scheduler is running. To make the changes immediately effective, without stopping and restarting the controller, issue the MODIFY command `/F procname, RFRDEST`.



Note: The RFRDEST command does not update any changes you make to the PROXY parameter in ROUTOPTS, it updates only the HTTP | HTTPS destinations, including the *proxy name*. For detailed information about RFRDEST, see *Managing the Workload*.

To list the HTTP and HTTPS destinations that are currently used by the controller, enter the following MODIFY command (the list is stored in the MLOG):

```
/F procname,DSPDEST
```

For detailed information about the DSPDEST command, see *Managing the Workload*.

You can add a maximum of 100 destinations, without having to restart the controller. The MODIFY command manages up to a total of 100 new destinations, regardless if you add them at once or at different times. After 100 destinations added, you must stop and restart the controller to have the MODIFY command manage other 100 new destinations.



Note: Deleting a destination does not increment the number of maximum additions allowed.

You can modify or delete an unlimited number of destinations. However, if you modify the destination *name* this operation is considered as adding a new destination.

Chapter 2. Customizing the HTTP connections

The communication between the IBM Z Workload Scheduler Agents and the IBM Z Workload Scheduler controller is direct, through the HTTP or HTTPS protocol.

This chapter includes the following customization topics for the HTTP/HTTPS connections:

- [Customizing the HTTP connection with the HTTPOPTS statement on page 42](#)
- [Customizing the HTTP timeout on the Dynamic Workload Console on page 42](#)

Customizing the HTTP connection with the HTTPOPTS statement

HTTPOPTS statement to customize HTTP connection.

You can optionally use the HTTPOPTS initialization statement to define a number of parameters for IBM Z Workload Scheduler Agents and dynamic domain managers, for example:

- Variable tables to be used when running job types with advanced options.
- Job log retrieval parameters.

You do not have to define the TCPIPJOBNAME and the HOSTNAME parameters if you already have a TCPOPTS initialization statement that includes them. Define the job log retrieval parameters if you want to change the defaults.

For a detailed description of the HTTPOPTS and TCPOPTS statements, see *Customization and Tuning*.

Customizing the HTTP timeout on the Dynamic Workload Console

You can increase the length of the HTTP connection timeout in the Dynamic Workload Console to allow more time for retrieving objects related to the z-centric environment.

If you find that you get message errors, such as:

```
AWSJZC019E A communication error has occurred with IBM Z Workload Scheduler.  
The connection wait has exceeded the timeout of "30,000" milliseconds...
```

when you try to retrieve z-centric objects from the z/OS engine on the Dynamic Workload Console, this implies that the HTTP/HTTPS connections within your z-centric environment might be suffering or that some of your plug-in applications require longer than expected.

To mitigate this problem you can increase the timeout allowed on the Dynamic Workload Console when receiving data from the IBM Z Workload Scheduler controller.

To do so, set the `zosHttpTimeout` option of the `TdwcGlobalSettings.xml` file that configures the global settings of the Dynamic Workload Console.

If the option is not set, it takes the default value of 30,000 milliseconds (ms), which is probably not enough to fulfill your requirements. In this case, add the option in `TdwcGlobalSettings.xml` and set a value larger than 30,000 ms (for example, 90,000 ms).

For additional information, see the *Dynamic Workload Console User's Guide*.

Configuring the heartbeat checking interval for z-centric agents and dynamic domain managers

You can configure the IBM Z Workload Scheduler controller to check at intervals if all or specific attached z-centric agents and dynamic domain managers are running. Without the use of this mechanism the controller does not register, until the next job submission occurs, that a workstation has become unavailable. This may impact your scheduling plans.

The use of this feature is optional and configurable by adding the PULSEIVL keyword in the following IBM Z Workload Scheduler initialization statements:

HTTPOPTS

The value for PULSEIVL is valid globally for all the attached z-centric agents and dynamic domain managers.

ROUTOPTS

PULSEIVL is featured as a parameter in the `HTTP/HTTPS destination` keyword and in this case defines the heartbeat checking interval of the specific agent defined in the destination. This option is useful when heartbeat checking is wanted only on the more strategic agents or when a different checking frequency is desired.

The pulse interval of a destination can be updated using the `HTTP refresh destination` command without having to stop the controller.

The value is expressed in minutes and can range from 0 to 1440. The value 0 means that the heartbeat checking:

- Is not run for z-centric agents.
- Is run for dynamic domain managers according to the policy established by the Z controller.

For example:

```
HTTPOPTS  ...
          PULSEIVL(60)
          ...

ROUTOPTS  HTTPS(DDM1:'162.17.129.32'/65232/B/5)
```

This configuration sets a heartbeat checking interval of 60 minutes for all z-centric agents and dynamic domain managers, but for DDM1 the interval is set to 5 minutes.

For details about the HTTPOPTS and ROUTOPTS initialization statements, see *Customization and Tuning*.

Configuring the connection between the IBM Z Workload Scheduler Agent and Z controller through a proxy

You can configure that one or more z-centric agents connect with the controller through a proxy server, by defining the proxy properties in the ROUTOPTS statement and `JobManager.ini` file.

About this task

On the controller, set the proxy properties in ROUTOPTS by performing the following steps:

1. Define one or more proxy destinations as follows:

```
ROUTOPTS PROXY (destination, ..., destination)
```

Each *destination* has the following syntax:

```
proxy name: 'IP address or hostname' /port
```

where:

proxy name

The name assigned to the destination, up to 8 alphanumerical characters.

IP address or hostname

The fully qualified host name or IP address used to communicate with the proxy server. It can be up to 52 alphanumeric characters.

port

The port number used to communicate with the proxy server.

2. In the HTTP|HTTPS parameter specify the *proxy name* that you defined, as follows:

```
ROUTOPTS HTTP|HTTPS (dest name: 'IPaddr or hostname' /port[/type] [/pulseiv] [/proxy name])
```

3. On the z-centric agent, edit the [ITA_Env] section of the `JobManager.ini` file as described in [Configuring proxy properties \[ITA_Env\] on page 21](#)

The following example shows a ROUTOPTS statement:

```
ROUTOPTS PROXY(PROX1: '192.27.144.13' /44113, PROX2: '192.27.144.14' /44118)      1
          HTTP(ZCENT1: '192.27.144.12' /44112//PROX2, ZCENT2: '192.14.55.231' /61424//PROX2)  2
          HTTPS(ZCENT3: '192.27.144.13' /8913/Z//PROX1)                          3
```

where:

1

Two proxy destinations, PROX1 and PROX2, are configured.

2

The z-centric agents ZCENT1 and ZCENT2 connect with the controller through proxy PROX2.

3

The z-centric agent ZCENT3 connects with the controller through proxy PROX1, in an SSL-secure connection.

Starting and stopping HTTP subtasks

An HTTP Server subtask and an HTTP Client subtask run the transmission of data between the scheduler and the IBM Z Workload Scheduler Agents on the distributed side. These subtasks automatically start at controller startup.

At anytime you can start or stop a subtask while IBM Z Workload Scheduler is running by using the MODIFY command. You can enter this command from a multiple console support (MCS) console or from a program such as the spool display and search facility (SDSF). In both cases, the terminal or console operator must have the required authority to enter operator commands.

Enter the MODIFY command as follows:

```
/F procname,modifyoption
```

where:

procname

Is the IBM Z Workload Scheduler controller JCL procedure name.

modifyoption

Can be one of the following:

S=taskname

Start the specified subtask.

P=taskname

Stop the specified subtask.

where *taskname* is either of the following:

HTC

The HTTP Client subtask

HTS

The HTTP Server subtask

Chapter 3. Defining user IDs and passwords with the USRREC statement

Use the USRREC initialization statement to define the passwords for the users who need to schedule jobs to run on Windows™ workstations. You do not need to do this if you run [native on page 99](#) job types on non-Windows workstations.

If you run job types with advanced options ([database on page 77](#), [file transfer on page 85](#), [web service on page 91](#), [xajob on page 95](#), [J2EE jms on page 97](#)) you can use USSREC regardless of the operating system. The advantage of doing this is that you do not have to hard code the password in the job definition as it will be retrieved from the IBM Z Workload Scheduler database at run time.

If the statement is very long and needs to be split into two lines, use the comma (,) as a continuation character to continue the string to the next line.

USRREC is defined in the member of the EQQPARM library specified by the USRMEM keyword in the HTTPOPTS statement.

You can make changes to the USRREC statement while IBM Z Workload Scheduler is running. To make them immediately effective, without stopping and restarting the controller, enter the following MODIFY operator command:

```
/F procname,RFRUSER
```

where *procname* is the IBM Z Workload Scheduler JCL procedure name.

For more information about the MODIFY command, see the section about modifying the scheduler in *Managing the Workload*. For more information about the USRREC statement, see [USRREC on page 46](#).

USRREC

Purpose

This statement defines the passwords for the users who need to schedule native jobs to run on Windows™ workstations, or who want to use this feature when defining job types with advanced options that will run on workstations based on any operating system.

After you have defined a user and its password, you use these values as follows in the definitions of job types with advanced options:

The value in USRNAM

- In the JOBREC job definition statement, it is specified as the value of the JOBUSR keyword.
- In the definition panel of the job in the Dynamic Workload Console it is specified in the *User Name* field of the *Credentials* section.

The value in USRPSW

Is resolved at run time if you:

- Specify `Y` for the JOBPWD keyword in the JOBREC job definition statement.
- Select `User` in the Password type widget that is displayed by clicking the ellipsis (...) located by the Password field in the Credentials section of the job definition panel in the Dynamic Workload Console.

Format

►► USRREC — USRCPU — (— *cpu name* —) — USRNAM — (— *logon ID* —) — USRPSW — (— *password* —) ►◄

Parameters

USRCPU(*cpu name*)

The IBM Workload Scheduler workstation on which the user can launch jobs. It consists of four alphanumeric characters, starting with a letter.

USRNAM(*logon ID*)

The user name of a Windows™ workstation or the name of the user authorized to run a job type with advanced options. The name is case-sensitive and can be of up to 47 characters.

If you are defining a user that will run native jobs on Windows workstations, note that:

- The name can include a domain name and might be case-sensitive, depending on the Windows™ version.
- The user must be able to log on to the computer on which IBM Workload Scheduler has launched jobs, and must also be authorized to *Log on as batch*.
- If the name is not unique in Windows™, it is considered to be either a local user, a domain user, or a trusted domain user, in that order.
- If you are defining a Windows™ domain user and the value contains a backslash (\), then the entire character string must be enclosed by single quotes, for example:

```
USRNAM('XXXXX\user1')
```

- If you are defining a Windows™ user in the *username@internet_domain* format that contains the at sign (@), for example `administrator@mywindow.com`, you must enclose the entire character string in single quotes:

```
USRNAM('administrator@mywindow.com')
```

USRPSW(*password*)

The password for the user specified in USRNAM. It can consist of up to 31 characters and must be enclosed in single quotation marks. It is case-sensitive.

The password is stored in the USRINFO member in plaintext, meaning that it is *not encrypted*. To encrypt it, run the sample provided in the EQQBENCR member of the EQQSAMP library.

If the password is for a user that will run native jobs on Windows workstations, note that:

- The password might be case-sensitive, depending on the Windows™ version.
- If you are defining a Windows™ domain user and the value contains a backslash (\), then the entire character string must be enclosed by single quotes, for example:

```
USRPSW('XXXXX\password1')
```

- If you are defining a password for the Windows™ user in the *username@internet_domain* format that contains the at sign (@), for example `administrator@mywindow.com`, you must enclose the entire character string in single quotes:

```
USRPSW('passw0rd')
```


Chapter 4. Defining variables and passwords for local resolution

You can set variables and passwords to be defined and resolved locally on the IBM Z Workload Scheduler agents.

This is particularly useful in the case of passwords because you are not required to specify them in the job definition. The advantage is that, if the password has to change, you do not modify the job definition, but you change it with the `param` command locally on the agents (or on the pool agents) that run or may run the job.

This feature is not restricted to Windows workstations alone. You can use it also on UNIX, as long as you apply it on job types with advanced options.

See the following sections for:

- [Creating and managing variables and passwords on the agents on page 49.](#)
- [Specifying variables and passwords in the JOBREC statement on page 52](#) in the JOBLIB data set.
- [Specifying local variables and passwords in job types with advanced options on page 53](#) to define your job in the Dynamic Workload Console so that the variables or passwords are resolved locally on the agents at run time.

Creating and managing variables and passwords on the agents

Use the `param` command to define and manage user passwords and variables locally on the IBM Z Workload Scheduler agents. The values are resolved at submission time on the agent where the job is submitted.

Authorization

To create, delete, or display variables or passwords, you must have Administrator or root user rights on the workstation that runs the agent or `TWS_user` rights on the agent.

Syntax

```
param -u | -V |  
    {-c | -ec} [file.section.|file..|section.] variable [value] |  
    [file.section.|file..|section.] variable |  
    {-d | -fd} [file.section.|file..|section.] variable
```

Arguments

-u

Displays command usage information and exits.

-V

Displays the command version and exits.

-c | -ec

Creates variable or password *variable* and defines its value *value*. The variable or password is placed in a namespace *file* that you can organize in one or more sections named *section*.

If you do not provide a file name *file*, the variable or password is placed in default file `jm_variables` in path `agent_installation_path\TWA\TWS\ITA\cpa\config\jm_variables_files` (`/TWA/TWS/ITA/cpa/config/jm_variables_files`) on the agent.

If you do not provide a section name *section*, the variable or password is placed in the main body of the file.



Important: If you are defining a password, you must specify a section named `password` for *variable*. This specifies that *variable* is a password.

If you are creating a variable, *variable* is the variable name and *value* is its value. If you are creating a password, *variable* is the user name and *value* is its password. If you do not enter *value* within the arguments, the command requests interactively to enter a value.

Argument **-c** creates the variable in clear form. Argument **-ec** creates the variable in encrypted form. Passwords are encrypted by default also if you use **-c**.

-d | -fd

Deletes (**-d**) or forces deletion (**-fd**) of a file, section, or variable (password). You can use the following wildcards:

Replaces one or more alphanumeric characters.

?

Replaces one alphanumeric character.

With **-d** the command asks for confirmation before deleting. With **-fd** it deletes without asking confirmation.

When you delete all the variables in a section, the section is removed from the file. When you delete all the sections and all the variables from a file, the file is removed.

file

The name of the file used as a namespace for *variable*. If you do not specify *file*, the command uses the default file `jm_variables` in path `agent_installation_path\TWA\TWS\ITA\cpa\config\jm_variables_files` (`/TWA/TWS/ITA/cpa/config/jm_variables_files`).

All the variable namespaces go in path

`agent_installation_path\TWA\TWS\ITA\cpa\config\jm_variables_files`
(`/TWA/TWS/ITA/cpa/config/jm_variables_files`).

section

The name of the section within *file* where *variable* is defined. When *variable* is used for a password, it must be placed in a section named `password`. No section name is required to store variables.

value

The value for *variable*.

variable

Can be a variable name or a user identification. If it is used for identification, it must be placed in a section named `password` within the namespace file.

Comments

To display a variable or password, a namespace file, or a section, use the command as follows:

```
param [file.section. | file.. | section.] variable
```

where you can use the * and ? wildcards described for the deletion command.

The namespace files, including default `jm_variables`, have no extension.

Variable names are case sensitive.

On IBM i systems, if you use the QP2TERM and the QSH shells, passwords are made visible during the creation process with `param` and are displayed clearly in the shell logs. To guarantee the obfuscation of a password, you need to use the AIXTERM or XTERM shells.

Example**Examples**

The command:

```
param -c compassets.hardware.platform1 unix
```

defines variable `platform1` with value `unix` in section `hardware` of the new or existing file named `compassets`. The value is not encrypted.

The command:

```
param -c compassets..platform1 unix
```

defines variable `platform1` with value `unix` in the new or existing file named `compassets`. The value is not encrypted.

The command:

```
param -ec hardware.platform1 unix
```

defines variable `platform1` with value `unix` in section `hardware` in the default file `agent_installation_path\TWA\TWS\ITA\cpa\config\jm_variables_files\jm_variables`. The value is encrypted.

The command:

```
param -c compassets.password.jladams san07rew
```

defines variable `jladams` with value `san07rew` in section `password` of the new or existing file named `compassets`. Since `jladams` is defined in section `password`, it is interpreted as a username. The value `san07rew` is encrypted by default since it is interpreted as a password.

The command:

```
param *.*.platform1
```

lists variable `platform1` in all its defined locations. That is:

```
..\TWA\TWS\ITA\cpa\config\jm_variables_files\compassets.hardware.platform1=unix
..\TWA\TWS\ITA\cpa\config\jm_variables_files\compassets..platform1=unix
..\TWA\TWS\ITA\cpa\config\jm_variables_files\jm_variables.hardware.platform1=**
```

The command:

```
param password.*adam*
```

lists all variables including the string `adam` contained in the `password` section of all files. In this case:

```
...\TWA\TWS\ITA\cpa\config\jm_variables_files\compassets.password.jladams=*****
```

The command:

```
param -d compassets.password.jladams
```

deletes variable `jladams`.

The command:

```
param -d compassets.password.*
```

deletes all the variables found in section `password` and therefore removes this section from file `compassets`.

The command:

```
param -d compassets.*.*
```

deletes all the contents (variables and sections containing variables) found in file `compassets` and therefore removes the file.

Specifying variables and passwords in the JOBRE statement

After defining a variable and its value with the `param` command, to add it within a job definition with the JOBRE statement, use the following format. The values are resolved at submission time on the agent where the job is submitted.

Format

```
//JOBREC
  JOBUSR(username)
  JOBPWD(AGENT)
//END JOBRE
```

where:

JOBUSR(username)

The name of the user to access WebSphere® Application Server, if required. If the user schedules jobs to run on Windows™ workstations, ensure that a user password is also defined with the JOBPWD keyword.

- If you are defining a Windows™ domain user, use the following format:

```
JOBUSR(domainName\user1)
```

- If you are defining a Windows™ user in the *username@internet_domain* format, use the following format:

```
JOBUSR('administrator@mywindow.com')
```

JOBPWD(AGENT)

Means that the password is resolved locally on the agent. The password must have been defined on the agent by issuing the [param on page 49](#) command. This feature is available independently of the operating system of the workstation.

Specifying local variables and passwords in job types with advanced options

After defining a variable and its value with the `param` command, to add it within a job definition so that it is resolved locally on the agent at run time, use the following syntax:

```
${agent:variable_name}
```

After defining a password with the `param` command, to add it within a job definition so that it is resolved locally on the agent at run time, use the following syntax:

```
${agent:password.user_name}
```

You can nest variables within passwords. If you used a variable to define a user, enter the password as follows within the job definition:

```
${agent:password.${agent:variable_name}}
```

where *variable_name* was previously defined as having value *user_name* with the `param` command.

Example**Example**

An IBM Workload Automation administrator needs to define a file transfer job that downloads a file from a remote server to one of the IBM Z Workload Scheduler Agents making up a pool of agents. The administrator wants to parametrize in the job definition:

- The name with which the remote file will be saved locally
- The remote user name and its password

The administrator proceeds in the following way on one of the agents:

1. Defines a variable named `localfile`. The variable is given a value equal to `./npp.5.1.1.Installer.DW.exe` and is created in a new variables file named `FTPvars` (no section). The command to do this is:

```
E:\IBM\TWA\TWS\CLI\bin>param -c FTPvars..localfile ./npp.5.1.1.Installer.DW.exe
```

2. Defines a variable named `remoteUser`. The variable is given a value equal to `FTPuser` and is created in the `FTPvars` file (no section). The command to do this is:

```
E:\IBM\TWA\TWS\CLI\bin>param -c FTPvars..remoteUser FTPuser
```

3. Defines the password for `FTPuser`. The password value is `tdwb8nxt` and is created in the `password` section of the `FTPvars` file. The command to do this is:

```
E:\IBM\TWA\TWS\CLI\bin>param -c FTPvars.password.FTPuser tdwb8nxt
```

4. With a text editor opens file `E:\IBM\TWA\TWS\ITA\cpa\config\jm_variables_files\FTPvars` and checks its contents:

```
localfile = ./npp.5.1.1.Installer.DW.exe
remoteuser = FTPuser

[password]
FTPuser = {aes}XMMYMY2zBHvDEDBo5DdZVmw0Jao60pX1K6x2HhRcovA=
```

5. Copies file `FTPvars` in the `agent_installation_path\TWA\TWS\ITA\cpa\config\jm_variables_files` path of every other agent defined in the pool.
6. Starts defining the new file transfer job in the Workload - Design windows of Dynamic Workload Console. In the `FileTransfer` window:
 - a. Enters `${agent:FTPvars..localfile}` in the `Local file` field.
 - b. Enters `${agent:FTPvars..remoteuser}` in the `Remote Credentials →; User Name` field.
 - c. Clicks the `...` button next to the `Remote Credentials →; Password` field. The `Password type` window pops up and the administrator selects `Agent User`.
 - d. After the administrator clicks the `OK` button for confirmation in the popup window, the `Remote Credentials →; Password` field is complete with the `${agent:password.${agent:FTPvars..remoteuser}}` value.
7. Type in all the other fields to complete the job definition.

When the job is run, the entities and the password entered as variables are resolved with the values defined in the `FTPvars` file.

Chapter 5. Defining and managing IBM Z Workload Scheduler Agent workstations in the database

After defining the destinations of your IBM Z Workload Scheduler Agents, you need to define the IBM Z Workload Scheduler Agents as IBM Z Workload Scheduler computer automatic workstations.

You can define the IBM Z Workload Scheduler Agents as IBM Z Workload Scheduler computer automatic workstations by using either of the following:

ISPF

The workstation definition and management panels in ISPF include the **Z-CENTRIC AGENT** field. You can take the following actions on IBM Z Workload Scheduler Agent workstation definitions:

- Creating on page 55
- Modifying on page 56
- Browsing on page 56
- Deleting on page 57
- Listing on page 57

Dynamic Workload Console

For details, refer to the Dynamic Workload Console online help.

Creating

Figure 4: EQQWCGEP - Creating general information about an IBM Z Workload Scheduler Agent workstation on page 55

shows the CREATING GENERAL INFORMATION ABOUT A WORK STATION panel, used to create general information about an IBM Z Workload Scheduler Agent workstation named ZCEN.

Figure 4. EQQWCGEP - Creating general information about an IBM Z Workload Scheduler Agent workstation

```

EQWQCGEP ----- CREATING GENERAL INFORMATION ABOUT A WORK STATION -----
Command ==>

Enter the command R for resources or A for availability or O for end-to-end
options, or enter/change data below:

WORK STATION NAME      ==> ZCEN
DESCRIPTION             ==> z-centric workstation
WORK STATION TYPE      ==> C      G General, C Computer, P Printer
                                R Remote Engine
REPORTING ATTR         ==> A      A Automatic, S Manual start and completion
                                C Completion only, N Non reporting
PRINTOUT ROUTING       ==> SYSPRINT The ddname of daily plan printout data set
SERVER USAGE           ==> B      Parallel server usage, B, N, P, or C
DESTINATION            ==> ITAAC1  Name of destination
Options: allowed Y or N
  SPLITTABLE           ==> N      JOB SETUP              ==> N
  STARTED TASK, STC    ==> N      WTO                      ==> N
  AUTOMATION           ==> N
  WAIT                 ==> N      Z-CENTRIC AGENT         ==> Y
  VIRTUAL              ==> N      DYNAMIC                  ==> N

REMOTE ENGINE TYPE     ==>        Z z/OS or D Distributed
Defaults:
TRANSPORT TIME         ==> 00.00   Time from previous work station HH.MM
DURATION               ==> 00.05.00 Duration for a normal operation HH.MM.SS

```

Define an IBM Z Workload Scheduler Agent as a computer automatic workstation and enter the destination name previously defined in the ROUTOPTS initialization statement. The Z-CENTRIC AGENT option excludes all other options.

Modifying

Figure 5: EQQWMGEP - Modifying an IBM Z Workload Scheduler Agent workstation description on page 56 shows the MODIFYING GENERAL INFORMATION ABOUT A WORK STATION panel, used to modify the database definition of an IBM Z Workload Scheduler Agent workstation named ZCEN.

Figure 5. EQQWMGEP - Modifying an IBM Z Workload Scheduler Agent workstation description

```

EQQWMGEP ----- MODIFYING GENERAL INFORMATION ABOUT A WORK STATION -----
Command ==>
Enter the command R for resources A for availability O for end-to-end options
or D for Destinations above, or enter data below:

Work station name      : ZCEN
DESCRIPTION            ==> z-centric workstation-----
WORK STATION TYPE      ==> C          G General, C Computer, P Printer
REPORTING ATTR         ==> A          A Automatic, S Manual start and completion
                                C Completion only, N Non reporting
PRINTOUT ROUTING       ==> SYSPRINT  The ddname of daily plan printout data set
SERVER USAGE           ==> N          Parallel server usage C , P , B or N
DESTINATION            ==> ITAAC1__  Name of destination
Options: allowed Y or N
SPLITTABLE             ==> N          JOB SETUP              ==> N
STARTED TASK, STC      ==> N          WTO                     ==> N
AUTOMATION              ==> N          Z-CENTRIC AGENT         ==> Y
WAIT                    ==> N          DYNAMIC                 ==> N
VIRTUAL                 ==> N
Defaults:
TRANSPORT TIME         ==> 00.00      Time from previous work station HH.MM
DURATION                ==> -----  Duration for a normal operation HH.MM.SS

```

Browsing

Figure 6: EQQWBGEF - Browsing an IBM Z Workload Scheduler Agent workstation description on page 56 shows the BROWSING A WORK STATION DESCRIPTION panel, used to browse the database description of an IBM Z Workload Scheduler Agent workstation named ZCEN.

Figure 6. EQQWBGEF - Browsing an IBM Z Workload Scheduler Agent workstation description

```

EQQWBGEF ----- BROWSING A WORK STATION DESCRIPTION -----
Command ==>

Enter the command R for resources, A for availability, O for end-to-end options
or D for Destinations above

Work station           : ZCEN
Description             : z-centric workstation

Work station type      : Computer
Reporting attribute    : Automatic
Printout routing       : SYSPRINT
Server usage           : Neither
Destination            : ITAAC1

Splittable             : No      Job setup              : No
Started task STC      : No      WTO                     : No
AUTOMATION              : No      z-centric agent         : Yes
WAIT                    : No      Dynamic                 : No
VIRTUAL                 : No
Transport time         : 00.00
Duration               :

```


Deleting

Figure 7: EQQWDGEP - Deleting an IBM Z Workload Scheduler Agent workstation description. on page 57 shows the CONFIRMING DELETION OF A WORK STATION DESCRIPTION panel, used to delete the database description of an IBM Z Workload Scheduler Agent workstation named ZCEN.

Figure 7. EQQWDGEP - Deleting an IBM Z Workload Scheduler Agent workstation description.

```

EQQWDGEP ----- CONFIRMING DELETION OF A WORK STATION DESCRIPTION -----
Command ==>

Enter Y in the command field to confirm deletion or
enter N to reject deletion.

Work station      : ZCEN
Description       : Accts Receivable

Work station type : Computer
Reporting attribute : Automatic

Printout routing  : SYSPRINT
Server usage      : Neither
Destination       : ITAAC1
Splittable        : No      Job setup      : No
Started task STC  : No      WTO            : No
AUTOMATION        : No
WAIT              : No      z-centric agent : Yes
VIRTUAL           : No      Dynamic        : No
Transport time    : 00.00
Duration          :

```

Listing

Figure 8: EQQWWSEP - Specifying list selection criteria for IBM Z Workload Scheduler Agent workstations on page 57 shows the SPECIFYING WORK STATION LIST CRITERIA panel, used for specifying selection criteria to list IBM Z Workload Scheduler Agent workstation definitions.

Figure 8. EQQWWSEP - Specifying list selection criteria for IBM Z Workload Scheduler Agent workstations

```

EQQWWSEP ----- SPECIFYING WORK STATION LIST CRITERIA -----
Command ==>

Specify selection criteria below and press ENTER to create a list.

WORK STATION NAME ==> ----
DESTINATION       ==> -----
TYPE              ==> ---      G, C, P in any combination, or blank
REPORTING ATTRIBUTE ==> ----      A, S, C, N in any combination or blank
AUTOMATION        ==> -          Y, N or blank
WAIT Work station ==> -          Y, N or blank
VIRT Work station ==> -          Y, N or blank
z-Centric Agent   ==> Y          Y, N or blank
Dynamic           ==> -          Y, N or blank

```

Managing the status of the IBM Z Workload Scheduler Agent workstation in the plan

Use the WSSTAT command when you want to:

- Change the status of an IBM Z Workload Scheduler Agent workstation in the current plan.
- Verify if the communication between the IBM Z Workload Scheduler Agent workstation and the controller is active.

You use WSSTAT on IBM Z Workload Scheduler Agents in the same way as you do with any other computer automatic workstation.

The status information is communicated to the controller to indicate that a workstation is in one of the following states:

ACTIVE

The workstation is running and scheduling its own plan.

OFFLINE

Communication is lost between the controller and the IBM Z Workload Scheduler Agent workstation on the system that the workstation represents. The HTTP client retries to establish the connection until it is successful.

FAILED

You manually set the workstation status to FAILED.

When you run the WSSTAT command, you can optionally define restart and routing options for the workload defined on the workstation when you are reporting a status of OFFLINE or FAILED.

You can invoke WSSTAT as a TSO command or by using a batch job which runs program EQQEVPGM. If you invoke WSSTAT as a TSO command, you must allocate the EQQMLIB data set to the address space of the TSO user, either by adding DD statements to the logon procedure, or by using the ALLOC command after TSO logon. In the TSO environment, error messages and trace records are sent directly to the terminal user. Messages are not delivered to indicate successful command execution.

Use of the WSSTAT command can be restricted with fixed resource code RL and subresource RL.WSSTAT. The authority of the requester is verified by the subsystem name identified in the command if an AUTHDEF statement is defined for that subsystem. When SUBSYS(MSTR) is specified, all tracker subsystems defined on the z/OS® system where the WSSTAT command is issued attempt to verify the authority of the requester before an event is generated. A requester might be rejected by one subsystem and accepted by another.

The subsystem to which you direct the command need not be active when the command is issued. An event is generated and queued in CSA together with other job-tracking events. If the subsystem is not active when the command is issued, the authority of the requester is verified using the class name specified in the AUTHDEF statement when the subsystem was last started. If the subsystem has not been started since a z/OS® IPL, no authority verification can be performed.



Note: If you used the panels to set the status of a workstation to OFFLINE, you cannot reset it to ACTIVE with WSSTAT.

Automatic and manual changes to the status of the IBM Z Workload Scheduler Agent workstation

At startup, the controller does not establish a connection with an IBM Z Workload Scheduler Agent workstation until the first job is submitted on that IBM Z Workload Scheduler Agent. The workstation status is automatically set to ACTIVE with extended status Waiting for first submission.

When the first job is submitted, the communication between the controller and agent is opened and the workstation is set to ACTIVE.

You can manually change the IBM Z Workload Scheduler Agent workstation status in the following ways:

- From ACTIVE to OFFLINE.
- From ACTIVE, Waiting for first submission to ACTIVE. This operation is also useful to check if the communication between the controller and the agent is established.
- From OFFLINE to ACTIVE.

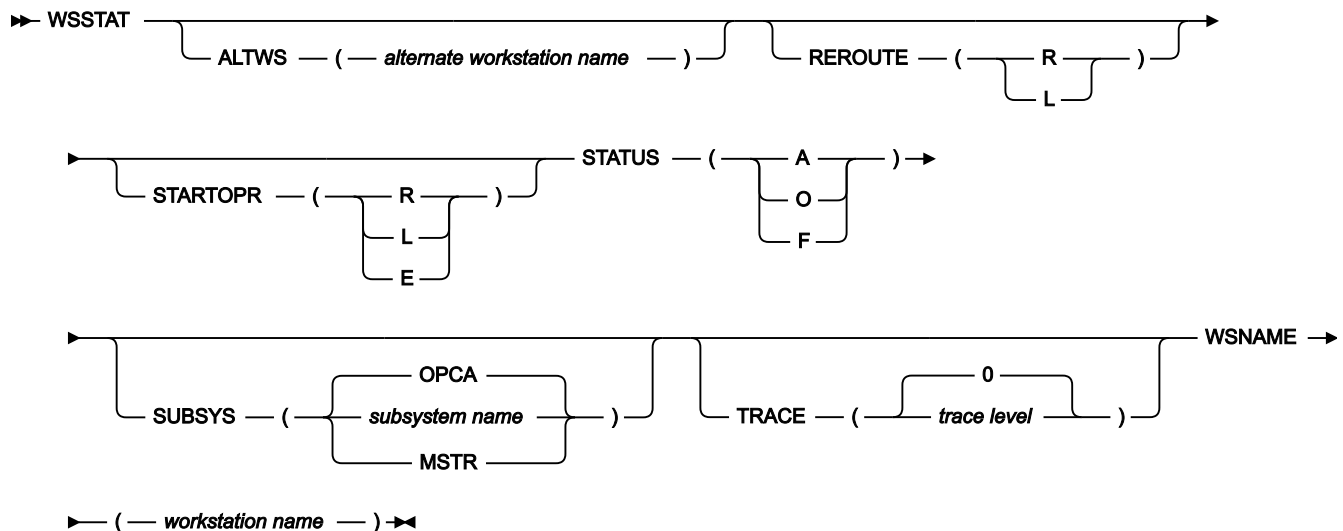
You cannot set the agent workstation to ACTIVE, if it was set to OFFLINE automatically. In this case, message EQQWL17W is issued:

```
WORK STATION WSNNAME CANNOT BE VARIED TO ACTIVE STATUS.
A QUERY AGENT INFO REQUEST HAS BEEN SENT TO THE AGENT AT DESTINATION DEST.
```

and the controller tries to establish a connection with the IBM Z Workload Scheduler Agent. If the attempt is successful, the workstation status is set to ACTIVE by the controller.

WSSTAT

Format



Note: These are the only WSSTAT parameters that apply to the IBM Z Workload Scheduler Agent workstation. For a complete list of the WSSTAT parameters, see *Managing the Workload*.

Parameters

ALTWS (*alternate workstation name*)

When the workstation status is set to OFFLINE or FAILED, you can specify the alternate workstation on which to start reroutable operations.

If you omit this parameter, the value defined for the current workstation open interval is used. If the REROUTE parameter specifies L, or if the default specifies no rerouting, the value of ALTWS is ignored.

The parameter is optional.

REROUTE (R | L)

When the workstation status is set to OFFLINE or FAILED, you can specify R for operations to be rerouted to the alternate workstation, or L for no rerouting (to leave the operations on the inactive workstation).

If you omit this parameter, the value defined in either the WSOFFLINE or the WSFAILURE keyword on the JTOPTS initialization statement is used as the default.

The parameter is optional.

STARTOPR (R | E | L)

When the workstation status is set to OFFLINE or FAILED, you can specify what the scheduler must do with operations that are currently in started status at the workstation, where:

R

Restart operations automatically on the alternate workstation.

E

Set all started operations to ended-in-error.

L

Leave the operations in started status.

If you omit this parameter, the value defined in either the WSOFFLINE or the WSFAILURE keyword on the JTOPTS initialization statement is used as the default.

The parameter is optional.



Note: If you select STARTOPR(E), a started job continues to run, because IBM Z Workload Scheduler never cancels jobs that have started.

STATUS (A | O | F)

The status you want to report for the workstation:

A

Active

O

Offline

F

Failed

You must specify either STATUS or one of the alternative parameters.

SUBSYS (*subsystem name* | **MSTR** | **OPCA**)

The name of the tracker subsystem that `WSSTAT` is directed to. This parameter can be up to 4 characters long. The first character must be alphabetic while the remaining characters must be alphanumeric. All lowercase characters are converted to uppercase.

If you specify **MSTR**, the command is directed to all tracker subsystems on the z/OS® system on which the command is issued.



Note: If the tracker and controller in your configuration run on different subsystems, specify the name of the tracker subsystem in this parameter.

TRACE (*level* | **0**)

The event tracing indicator. When any nonzero positive number is specified, a trace entry is created for each event generated by `WSSTAT`. The trace record is written to the message log file identified by ddname `EQQMLOG`. The record identifies the name of each receiving subsystem. The default value 0 does not generate trace records.

WSNAME (*workstation name*)

The name of the workstation to be updated. You must specify this parameter.

The `WSSTAT` parameters are checked for validity and consistency. The validity checks are performed on the tracker where the command is run. The validity check processes parameter names, and length and type of parameter values.

If the input is valid, a *workstation status event* is generated and communicated to the controller. Processing of the event includes a consistency check of the values specified in the parameters. The following consistency checks are made:

- The workstation name exists.
- The alternate workstation exists.
- If the value given in the STATUS parameter is the same as the current status of the workstation, the command is ignored.

Example

Examples

The following two examples show how you can use `WSSTAT` in TSO, or in a batch job (using the batch program `EQQEVPGM`).

Example 1 - TSO command

```
ALLOC F(EQQMLIB) DA('OPC.MESSAGE.LIBRARY') SHR REUSE
WSSTAT SUBSYS(OPCB) WSNAME(AS4H) STATUS(0) START(R)
```

In this example, the status of workstation AS4H is set to OFFLINE. Started operations are restarted on the alternate workstation.

Example 2 - Batch job

```
//CHSTATUS JOB (ACCOUNT),'Change WS status',CLASS=A
//STEP1 EXEC PGM=EQQEVPGM
//STEPLIB DD DSN=OPC.LOAD.MODULE.LIBRARY,DISP=SHR
//EQQMLIB DD DSN=OPC.MESSAGE.LIBRARY,DISP=SHR
//EQQMLOG DD SYSOUT=A
//SYSIN DD *
WSSTAT SUBSYS(OPCB) WSNAME(AS4H) STATUS(A)
/*
```

In this example, the status of workstation AS4H is set to ACTIVE.

Browsing system information of IBM Z Workload Scheduler Agent workstations

You can view the system information of an IBM Z Workload Scheduler Agent workstation by entering the I row command in the following panels:

- Modifying Workstations in the Current® Plan (EQQMWSLL)
- Browsing Summary of Activities at a Workstation (EQQSWSSP)

Figure 9: EQQSWIZC - Browsing an IBM Z Workload Scheduler Agent workstation system information on page 62 shows what type of information is provided in the panel.

Example

Figure 9. EQQSWIZC - Browsing an IBM Z Workload Scheduler Agent workstation system information

```
EQQSWIZC ----- BROWSING Z-CENTRIC SYSTEM INFORMATION -----
Command ==>

Work station      : LWE1           Accts Receivable
Destination      : ITAAC1         HTTP
Code page        : IBM-037
Hostname         : AR11.ZAPHOD.COM
Port number      : 44112

System Information:
Operating system name : WINDOWS
Operating system level:
Agent version       : 8.6
Agent fixpack level  : 0

Default JOBREC values
JOBUSR            : user01
JOBPWD           : Y
JOBTYPE          : accessmeth1

User name
Password (Y/N/A)
Access Method
```

Chapter 6. Defining and managing dynamic workstations

When you add dynamic capabilities to your environment, you can schedule your workload dynamically.

You can add dynamic capabilities to your environment by connecting the Z controller to a dynamic domain manager. The dynamic domain manager is installed in the distributed environment and manages IBM Z Workload Scheduler Agents with dynamic capabilities, also installed in the distributed environment.

The type of dynamic capabilities you add to your IBM Workload Scheduler agents depends on the type of workload you plan to run:

- If you plan to run existing IBM Workload Scheduler jobs dynamically, install the IBM Z Workload Scheduler Agents with dynamic capabilities and connect the agents to a dynamic domain manager connected to an IBM Z Workload Scheduler controller.
- If you plan to run both existing IBM Workload Scheduler jobs and job types with advanced options dynamically, install the IBM Z Workload Scheduler Agents with dynamic capabilities and Java™ run time and connect the agents to a dynamic domain manager connected to an IBM Z Workload Scheduler controller.

In the z/OS® environment, you can have the following dynamic workstations:

Broker

Also known as dynamic domain manager in the distributed environment. An installed component in an IBM Workload Scheduler distributed or z/OS® network that is the management hub in a domain where you run both static and dynamic workload. All communications to and from the dynamic agents in the domain are routed through the dynamic domain manager. You can use this workstation to submit jobs by reference. You can define the broker workstation either from ISPF panels or from the Dynamic Workload Console. For more information, see *IBM Workload Scheduler: Planning and Installation*.

Pool

A workstation that groups a set of IBM Z Workload Scheduler Agents with dynamic capabilities that have similar hardware or software characteristics to submit jobs to. IBM Workload Scheduler balances the jobs among the IBM Z Workload Scheduler Agents with dynamic capabilities within the pool and automatically reassigns jobs to available agents if an agent is no longer available. To create a pool of IBM Z Workload Scheduler Agents with dynamic capabilities, define a workstation of type pool hosted by the workload broker workstation, then select the IBM Z Workload Scheduler Agents with dynamic capabilities you want to add to the pool. You can define the pool only using the Dynamic Workload Console.

Dynamic pool

A workstation that groups a set of IBM Z Workload Scheduler Agents with dynamic capabilities, which is dynamically defined based on the resource requirements you specify. For example, if you require a workstation with low CPU usage and Windows™ installed to run your job, you specify these requirements using the Dynamic Workload Console. This workstation maps all the IBM Z Workload Scheduler Agents with dynamic capabilities in your environment that meet the requirements you specified. The resulting pool is dynamically updated

whenever a new suitable IBM Z Workload Scheduler Agent with dynamic capabilities becomes available. You can define the dynamic pool only using the Dynamic Workload Console.



Note: For these types of workstations, the WSOFFLINE keyword of the JTOPTS initialization statement of the controller is forced to the LEAVE value.

To enable dynamic end-to-end communication, perform the following steps:

1. Install the dynamic domain manager in the distributed environment. For more information, see *IBM Workload Scheduler: Planning and Installation*.
2. Establish a connection between the Z controller and the dynamic domain manager: in the ROUTOPTS statement, set the DESTINATION keyword to the host name of the dynamic domain manager previously installed, in the following format:

```
DESTINATION NAME=DDM_hostname/DDM:port_number/B
```

3. Define the dynamic domain manager workstation. Depending on how you plan to use the dynamic domain manager, the procedure varies:

Submission of jobs by reference

Define the dynamic domain manager using the following ISPF panels:

- a. CREATING GENERAL INFORMATION ABOUT A WORKSTATION
- b. WORKSTATION END-TO-END OPTIONS

For more information, see [Creating broker workstations on page 64](#).

Dynamic submission of jobs to pools and dynamic pools

Define the dynamic domain manager in the Dynamic Workload Console using the **Create Workstations** panel. For more information, see [Creating pools and dynamic pools on page 66](#).

4. Install the IBM Z Workload Scheduler Agents with dynamic capabilities.
5. *Optionally*, define the pools and dynamic pools in the distributed environment using the Dynamic Workload Console.

To schedule jobs from IBM Z Workload Scheduler to a distributed environment, you have the following options:

- Submission to pools and dynamic pools. For more information, see [Submitting jobs to pools or dynamic pools on page 66](#).
- Submission of jobs defined on the dynamic domain manager. For more information, see [Submitting jobs by reference on page 67](#).

Creating broker workstations

Create broker workstations using the ISPF panels and the Dynamic Workload Console.

You can use broker workstations to submit jobs by reference, as explained in [Submitting jobs by reference on page 67](#). To define a broker workstation, perform the following steps:

1. Define a broker workstation as a computer automatic workstation, as described in [Figure 10: EQQWCGEP - Creating general information about a broker workstation on page 65](#) and enter the destination name previously defined in the ROUTOPTS statement. The DYNAMIC option excludes all other options.

Figure 10. EQQWCGEP - Creating general information about a broker workstation

```

EQQWCGEP ----- CREATING GENERAL INFORMATION ABOUT A WORK STATION -----
Command ==>

Enter the command R for resources or A for availability or 0 for end-to-end
options, or enter/change data below:

WORK STATION NAME    ==> DYNW
DESCRIPTION          ==> dynamic workstation
WORK STATION TYPE    ==> C          G General, C Computer, P Printer
                                R Remote Engine
REPORTING ATTR       ==> A          A Automatic, S Manual start and completion
                                C Completion only, N Non reporting
PRINTOUT ROUTING     ==> SYSPRINT  The ddname of daily plan printout data set
SERVER USAGE         ==> B          Parallel server usage, B, N, P, or C
DESTINATION          ==> ITAAC1     Name of destination
Options: allowed Y or N
SPLITTABLE           ==> N          JOB SETUP              ==> N
STARTED TASK, STC    ==> N          WTO                    ==> N
AUTOMATION            ==> N          Z-CENTRIC AGENT       ==> N
WAIT                  ==> N          DYNAMIC                ==> Y
VIRTUAL               ==> N
REMOTE ENGINE TYPE   ==>           Z z/OS or D Distributed

Defaults:
TRANSPORT TIME       ==> 00.00      Time from previous work station HH.MM
DURATION              ==> 00.05.00  Duration for a normal operation HH.MM.SS

```

2. Select 0 to open the WORK STATION END-TO-END OPTIONS panel.
3. Select BROKER=Y.

Figure 11. EQQWCEOD - Defining end-to-end options

```

EQQWCEOD ----- WORK STATION END-TO-END OPTIONS-----
Command ==>

Enter/change data below:

WORK STATION NAME    ==> BRK

Default JOBREC values on this workstation:
JOBUSER              ==>           User name
JOBPWD               ==>           Password (Y/N/A)
JOBTYP               ==>           Access method

Dynamic options:
BROKER               ==> Y

```

For more information, see the section about creating workstations in *IBM Z Workload Scheduler: Managing the Workload*.

Defining a backup dynamic domain manager

How you can define a backup dynamic domain manager.

To define a backup dynamic domain manager, perform the following steps:

1. Install the backup dynamic domain manager in the distributed environment. For more information, see *IBM Workload Scheduler: Planning and Installation*.
2. Establish a connection between the Z controller and the backup dynamic domain manager: in the ROUTOPTS statement, set the DESTINATION keyword to the host name of the backup dynamic domain manager previously installed, in the following format:

```
DESTINATION NAME=backup_DDM_hostname/backup_DDM:port_number/B
```

3. Define a dynamic workstation with type broker.
4. Define the dynamic workstation as alternate workstation of the workstation corresponding to the dynamic domain manager, the pool, and the dynamic pool.

You do not need to define pools and dynamic pools associated to the backup dynamic domain manager. When the dynamic domain manager fails and the backup dynamic domain manager takes over, the backup dynamic domain manager automatically inherits the dynamic agents, pools, and dynamic pools defined on the dynamic domain manager.

Creating pools and dynamic pools

You can create pools and dynamic pools using the Dynamic Workload Console.

To create pools and dynamic pools using the Dynamic Workload Console, perform the following steps:

1. In the Z controller, define the ROUTOPTS statement by setting the DESTINATION parameter to the host name of the dynamic domain manager. For detailed information about ROUTOPTS, see *Customization and Tuning*.
2. From the Dynamic Workload Console.
3. Click **Design > Workload Designer**.
4. Select a z/OS® engine from the list and click **Create new > Workstation**.
5. Select **Pool** as the category of workstation you want to create.
6. In the properties pane, specify the workstation attributes in the **General Info**, **Resources**, and **Defaults** tabs. Depending on the type of workstation you select, some attributes are required.
7. Click **Save**.

For more information about workstation characteristics and attributes, see the Dynamic Workload Console online help.

Submitting jobs to pools or dynamic pools

About this task

To submit jobs to a pool or dynamic pool, perform the following steps:

1. Define a [JOBREC on page 75](#) statement using the appropriate syntax, as explained in [Defining a job in the JOBLIB data set on page 74](#). Alternatively, you can create the job from the Dynamic Workload Console.
2. Submit the JOBREC statement to the workstation defined in [Creating pools and dynamic pools on page 66](#). The job is submitted to the workstation you specified. IBM Workload Scheduler dynamically assigns the job to the best available resource.

Submitting jobs by reference

About this task

When you submit a job by reference, you create a job definition in the distributed environment and submit it from IBM Z Workload Scheduler to the distributed environment. In the distributed environment, the job is dynamically assigned to the best available resource. You write in the IBM Z Workload Scheduler JCL only the job name of the job, without having to write or import the whole job into the JCL. You can submit jobs by reference only on broker workstations.

To submit jobs by reference from IBM Z Workload Scheduler, perform the following steps:

1. Define a job with name **broker_test** by using the Dynamic Workload Console. This job contains the operations to be performed:
2. Create a JCL, by using the ISPF panels or the Dynamic Workload Console, specifying the name of the job you created in step 1. This references the job in the distributed environment.
3. Submit the JCL. As a result, the **broker_test** job is submitted in the distributed environment.

Chapter 7. Setting the security features

IBM Z Workload Scheduler provides a secure, authenticated, and encrypted connection mechanism for communicating across the network. This mechanism is based on the Secure Sockets Layer (SSL) protocol.

By default, the IBM Z Workload Scheduler Agent is installed with the SSL protocol enabled. According to this default, in the HTTPOPTS statement you must set the connection security for end-to-end communication to enable SSL.

According to the certificates that you are using (USS or SAF) to enable the SSL communication see the relevant section:

- [Customizing the SSL connection with the Z controller when using your certificates \(SAF\) on page 68](#)
- [Customizing the SSL connection with the Z controller when using your certificates \(USS\) on page 71](#)

Customizing the SSL connection with the Z controller when using your certificates (SAF)

The management of security certificates is different between product versions. Depending on the z-centric agent version you are using, customize the SSL connection between the agent and Z controller by either using your SAF certificates or the default certificates.

About this task

Depending on the agent version that you are using:

For an agent version 10.2.1, or later

Customize the Z controller by using the SAF (System authorization facility) interface, the agent certificates and the configuration file. For more information, see [Customizing the SSL connection between the agents and Z controller when using your certificates on page 68](#).

For an agent version 10.2.0, or earlier

You can either use default certificates or create your own. For more information, see [Setting SSL-secure connections for communication using the default certificates on page 70](#).

Customizing the SSL connection between the agents and Z controller when using your certificates

About this task

To communicate, the IBM Z Workload Scheduler Agents (z-centric agents) and the Z controller use the HTTPS protocol. The communication process uses the certificates obtained by customizing the Z controller using the SAF (System authorization facility) interface. In addition to customizing those certificates, you need to customize the agent certificates and the configuration file. To enable SSL communication, perform the following steps.

1. Generate the distributed certificates for the agent as follows:

If the certification authority (CA) does not exist

Run the following commands:

```
° openssl genrsa -out ca.key 4096
° openssl req -x509 -new -nodes -key ca.key -subj "/CN=<common_name>" -days 3650 -out ca.crt
° openssl genrsa -des3 -out tls.key 4096
° openssl req -new -key tls.key -out tls.csr
° openssl x509 -req -in tls.csr -CA ca.crt -CAkey ca.key -CAcreateserial -out tls.crt -extfile /etc/pki/tls/openssl.cnf -extensions v3_req
```

If the certification authority (CA) already exists

- a. Run the following command to generate the private key and unsigned certificate:

```
▪ openssl genrsa -des3 -out tls.key 4096
▪ openssl req -new -key tls.key -out tls.csr
```

- b. Send the `tls.csr` file to your certificate administrator to have it signed with your CA and be resent to you.

The `ca.crt`, `tls.crt`, and `tls.key` files are now ready for use.

2. Store `ca.crt`, `tls.crt`, and `tls.key` on the agent in a folder of your choice. When running the `twinsinst` script to install the agent, specify that folder with the **`sslkeysfolder`** parameter.
3. Generate a z/OS certificate as follows:
 - a. Create a z/OS certificate and save it with the `.crt` extension.
 - b. Export the certificate in ASCII mode to the distributed environment.
 - c. On the workstation where you plan to install the agent, create a folder named `additionalCAs` nested in the folder where you previously stored the distributed certificates created in [step 2 on page 69](#).
 - d. Store the certificate in the `additionalCAs` folder. The `additionalCAs` folder must also contain the public key certificate or public certificate chain of the Z controller SSL key ring.
4. Import the distributed certificates (`ca.crt` and `tls.crt`) in the z/OS environment in ASCII mode.
5. Install the agent by running the `twinsinst` script and specifying the `sslkeysfolder` and `sslpassword` parameters.

Consider the following example:

```
./twinsinst -new -uname <agent_name> -acceptlicense yes
-addjruntime true -inst_dir <inst_dir> -jimport xxxxx -jimportssl true -sslkeysfolder <path_to_distr_cert>
-sslpassword <keystore_password>
```

where:

new

A fresh installation of the agent.

uname

Name of the user for which the agent is being installed.

acceptlicense

Whether to accept the License Agreement.

addjruntime

Adds the Java™ run time to run job types with advanced options, both those types that are supplied with the product and the additional types that are implemented through the custom plug-ins.

inst_dir

Folder of the agent installation.

jimport

JobManager port number used by the Z controller to connect to the agent.

jimportssl

JobManager port used by the Z controller to connect to the agent. Supported values are `true` and `false`.

sslkeysfolder

Name and path of the folder on the agent containing certificates. The folder must contain the following items:

- `ca.crt`
- `tls.crt`
- `tls.key`
- `additionalCAs` folder

sslpassword

Password to access the certificates.

Results

The agent is now completely configured.

Setting SSL-secure connections for communication using the default certificates

About this task

To provide SSL security for an HTTP connection between the Z controller and HCL Workload Automation Agent, set the HTTPS keyword in the in the ROUTOPTS statement. For details about ROUTOPTS, see *Customization and Tuning* statement.

At installation time, the default security certificates are automatically stored in the SEQQDATA library:

EQQCERCL

The security certificate for the client.

EQQCERSR

The security certificate for the server.

You can decide to use these default certificates or create your own. However, in a production environment, it is recommended that you customize SSL communication with your own certificates.

In both cases, you need to import them into your security system. If you are using RACF, you are provided with the EQQRCERT sample job that you can run to import the certificates. To run this job, ensure that you use the same user ID that RACF associates with the controller started task.

The EQQRCERT job:

- Copies the EQQCERCL certificate to a temporary sequential data set
- Copies the EQQCERSR certificate to a temporary sequential data set
- Imports EQQCERCL to RACF
- Imports EQQCERSR to RACF
- Deletes the temporary sequential data sets
- Creates the SAF key ring that is used to connect the imported certificates
- Updates the RACF database with the new certificates and key ring

Customizing the SSL connection with the Z controller when using your certificates (USS)

About this task

To communicate, the z-centric agents and Z controller use the HTTPS protocol. The communication process uses the certificates obtained by customizing the Z controller certificates with the USS (UNIX System Services) component. In addition to customizing those certificates, you need to customize the agent certificates and the configuration file. To enable SSL communication, perform the following steps:

1. Generate a .kdb CMS key store file. This file must contain a private key trusted by the Z controller to which the agent is registered, and the Z controller public key to allow the agent to trust it.
2. Save the password of the key store in a stash file that has the same name as the file that you generated in [step 1 on page 71](#) and give it extension .sth.
3. Edit the ita.ini agent configuration file by setting the following properties to the values specific for your environment:

```
cert_label=<label_agent_private_key>
key_db_name=<file_name>
key_repository_dir=<directory>
tcp_port=0
ssl_port=<ssl_port_value>
verify_cn_string=<common_name>
```

Where:

label_agent_private_key

Label of the agent private key that you want to use to communicate. The default is **client**.

file_name

Name of the file, without its extension. The default value is **TWSCClientKeyStore**.

directory

Name of the directory that contains the files generated in [step 1 on page 71](#) and in [step 2 on page 71](#). The default path is `/opt/IBM/TWA_<TWS_user>/TWS/ITA/cpa/ita/cert`.

tcp_port_value

TCP/IP port value. Specify **0**.

ssl_port_value

Same as *tcp_port_value*.

common_name

IBM Z Workload Scheduler checks the validity of the certificate and verifies that the peer certificate has been issued by a recognized CA. If you set the *verify_cn_string* parameter, IBM Z Workload Scheduler verifies that the Common Name (CN) of the Certificate Subject matches the *common_name* that you set in this parameter.

This setting is valid for both dynamic and z-centric agents. To make the changes effective, you must restart the agent.

To configure the TLS v1.2 connection, in the `ita.ini` file add the following properties to the [ITA SSL] section:

```
ssl3_cipher = NONE
tls10_cipher = NONE
tls11_cipher = NONE
tls12_cipher = DFLT
```

4. Stop the agent:

```
ShutDownLwa
```

5. Restart the agent:

```
StartUpLwa
```

After you complete the procedure, depending on the SSL storing certificate method you use, import the certificates in a RACF KEYRING or in a keystore created in the UNIX System Services. Depending on the method you use refer either to the RACF or the Unix System Services documentation.

Enabling FIPS compliance over IBM Z Workload Scheduler server SSL secured connection

Federal Information Processing Standard Security Requirements for Cryptographic Modules, referred to as FIPS 140-2, is a standard published by the National Institute of Standards and Technology (NIST). Organizations can require compliance to the FIPS 140-2 standard to provide protection for sensitive or valuable data to cryptographic-based security systems.

System SSL was designed to meet the Federal Information Processing Standard - FIPS 140-2 Level 1 criteria.

System SSL can run in either "FIPS mode" or "non-FIPS mode". By default, System SSL runs in "non-FIPS" mode.

IBM Z Workload Scheduler uses the System SSL configuration. To run IBM Z Workload Scheduler in "FIPS mode", you must enable FIPS compliance over System SSL connections.

To implement FIPS 140-2 Level 1 standards perform the following actions:

1. Ensure that FIPS-compliance over a SSL connection is enabled on the controller, as described in *IBM z/OS Cryptographic Services System Secure Sockets Layer Programming manual, Chapter 4. System SSL and FIPS 140-2*.
2. On the controller, set ENABLEFIPS to YES in the HTTPOPTS initialization statement. For more information about HTTPOPTS, see *Customization and Tuning*.
3. On the z-centric agent, ensure that the FIPS-compliance is enabled. For more information about how to configure SSL to be FIPS-compliant, see the *IBM Workload Scheduler: Administration Guide*.
4. On the z-centric agent ensure that:
 - SSL is configured, as described in [Customizing the SSL connection with the Z controller when using your certificates \(USS\) on page 71](#)
 - In the `ita.ini` file, the `ssl_port` is set and `fips_enable = 1`.

Encrypting the user password

You define user IDs and passwords for Windows™ users by using the USRREC statement in a PARMLIB member (the default member name is `USRINFO`). For details, see *Customization and Tuning*.

To encrypt the user password, run the sample provided in the EQQBENCR member of the EQQSAMP library.

Chapter 8. Defining and managing jobs

How to define and manage jobs within the z-centric end-to-end scheduling solution.

Defining a job from the Dynamic Workload Console

How to define a new job definition using the Dynamic Workload Console.

About this task

From the Workload Designer, panel of the Dynamic Workload Console you can define multiple types of jobs, running on both distributed and z/OS environments. Some predefined job types are organized into categories including native jobs, and job types with advanced options, which are specific job types that you use to perform operations on external applications. In addition to the existing ones, you can define your own job types and add them to this list. Starting with IBM Z Workload Scheduler version 8.6, it is possible to create JOBLIB members containing z-centric job types using the Dynamic Workload Console. These jobs leverage the job type plug-ins that are included in the z-centric agent Java™ extensions.

These jobs created using the Dynamic Workload Console are saved in XML format in the EQQJBLIB and should not be edited directly. If they need to be modified, changes are to be made by using the Dynamic Workload Console.

Also, if a user ID is provided in the job definition, it is not possible to modify the user using either the IBM Z Workload Scheduler user exit EQQUX001 or the workstation end-to-end options. Only the user specified when the job is defined from the Dynamic Workload Console will be considered when the job runs.

Many of the old plug-ins previously provided with the product are now out-of-the-box integrations available on [Automation Hub](#), together with the related documentation. In addition to these job plug-ins, new integrations to extend your automation processes are constantly added to [Automation Hub](#). For an up-to-date list of all available integrations, visit the [Automation Hub](#) site.

To create a new job definition from the Dynamic Workload Console, perform the following procedure:

1. From the Dynamic Workload Console portfolio, click **Design >Workload Designer**.
2. Specify a z/OS engine name.
3. Click **Create new >Job**.
4. Select the type of job you want to create.
5. In the properties panel, specify the attributes for the job definition you are creating. For all the details about available fields and options, see the online help by clicking the "?" on the top-right corner.
6. Click **Save** to save the job definition in the database.

Defining a job in the JOBLIB data set

You define the jobs to be run on IBM Z Workload Scheduler Agent workstations in a job definition member of the JOBLIB data set. In the JOBLIB member, you can either define the job contents or invoke a script or an executable that is stored locally on the IBM Z Workload Scheduler Agent workstation.

If you do not specify any content in the job definition member, the error code JCLI is issued.

To define the job contents, the following syntax rules apply:

- The lines that start with `/* OPC`, `/*%OPC`, or `/*>OPC` are used for variable substitution and automatic recovery. They are removed before the script is downloaded onto the IBM Z Workload Scheduler Agent.
- Each line starts between column 1 and column 80.
- The last backslash (`\`) in a line is considered the character of continuation.
- Blanks at the end of the line are automatically removed.

To define the job properties, use the [JOBREC on page 75](#) statement. In the [JOBREC on page 75](#) statement, you can define different job types.

JOBREC

To define the properties of a job to be run on IBM Z Workload Scheduler Agent workstations, use the following format:

```
//JOBREC
keyword(value)
keyword(value)
//END JOBREC
```

The following syntax rules apply:

- The value of each keyword can continue over one or more lines.
- Each statement must be on a separate line.
- The value of each keyword must be enclosed within parentheses. This means that every character enclosed within the parentheses is considered to be part of the keyword value, including blanks and single quotes.
- The script or executable must follow the `//END JOBREC` line.
- If you specify the same keyword more than once, only the last keyword specified is considered valid.
- You can insert any comment text by using `/*` as the starting characters.
- Unsupported keywords inserted in the statement are ignored without warning.
- Keywords are optional, unless otherwise indicated.
- Keywords and values are case-sensitive.



Note: When the job is defined using a `JOBREC` and specifying the `JOBTYP` keyword, or when it is defined using the Dynamic Workload Console, any `USERID` specified in the job definition is not used as the user under which the job runs on the z-centric agent. Rather, the specified `USERID` is needed to perform logon to the remote back end processes, such as, database servers or web servers, with which the job must interact.

In contrast, with native z-centric jobs defined directly in the `EQQJBLIB` without any `JOBREC`, or with a `JOBREC` but without the `JOBTYP` keyword, the `USERID` provided by the `JOBUSER` keyword of the `JOBREC`, or by `EQQUX001`, is used by the z-centric agent as the ID under which it submits the job on the server where the agent runs.



JOBREC is supported by z-centric agent (IBM Z Workload Scheduler Agent) on IBM i systems, provided that you set `RunExecutableAsIBMiJobs=true` in the `JobManager.ini` file.

In the JOBREC statement, you can define different job types.

Supported job types and their related keywords are listed in [Table 3: Jobs and their keywords on page 76](#).

Table 3. Jobs and their keywords

Job to be run	Keywords	Section
database	DBNAME (required) DBTYPE (required) JOBTYP (required) PORT (required) SERVER (required) STATEMENT(<i>id</i>) (required) CMDTYPE(<i>id</i>) (optional) JOBPWD (optional) JOBUSR (optional) POLLINGIVL(<i>id</i>) (optional, applies only to the MSSQL database type) SYNCTYPE(<i>id</i>) (optional, applies only to the MSSQL database type) WSNAME (optional)	Syntax diagram for database job types on page 77
file transfer	JOBPWD (required) JOBTYP (required) JOBUSR (required) LOCALFILE (required) TRANSFERTYPE (required) APPENDTEXT (optional) DELETEAFTERDOWNLOAD (optional) DELETEAFTERUPLOAD (optional) EXTRACOMMANDS (optional) LOCALCODEPAGE (optional) LOCALPWD (optional) LOCALUSR (optional) MAXPORT (optional) MINPORT (optional) PASSIVEMODE (optional) PROTOCOL (optional) REMOTECODEPAGE (optional) REMOTEFIL (required)	Syntax diagram for file transfer job types on page 85

Table 3. Jobs and their keywords (continued)

Job to be run	Keywords	Section
	SERVER (required) TIMEOUT (optional) TRANSFERMODE (optional)	
web service	JOBTYPE (required) OPNAME (required) URL (required) JOBPWD (optional) JOBUSR (optional) PARM(<i>id</i>) (optional) WSNAME (optional)	Syntax diagram for web service job types on page 91
java	CLASSNAME (required) JOBTYPE (required) JARPATH (optional) JAVAPARM(<i>id</i>) (optional)	Syntax diagram for Java job type on page 93
xajob	JOBTYPE (required) ENVVAR(<i>id</i>) (optional) JOBPWD (optional) JOBUSR (optional) WSNAME (optional)	Syntax diagram for xajob job type on page 95
j2ee jms	CONNFACTORY (required) DESTINATION (required) JOBTYPE (required) AUTHALIAS (optional) INVOKEYTYPE (optional) JOBPWD (optional) JOBUSR (optional) MESSAGE (optional) WSNAME (optional)	Syntax diagram for J2EE jms job type on page 97

Syntax diagram for database job types

The database job type:

If you set JOBTYPE to (/database), you can choose among the following database types:

- DB2®
- Oracle
- MSSQL

The syntax for DB2® and Oracle is identical, but the syntax for MSSQL supports MSSQL-specific operations. For details, see [Syntax diagram for DB2 and Oracle database job types on page 78](#) and [Syntax diagram for MSSQL database job type on page 81](#).

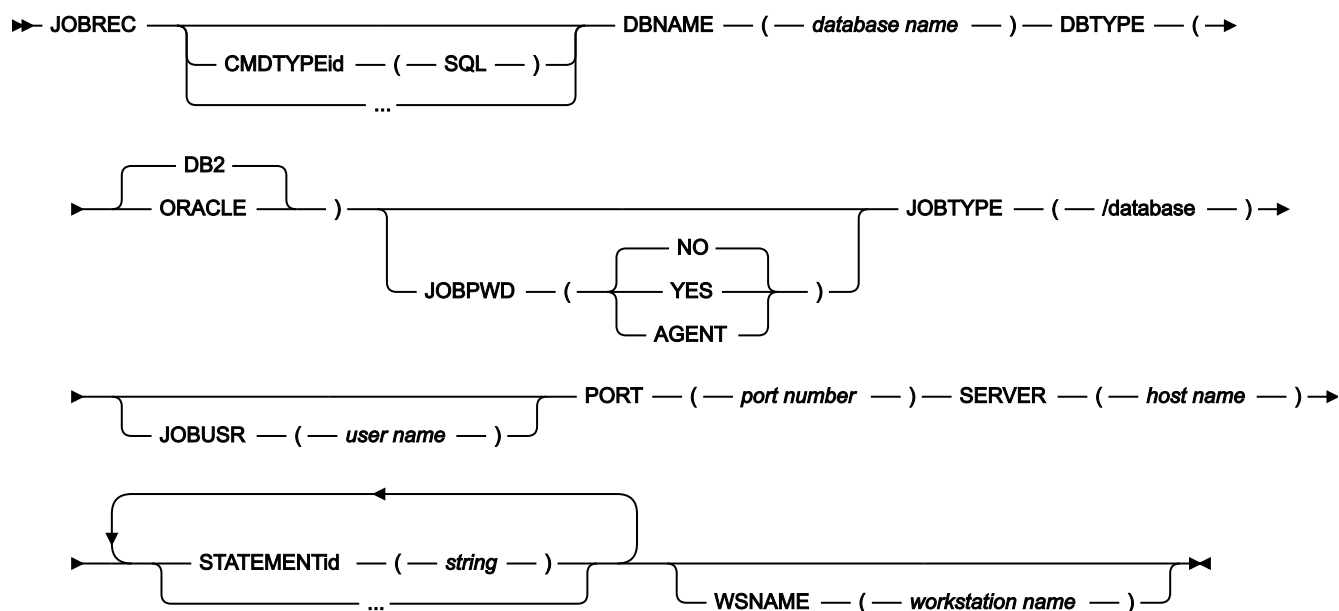
Syntax diagram for DB2® and Oracle database job types

Purpose

Syntax diagram for the DB2® and Oracle database job types. The following syntax rules apply:

- You can define multiple statements in the same job by specifying a series of `STATEMENTid` keywords, where *id* is a unique numeric value, including null and zero. For example, you can define `STATEMENT`, `STATEMENT0`, `STATEMENT1`, `STATEMENT2`, and so on.
- If you set `DBTYPE` to `DB2` or `Oracle`, you can set `CMDTYPE` to `SQL`.
- You can specify any number of `CMDTYPE` keywords, according to the corresponding `STATEMENT` keyword. The `STATEMENT` keyword must be specified before the corresponding `CMDTYPE` keyword. For example, you can specify `CMDTYPE3` only if you have previously specified `STATEMENT3`.

Format



Parameters

CMDTYPE*id*(SQL)

The type of command to be run. You can specify a series of `CMDTYPE $id$` keywords, where id is a unique numeric value, including null and zero. The id of the `CMDTYPE` keyword must match the id of the related `STATEMENT` keyword. For example, you can specify `CMDTYPE3` only if you specify `STATEMENT3`.

If you set DBTYPE to (/DB2) or to (/ORACLE), you can set CMDTYPE to SQL.

DBNAME(database name)

The name of the database. This keyword is required.

DBTYPE(DB2|ORACLE)

The database type. This keyword is required.

JOBPWD(YES|NO|AGENT)

Specifies if the user name entered in JOBUSR or set by using the job-submit exit EQQUX001 is associated with a password.

If you set JOBPWD to **YES**, IBM Z Workload Scheduler searches for the user password in the USRPSW keyword of the USRREC statement (for details, see *IBM Z Workload Scheduler: Customization and Tuning*).

If you set JOBPWD to **AGENT**, this means that the password is resolved locally on the agent. The password must have been defined on the agent by means of the [param on page 49](#) command. This feature is available independently of the operating system of the workstation.

Typically, the password is required for users who schedule jobs to run on Windows™ workstations. Set JOBPWD to **NO** if the user works with UNIX™ workstations and if no password is required.

JOBTYP(/xajob/access method name|/j2ee/jms|/web service|/file transfer|/database|/java)

To run a database job, specify **/database**. For information about the other types of job, see the specific section for each job type. This keyword is required.

/database

Set this value to submit a database job. Before you can run a query on a database, download the JDBC drivers for your database client to each IBM Z Workload Scheduler Agent on which you want to submit database jobs. Specify the path to the database client jar files in the `DatabaseJobExecutor.properties` file, located in the `JavaExt\cfg` directory in your IBM Workload Scheduler installation directory. Define the `jdbcdDriversPath` property to point to the JDBC jar files directory, for example, `jdbcdDriversPath=c:\\mydir\\jars\\jdbc`. The JDBC jar files must be located in the specified directory or its subdirectories. Ensure that you have list permissions for the directory and its subdirectories.

JOBUSR(user name)

The user name for accessing the database.

If the user schedules jobs to run on Windows™ workstations, ensure that a user password is also defined (see the JOBPWD keyword).

- If you are defining a Windows™ domain user, use the following format:

```
JOBUSR(domainName\user1)
```

- If you are defining a Windows™ user in the `username@internet_domain` format, use the following format:

```
JOBUSR('administrator@mywindom.com')
```

To specify the user name, you can also use the job-submit exit EQQUX001. This user name overrides the value specified for JOBUSR. In turn, the value specified for JOBUSR overrides the value specified for USRNAM in the USRREC statement.

PORT(port number)

The port number for the database job. This keyword is required.

SERVER(host name)

The host name of the server where the file transfer is to be performed or where the database is located. This keyword is required if you set JOBTYP to (/file transfer) or to (/database).

STATEMENTid(string)

The string defining the SQL query or job. This keyword is required. You can define multiple statements in the same job by specifying a series of STATEMENT id keywords, where id is a unique numeric value, including null and zero. For example, you can define STATEMENT, STATEMENT0, STATEMENT1, STATEMENT2, and so on.

When running stored procedures, ensure that the returned result is in tabular format. Results in any other format are not supported.

WSNAME(workstation_name)

The name of the workstation from which user name and password must be retrieved. User name and password can be specified in the statement, using the JOBUSR and JOBPWD keywords, or can be associated to a workstation with the USRREC statement. With the WSNAME keyword, you can specify a workstation where the user name and password are stored, which is different from the workstation where the job runs.

Example

Examples

The following is an example of a JOBTYP statement to select all records from table SYSTOOLS.POLICY:

```
//JOBTYP
JOBTYP(/database)
STATEMENT20(SELECT * FROM SYSTOOLS.POLICY)
DBNAME(DBWEB)
DBTYPE(DB2)
SERVER(9.168.99.87)
PORT(50000)
JOBUSR(Administrator)
JOBPWD(YES)
CMDTYPE20(SQL)
//END JOBTYP
```

The following is an example of a JOBTYP statement to select columns country_id and country_name FROM table countries:

```
//JOBTYP
JOBTYP(/database)
STATEMENT1(SELECT country_id, country_name FROM countries)
DBNAME(SAMPLE)
DBTYPE(ORACLE)
SERVER(9.168.115.37)
```



```
PORT(70000)
CMDTYPE1(SQL)
JOBUSR(ZCENTUMB)
JOBPWD(YES)
//END JOBREC
```

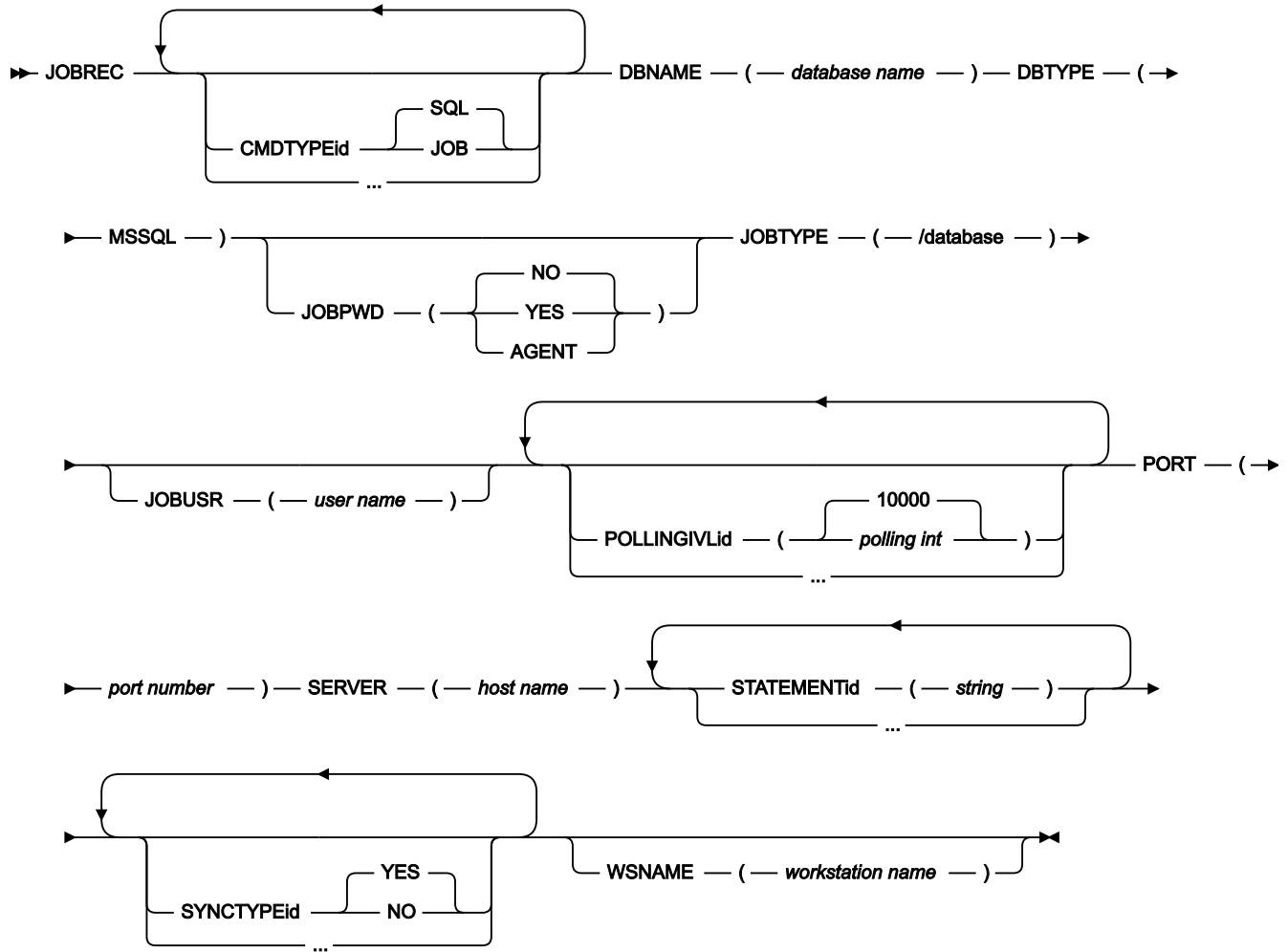
Syntax diagram for MSSQL database job type

Purpose

This section provides detailed information about the syntax diagram for the MSSQL database job type. The following syntax rules apply:

- You can define multiple statements in the same job by specifying a series of `STATEMENT $id$` keywords, where id is a unique numeric value, including null and zero. For example, you can define `STATEMENT`, `STATEMENT0`, `STATEMENT1`, `STATEMENT2`, and so on.
- If you set `DBTYPE` to `(/MSSQL)`, you can set `CMDTYPE` to either `JOB` or `SQL` and specify the related job or SQL query in the `STATEMENT` keyword. If you set `CMDTYPE` to `JOB`, you can specify a job saved on the server; if you set `CMDTYPE` to `SQL`, you can specify a statement in the native language of the database that you selected.
- If you set `CMDTYPE` to `JOB`, you can optionally define the `SYNCTYPE` and `POLLINGIVL` keywords. This information is not visible in the syntax diagram.
- If you set `CMDTYPE` to `SQL`, you cannot define the `SYNCTYPE` and `POLLINGIVL` keywords. This information is not visible in the syntax diagram.
- You can specify any number of `CMDTYPE`, `SYNCTYPE`, and `POLLINGIVL` keywords, according to the corresponding `STATEMENT` keyword. The `STATEMENT` keyword must be specified before the corresponding `CMDTYPE`, `SYNCTYPE`, and `POLLINGIVL` keywords. For example, you can specify `CMDTYPE3`, `SYNCTYPE3`, and `POLLINGIVL3` only if you have previously specified `STATEMENT3`.

Format



Parameters

CMDTYPEid(SQL|JOB)

The type of command to be run. You can specify a series of CMDTYPEid keywords, where *id* is a unique numeric value, including null and zero. The *id* of the CMDTYPE keyword must match the *id* of the related STATEMENT keyword. For example, you can specify CMDTYPE3 only if you specify STATEMENT3.

If you set DBTYPE to (/MSSQL), you can set CMDTYPE to either JOB or SQL and specify the related job or SQL query in the STATEMENT keyword. If you set CMDTYPE to JOB, you can specify a job saved on the server; if you set CMDTYPE to SQL, you can specify a statement in the native language of the database that you selected.

DBNAME(database name)

The name of the database. This keyword is required.

DBTYPE(MSSQL)

The database type. This keyword is required.

JOBPWD(YES|NO|AGENT)

Specifies if the user name entered in JOBUSR or set by using the job-submit exit EQQUX001 is associated with a password.

If you set JOBPWD to **YES**, IBM Z Workload Scheduler searches for the user password in the USRPSW keyword of the USRREC statement (for details, see *IBM Z Workload Scheduler: Customization and Tuning*).

If you set JOBPWD to **AGENT**, this means that the password is resolved locally on the agent. The password must have been defined on the agent by means of the [param on page 49](#) command. This feature is available independently of the operating system of the workstation.

Typically, the password is required for users who schedule jobs to run on Windows™ workstations. Set JOBPWD to **NO** if the user works with UNIX™ workstations and if no password is required.

JOBTYPE(/xajob/access method name|/j2ee/jms|/web service|/file transfer|/database|/java)

To run a database job, specify **/database**. For information about the other types of job, see the specific section for each job type. This keyword is required.

/database

Set this value to submit a database job. Before you can run a query on a database, download the JDBC drivers for your database client to each IBM Z Workload Scheduler Agent on which you want to submit database jobs. Specify the path to the database client jar files in the `DatabaseJobExecutor.properties` file, located in the `JavaExt\cfg` directory in your IBM Workload Scheduler installation directory. Define the `jdbcDriversPath` property to point to the JDBC jar files directory, for example, `jdbcDriversPath=c:\mydir\jars\jdbc`. The JDBC jar files must be located in the specified directory or its subdirectories. Ensure that you have list permissions for the directory and its subdirectories.

JOBUSR(user name)

The user name for accessing the database.

If the user schedules jobs to run on Windows™ workstations, ensure that a user password is also defined (see the JOBPWD keyword).

- If you are defining a Windows™ domain user, use the following format:

```
JOBUSR(domainName\user1)
```

- If you are defining a Windows™ user in the `username@internet_domain` format, use the following format:

```
JOBUSR('administrator@mywindow.com')
```

To specify the user name, you can also use the job-submit exit EQQUX001. This user name overrides the value specified for JOBUSR. In turn, the value specified for JOBUSR overrides the value specified for USRNAM in the USRREC statement.

POLLINGIVLid(polling int|10000)

The polling interval in milliseconds. This keyword is valid only if you set CMDTYPE to (/JOB). You can specify a series of POLLINGIVLid keywords, where *id* is a unique numeric value, including null and zero. The *id* of the POLLINGIVL keyword must match the *id* of the related STATEMENT keyword. For example, you can specify POLLINGIVL3 only if you specify STATEMENT3.

PORT(port number)

The port number for the database job. This keyword is required.

SERVER(host name)

The host name of the server where the file transfer is to be performed or where the database is located. This keyword is required if you set JOBTYP to (/file transfer) or to (/database).

STATEMENTid(string)

The string defining the SQL query or job. This keyword is required. You can define multiple statements in the same job by specifying a series of STATEMENTid keywords, where *id* is a unique numeric value, including null and zero. For example, you can define STATEMENT, STATEMENT0, STATEMENT1, STATEMENT2, and so on.

When running stored procedures, ensure that the returned result is in tabular format. Results in any other format are not supported.

SYNCTYPEid(YES|NO)

Specifies if synchronization is required. This keyword is valid only if you set CMDTYPE to (/JOB). You can specify a series of SYNCTYPEid keywords, where *id* is a unique numeric value, including null and zero. The *id* of the SYNCTYPE keyword must match the *id* of the related STATEMENT keyword. For example, you can specify SYNCTYPE3 only if you specify STATEMENT3.

WSNAME(workstation_name)

The name of the workstation from which user name and password must be retrieved. User name and password can be specified in the statement, using the JOBUSR and JOBPWD keywords, or can be associated to a workstation with the USRREC statement. With the WSNAME keyword, you can specify a workstation where the user name and password are stored, which is different from the workstation where the job runs.

Example**Examples**

The following is an example of a JOBTYP statement to run job SPMTEST_BACKUP without synchronization and with a polling interval of 100 milliseconds, and to select all records from the DBO.TWSRECORDS table:

```
//JOBTYP
JOBTYP(/database)
DBNAME(SCVTEST)
DBTYPE(MSSQL)
SERVER(SQLCON008)
PORT(1278)
STATEMENT1(SPMTEST_BACKUP)
CMDTYPE1(JOB)
```

```
SYNCTYPE1(NO)
POLLINGIVL1(100)
STATEMENT2(SELECT * FROM DBO.TWSRECORDS)
CMDTYPE2(SQL)
JOBUSR(PWR_USER)
JOBPWD(YES)
//END JOBREC
```

Syntax diagram for file transfer job types

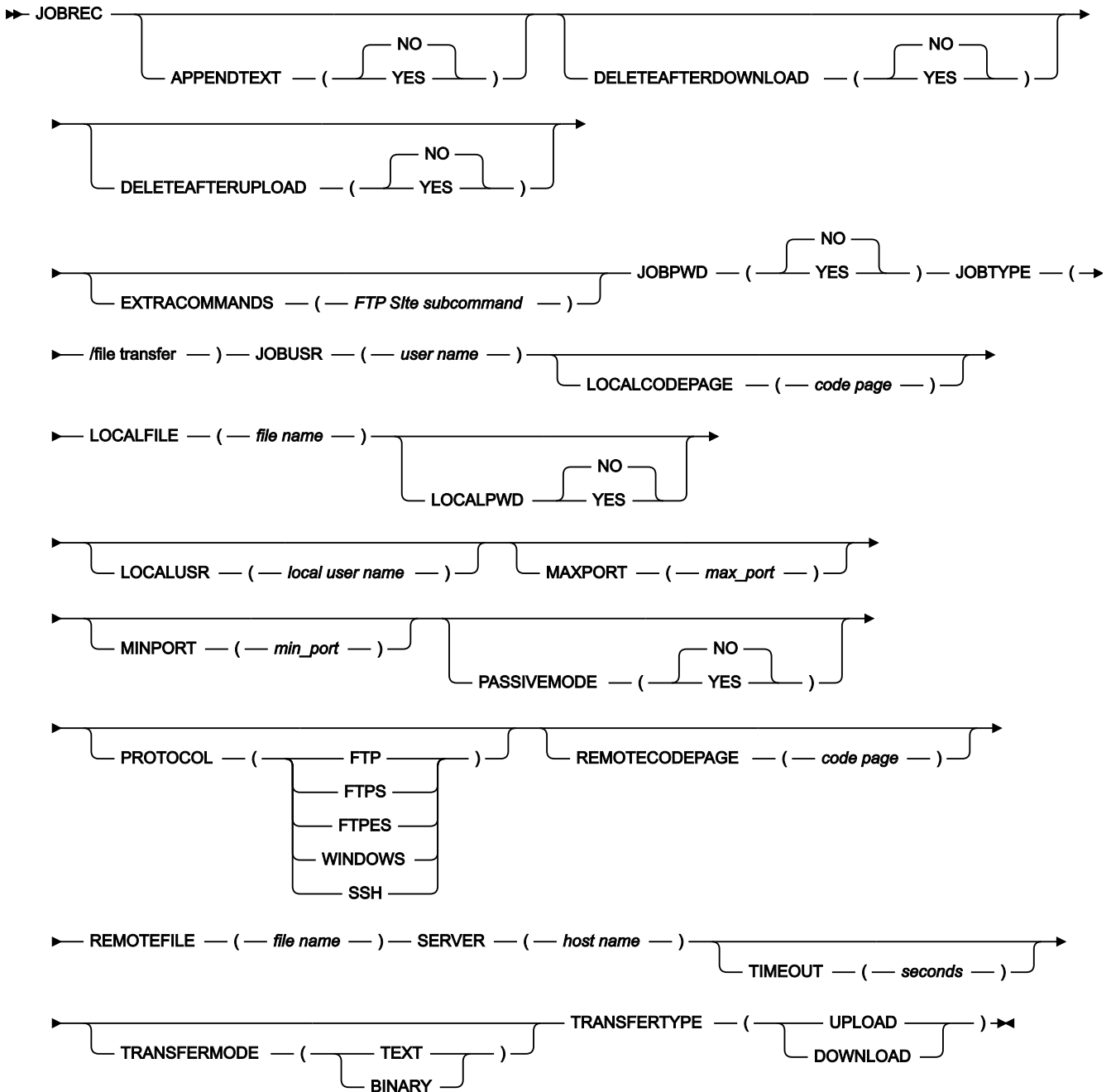
Purpose

If you set JOBTYP to `(/file transfer)`, you can choose between the following transfer types for transferring binary and text files:

- Upload: Sends a file to a server.
- Download: Downloads a file from a server.

This section provides detailed information about the syntax diagram for both the upload and download transfer types. The following syntax rules apply:

- The syntax of the selected protocol applies.

Format**Parameters**

The MINPORT and the MAXPORT options define the port range to use on the client side of TCP data connections. When the active mode is enabled, the MINPORT and the MAXPORT options restrict the port numbers sent by the PORT command. These options are meant to accommodate highly restrictive firewall rules.

If you do not specify the MINPORT and the MAXPORT options, the operating system determines the port numbers to be used. If you specify one of these parameters, the other parameter is required.

APPENDTEXT(YES|NO)

Specify this attribute and set it to YES to append the data contained in the file being transferred to the end of the file on the destination system, if it already exists. If the file does not already exist, it is created. This attribute applies to file transfers of type download and when TRANSFERMODE is set to TEXT. Omit this attribute if you do not want to append the data to an existing file. The default value is NO.

DELETEAFTERDOWNLOAD(YES|NO)

Set this attribute to YES to delete the transferred files from the server after the download. The default value is NO.

DELETEAFTERUPLOAD(YES|NO)

Set this attribute to YES if the source files must be deleted after the transfer. The default value is NO.

EXTRACOMMANDS(FTP Site subcommand)

Specify a Site subcommand to send information that is used by the remote host to provide services specific to that host system. Valid for an upload file transfer using the FTP protocol when the target workstation is a z/OS system. For example, you can specify a subcommand to define how to allocate the data set (LR,BLK, VOL, to name a few).

JOBPWD(YES|NO)

Specifies that the user name set in JOBUSR or by using the job-submit exit EQQUX001 is associated with a password.

If you set JOBPWD to YES, IBM Z Workload Scheduler searches for the user password in the USRPSW keyword of the USRREC statement (for details, see [USRREC on page 46](#)).

Typically, the password is required for users who schedule jobs to run on Windows™ workstations. Set JOBPWD to NO if the user works with UNIX™ workstations.

JOBTYPE(/xajob/access method name|/j2ee/ejb|/j2ee/jms|/web service|/file transfer|/database|/java)

To run a file transfer job, specify **/file transfer**. For information about the other types of job, see the specific section for each job type. This keyword is required.

/file transfer

Set this value to submit a file transfer job.

JOBUSR(user name)

The user name for accessing the FTP, FTPS, FTPES, SSH, or Windows™ server.

If the user schedules jobs to run on Windows™ workstations, ensure that a user password is also defined (see the JOBPWD keyword). Use the following syntax:

To define a Windows™ domain user

```
JOBUSR (domainName\userName)
```

To define a Windows™ domain user in the *username@web_domain* format

```
JOBUSR ( 'userName@web_domain' )
```

To specify the user name, you can also use the job-submit exit EQQUX001. This user name overrides the value specified for JOBUSR. In turn, the value specified for JOBUSR overrides the value specified for USRNAM in the USRREC statement.

LOCALCODEPAGE(*code page*)

The name of the code page used on the local system. This keyword is required only when transferring text files from a distributed to a z/OS environment and vice versa. If you specify this parameter, the REMOTECODEPAGE parameter is required.

LOCALFILE(*file name*)

Required. The name of the local file that you want to transfer. When uploading, this is the source file, when downloading, this is the target file.

LOCALPWD(*YES|NO*)

Specifies that the user name set for LOCALUSR is associated with a password. Typically, the password is required for users who schedule jobs to run on Windows workstations. LOCALPWD must be set to YES also if TRANSFERTYPE is set to UPLOAD and the file transfer is towards a z/OS server.

By setting LOCALPWD to YES, IBM® Z Workload Scheduler searches for the user password in the USRPSW parameter of the USRREC statement (for details, see [USRREC on page 46](#)).

LOCALUSR(*local user name*)

Specifies the local user, if the scheduled jobs run on Windows workstations. The value specified for LOCALUSR overrides the value specified for USRNAM in the USRREC statement. If you set LOCALUSR, the LOCALPWD parameter must also be set to YES.

Use the following syntax:

To define a Windows domain user

```
LOCALUSR(domainName\userName)
```

To define a Windows user in the *username@web_domain* format

```
LOCALUSR( 'userName@web_domain' )
```

MAXPORT(*max_port*)

The maximum port value to use on the client side of TCP data connections. For example, if you specify MAXPORT=8009, IBM Z Workload Scheduler restricts the port number to be lesser than or equal to the port 8009. The default value is 1035.

MINPORT(*min_port*)

The minimum port value to use on the client side of TCP data connections. For example, if you specify MINPORT=8000, IBM Z Workload Scheduler restricts the port number to be greater than or equal to the port 8000.

PASSIVEMODE(YES|NO)

Specifies whether the server is passive or active in establishing connections for data transfers. If you set this option to NO, the server establishes the data connection with the client (active mode), if you set this option to YES, the client establishes the data connection with the server (passive mode). The default value is NO.

PROTOCOL(FTP|FTPS|FTPES|WINDOWS|SSH)

The protocol to be used for the file transfer. Allowed values are:

FTP

A standard network protocol used to exchange files over a TCP/IP-based network, such as the Internet.

FTPS

An extension to the File Transfer Protocol (FTP) that adds support for the Transport Layer Security (TLS) and the Secure Sockets Layer (SSL) cryptographic protocols. Specifically, the supported protocol is TLS security for FTP sessions with **implicit** secure TLS login and with private security level for the data connection.

FTPES

An extension to the File Transfer Protocol (FTP). Specifically, the supported protocol is TLS security for FTP sessions with **explicit** TLS security and private security level for the data connection.

WINDOWS

The Microsoft™ file sharing protocol. Specify the root shared directory in the REMOTEFIL keyword. Specify the address of the workstation hosting the shared directory in the SERVER keyword.

SSH

A network protocol that provides file access, file transfer, and file management functions over any data stream.

REMOTECODEPAGE(*file name*)

The name of the code page used on the remote system. This keyword is required only when transferring text files from a distributed to a z/OS environment and vice versa. If you specify this parameter, the LOCALCODEPAGE parameter is required. If you want to use a custom code page, define the REMOTECODEPAGE parameter as follows:

```
REMOTECODEPAGE(USER:custom_code_page)
```

where *custom_codepage* is the code page defined by the user.

For example, to use the tcpip.ftpd.ftpxlbin.frence3 custom code page, define the REMOTECODEPAGE parameter as follows:

```
REMOTECODEPAGE(USER:tcpip.ftpd.ftpxlbin.frence3)
```

REMOTEFILE(*file name*)

The name of the remote file that you want to transfer. When uploading, this is the target file, when downloading, this is the source file. This keyword is required.

SERVER(*host name*)

The host name of the server where the file transfer is to be performed or where the database is located. This keyword is required if you set JOBTYP to (/file transfer) or (/database).

TIMEOUT(*seconds*)

Specifies the number of seconds to be used for the file transfer operation. The default value is 60 seconds.

TRANSFERMODE(TEXT|BINARY)

The mode of file transfer that you want to perform, either text or binary.

TRANSFERTYPE(UPLOAD|DOWNLOAD)

The type of file transfer that you want to perform; either upload or download. This keyword is required.

Example**Examples**

The following is an example of a JOBTYP statement to transfer a file using the FTP protocol:

```
//JOBTYP
JOBTYP(/file transfer)
TRANSFERTYPE(DOWNLOAD)
SERVER(server address)
LOCALFILE(C:\file1.txt)
REMOTEFILE(/file1.txt)
PROTOCOL(FTP)
JOBTYP(username)
JOBTYP(YES)
//END JOBTYP
```

The following is an example of a JOBTYP statement to transfer a file using the SSH protocol:

```
//JOBTYP
JOBTYP(/file transfer)
TRANSFERTYPE(DOWNLOAD)
SERVER(server address)
LOCALFILE(C:\file1.txt)
REMOTEFILE(/home/provaftp/file1.txt)
PROTOCOL(SSH)
JOBTYP(username)
JOBTYP(YES)
//END JOBTYP
```

The following is an example of a JOBTYP statement to transfer a file using the WINDOWS protocol:

```
//JOBTYP
JOBTYP(/file transfer)
TRANSFERTYPE(UPLOAD)
SERVER(server address)
LOCALFILE(C:\file3.txt)
```

```

REMOTEFILE(shared directory\file88.txt)
PROTOCOL(WINDOWS)
JOBUSR(username)
JOBPWD(YES)
//END JOBREC

```

The following example shows a JOBREC statement to transfer a native z/OS file from a z/OS workstation to a Windows workstation:

```

//JOBREC
JOBTYPE(/file transfer)
TRANSFERTYPE(DOWNLOAD)
TRANSFERMODE(TEXT)
SERVER(9.168.101.41)
LOCALFILE(d:\MyFTPRemoteTextFile.txt)
REMOTEFILE('TWSZ.REMOTE.TEXTFILE')
LOCALCODEPAGE(IS08859-1)
REMOTECODEPAGE(IBM-037)
JOBUSR(root)
JOBPWD(YES)
PROTOCOL(FTP)
//END JOBREC

```

The following example shows a passive mode transfer with a timeout of 10 seconds. The system determines which port to use between 1034 and 1035:

```

//JOBREC
JOBTYPE(/file transfer)
TRANSFERTYPE(DOWNLOAD)
TRANSFERMODE(TEXT)
SERVER(9.168.107.126)
LOCALFILE(d:\MyFTPRemoteTextFile11.txt)
REMOTEFILE('TWS851.REMOTE.TEXTFILE')
LOCALCODEPAGE(IS08859-1)
REMOTECODEPAGE(IBM-037)
TIMEOUT(10)
MINPORT(1034)
MAXPORT(1035)
PASSIVEMODE(YES)
JOBUSR(LNDN_USER)
JOBPWD(YES)
PROTOCOL(FTPES)
//END JOBREC

```

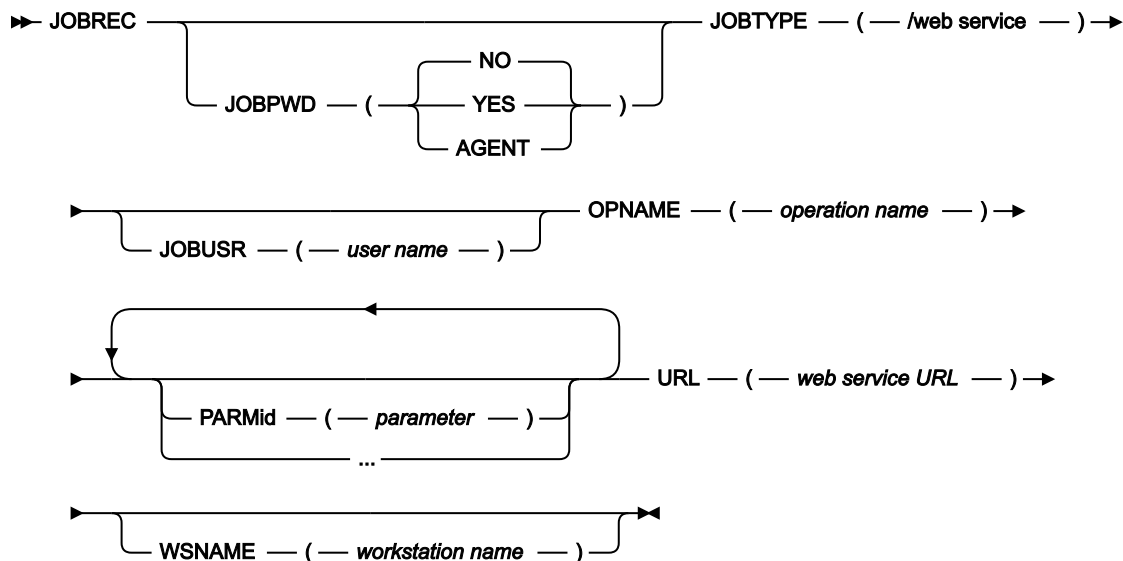
Syntax diagram for web service job types

Purpose

If you set JOBTYPE to (/web service), you can run an existing web service:

This section provides detailed information about the syntax diagram for the web service job type.

Format



Parameters

JOBPWD(YES|NO|AGENT)

Specifies if the user name entered in JOBUSR or set by using the job-submit exit EQQUX001 is associated with a password.

If you set JOBPWD to **YES**, IBM Z Workload Scheduler searches for the user password in the USRPSW keyword of the USRREC statement (for details, see *IBM Z Workload Scheduler: Customization and Tuning*).

If you set JOBPWD to **AGENT**, this means that the password is resolved locally on the agent. The password must have been defined on the agent by means of the [param on page 49](#) command. This feature is available independently of the operating system of the workstation.

Typically, the password is required for users who schedule jobs to run on Windows™ workstations. Set JOBPWD to **NO** if the user works with UNIX™ workstations and if no password is required.

JOBTYP(/xajob/access method name|/j2ee/jms|/web service|/file transfer|/database|/java)

To run a web service job, specify **/web service**. For information about the other types of job, see the specific section for each job type. This keyword is required.

/web service

Set this value to submit a web service job.

JOBUSR(*user name*)

The user name for accessing the web service, if required.

If the user schedules jobs to run on Windows™ workstations, ensure that a user password is also defined (see the JOBPWD keyword).

- If you are defining a Windows™ domain user, use the following format:

```
JOBUSR(domainName\user1)
```

- If you are defining a Windows™ user in the *username@internet_domain* format, use the following format:

```
JOBUSR('administrator@mywindow.com')
```

To specify the user name, you can also use the job-submit exit EQQUX001. This user name overrides the value specified for JOBUSR. In turn, the value specified for JOBUSR overrides the value specified for USRNAME in the USRREC statement.

OPNAME(operation name)

The name of the operation to run. This keyword is required.

PARMid(parameter)

The parameter to be used when running the script or command. You can specify a series of PARM*id* keywords, where *id* is a unique numeric value, including null and zero.

URL(web service URL)

The URL of the web service to be called. This keyword is required.

WSNAME(workstation_name)

The name of the workstation from which user name and password must be retrieved. User name and password can be specified in the statement, using the JOBUSR and JOBPWD keywords, or can be associated to a workstation with the USRREC statement. With the WSNAME keyword, you can specify a workstation where the user name and password are stored, which is different from the workstation where the job runs.

Example

Examples

The following is an example of a JOBRECE statement to call a web service that calculates the sum of the specified parameters:

```
//JOBREC
JOBTYP(/web service)
PARM1(10)
PARM2(20)
JOBUSR(administrator)
JOBPWD(YES)
OPNAME(getSum)
URL(http://manchester.testlab.uk.service.com:9467/Sum/services/Sum/wsd/Sum.wsd)
//END JOBRECE
```

Syntax diagram for Java job type

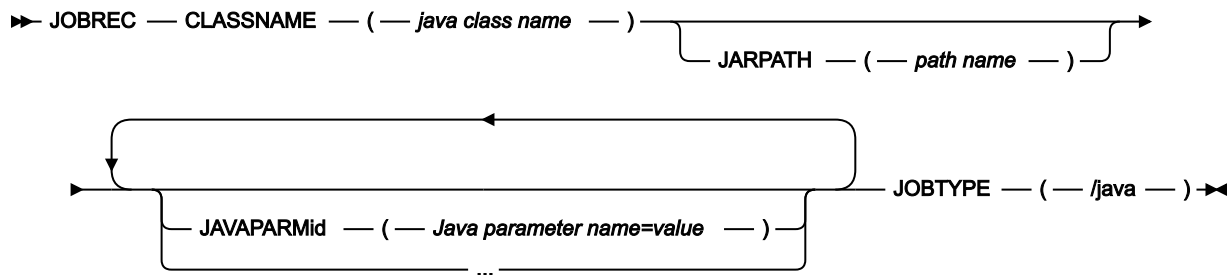
Purpose

If you set JOBTYP to (/java), you can run an existing Java class:

This section provides detailed information about the syntax diagram for the Java job type. The following syntax rules apply:

- Use the UNIX format for the value of the JARPATH keyword.

Format



Parameters

CLASSNAME(*java class name*)

The name of the Java class to run. This keyword is required.

JARPATH(*path name*)

The path to the jar file.

JAVAPARMid(*Java parameter name=value*)

The Java parameter used by the Java class. You can define multiple parameters in the same job by specifying a series of JAVAPARMid keywords, where *id* is a unique numeric value, including null and zero. For example, you can define JAVAPARM1, JAVAPARM2, and so on.

Use the following format, which is case sensitive:

```
JAVAPARMid(Java parameter name=value)
```

Where:

Java parameter name

The name of the Java parameter.

value

The value associated with the Java parameter.

JOBTYPE(/xajob/*access method name*/j2ee/jms/web service/file transfer/database/*java*)

To run a Java job, specify **/java**. For information about the other types of job, see the specific section for each job type.

/java

Set this value to submit a Java job. Ensure that you have list permissions for the folder specified in the jarPath property.

Example

Examples

The following is an example of a JOBRREC statement to run a class with name mypackage.base:

```
//JOBRREC
JOBTYPED(/java)
CLASSNAME(mypackage.base)
JAVAPARM1(name=backup)
JARPATH(c:/jarPath2)
//END JOBRREC
```

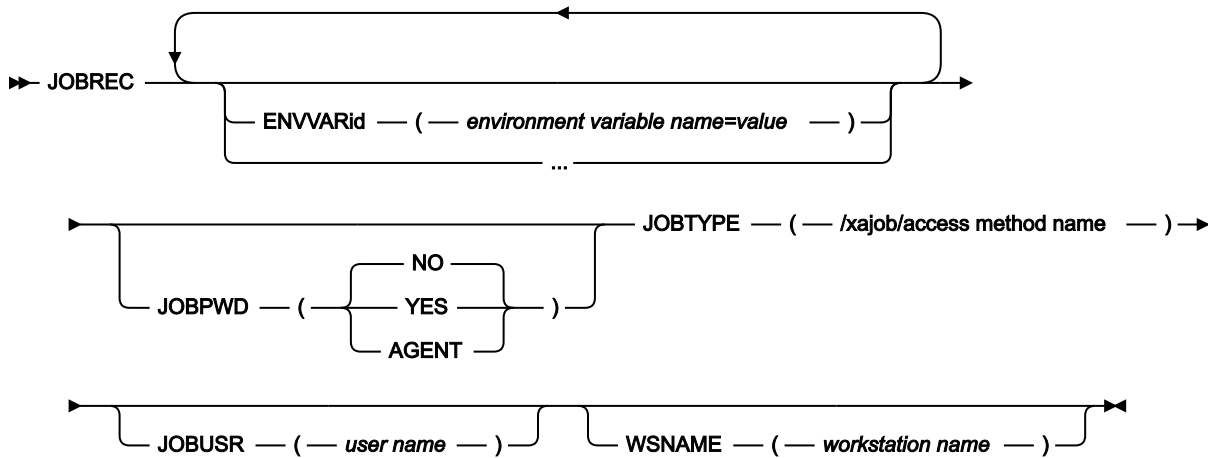
Syntax diagram for xajob job type

Purpose

If you set JOBTYPED to /xajob/access *method name*, you can run a job on an extended agent:

This section provides detailed information about the syntax diagram for the xajob job type.

Format



Parameters

ENVVARid(environment variable name=value)

The environment variable that the task launcher must set before submitting the job. You can specify a series of ENVVARid keywords, where *id* is a unique numeric value, including null and zero.

Use the following format, which is case sensitive:

```
ENVVARid(environment variable name=value)
```

Where:

environment variable name

The name of the environment variable.

value

The value associated with the environment variable.

JOBPWD(YES|NO|AGENT)

Specifies if the user name entered in JOBUSR or set by using the job-submit exit EQQUX001 is associated with a password.

If you set JOBPWD to **YES**, IBM Z Workload Scheduler searches for the user password in the USRPSW keyword of the USRREC statement (for details, see *IBM Z Workload Scheduler: Customization and Tuning*).

If you set JOBPWD to **AGENT**, this means that the password is resolved locally on the agent. The password must have been defined on the agent by means of the [param on page 49](#) command. This feature is available independently of the operating system of the workstation.

Typically, the password is required for users who schedule jobs to run on Windows™ workstations. Set JOBPWD to **NO** if the user works with UNIX™ workstations and if no password is required.

JOBTYPE(/xajob/access method name|/j2ee/jms|/web service|/file transfer|/database|/java)

To run an xajob, specify **/xajob/access method name**. For information about the other types of job, see the specific section for each job type. This keyword is required.

/xajob/access method name

The name of the access method hosted by the extended agent on which you want to submit the job. For example, for a job to be submitted on a SAP extended agent, this value might be `/xajob/r3batch`. If you set this keyword, the JOBCMD keyword value is ignored.

JOBUSR(user name)

The name of the user submitting the job.

If the user schedules jobs to run on Windows™ workstations, ensure that a user password is also defined (see the JOBPWD keyword).

- If you are defining a Windows™ domain user, use the following format:

```
JOBUSR(domainName\user1)
```

- If you are defining a Windows™ user in the *username@internet_domain* format, use the following format:

```
JOBUSR('administrator@mywindow.com')
```

To specify the user name, you can also use the job-submit exit EQQUX001. This user name overrides the value specified for JOBUSR. In turn, the value specified for JOBUSR overrides the value specified for USRNAM in the USRREC statement.

WSNAME(workstation_name)

The name of the workstation from which user name and password must be retrieved. User name and password can be specified in the statement, using the JOBUSR and JOBPWD keywords, or can be associated to a

workstation with the USRREC statement. With the WSNAME keyword, you can specify a workstation where the user name and password are stored, which is different from the workstation where the job runs.

Example

Examples

The following is an example of a JOBRECE statement to un a job using the r3batch access method. The content of the JOBTYPK keyword is case sensitive:

```
//JOBREC
JOBTYPK(/xajob/r3batch)
ENVVAR10(VAL=10)
//END JOBREC
-JOB SAPJOB -USER SAPUSER
```

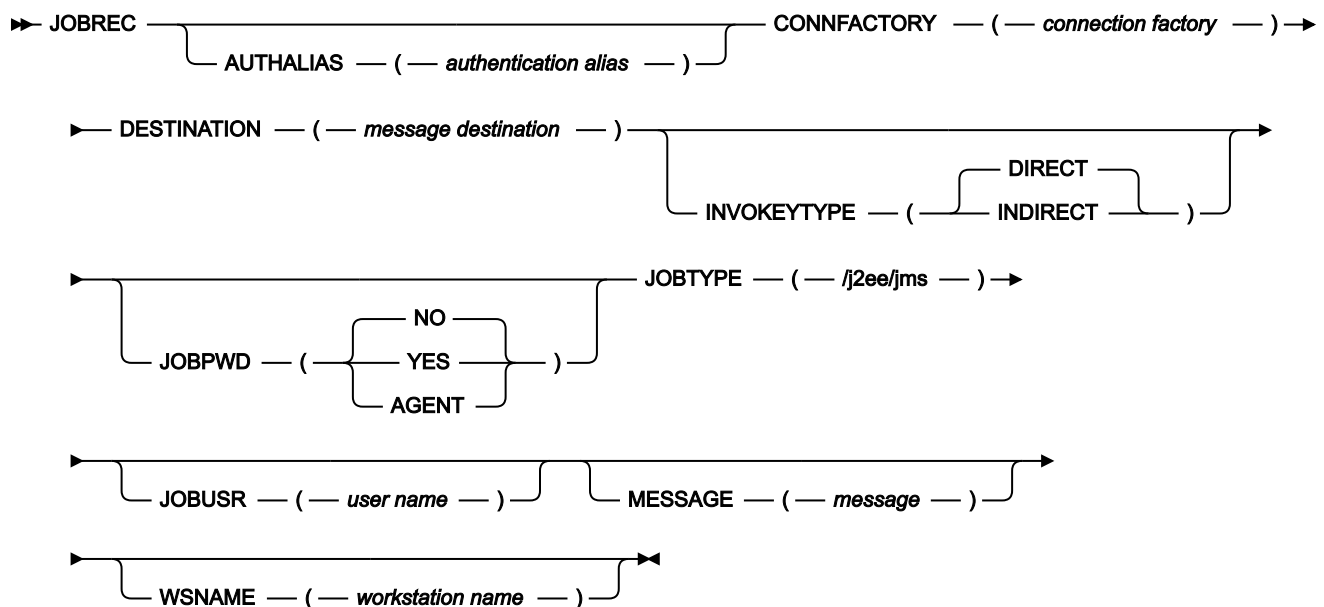
Syntax diagram for J2EE jms job type

Purpose

If you set JOBTYPK to (/j2ee/jms), you can run an existing j2ee jms class:

This section provides detailed information about the syntax diagram for the j2ee jms job type.

Format



Parameters

AUTHALIAS(authentication alias)

The Java™ Authentication and Authorization Service (JAAS) authentication alias. This keyword is valid only if you set INVOKEYTYPE to (DIRECT).

CONNFAC**FACTORY**(*connection factory*)

The administered object that a client uses to create a connection to the Java™ Messaging Service (JMS) provider. This keyword is required.

DESTINATION(*message destination*)

The administered object that encapsulates the identity of a message destination, which is where messages are delivered and consumed. This keyword is required.

INVOKEYTYPE(**INDIRECT**|**DIRECT**)

The type of scheduling. This keyword is valid only if you set JOBTYP to (/j2ee/jms). Allowed values are:

INDIRECT

IBM Z Workload Scheduler uses an existing WebSphere® Application Server scheduling infrastructure already configured on an external WebSphere® Application Server.

DIRECT

IBM Z Workload Scheduler immediately forwards the job to the components of the external WebSphere® Application Server instance.

JOBPWD(**YES**|**NO**|**AGENT**)

Specifies if the user name entered in JOBUSR or set by using the job-submit exit EQQUX001 is associated with a password.

If you set JOBPWD to **YES**, IBM Z Workload Scheduler searches for the user password in the USRPSW keyword of the USRREC initialization statement (for details about this statement, see *Customization and Tuning*).

If you set JOBPWD to **AGENT**, this means that the password is resolved locally on the agent. The password must have been defined on the agent by issuing the [param on page 49](#) command. This feature is available independently of the operating system of the workstation.

Typically, the password is required for users who schedule jobs to run on Windows™ workstations. Set JOBPWD to **NO** if the user works with UNIX™ workstations and if no password is required.

JOBTYP(/xajob/access *method name*)/j2ee/jms/web service/file transfer/database/java)

To run a j2ee jms job, specify **/j2ee/jms**. For information about the other types of job, see the specific section for each job type. This keyword is required.

/j2ee/jms

Set this value to submit a Java™ Messaging Service (JMS) job.

JOBUSR(*user name*)

The user name for accessing WebSphere® Application Server, if required.

If the user schedules jobs to run on Windows™ workstations, ensure that a user password is also defined (see the JOBPWD keyword).

- If you are defining a Windows™ domain user, use the following format:

```
JOBUSR(domainName\user1)
```

- If you are defining a Windows™ user in the *username@internet_domain* format, use the following format:

```
JOBUSR('administrator@mywindow.com')
```

To specify the user name, you can also use the job-submit exit EQQUX001. This user name overrides the value specified for JOBUSR. In turn, the value specified for JOBUSR overrides the value specified for USRNAM in the USRREC statement.

MESSAGE(*message*)

The object that is sent from one application to another.

WSNAME(*workstation_name*)

The name of the workstation from which user name and password must be retrieved. User name and password can be specified in the statement, using the JOBUSR and JOBPWD keywords, or can be associated to a workstation with the USRREC statement. With the WSNAME keyword, you can specify a workstation where the user name and password are stored, which is different from the workstation where the job runs.

Example

Examples

The following is an example of a JOBREC statement to submit a JMS job:

```
//JOBREC
JOBTYPE(/j2ee/jms)
INVOKEYTYPE(DIRECT)
CONNFACTORY(jms/MyCF)
DESTINATION(jms/MyQueue)
MESSAGE(This is my message)
JOBUSR(administrator)
JOBPWD(YES)
//END JOBREC
```

Syntax diagram for native job type

This section describes the native job type.

Syntax diagram for script and executable job type

Purpose

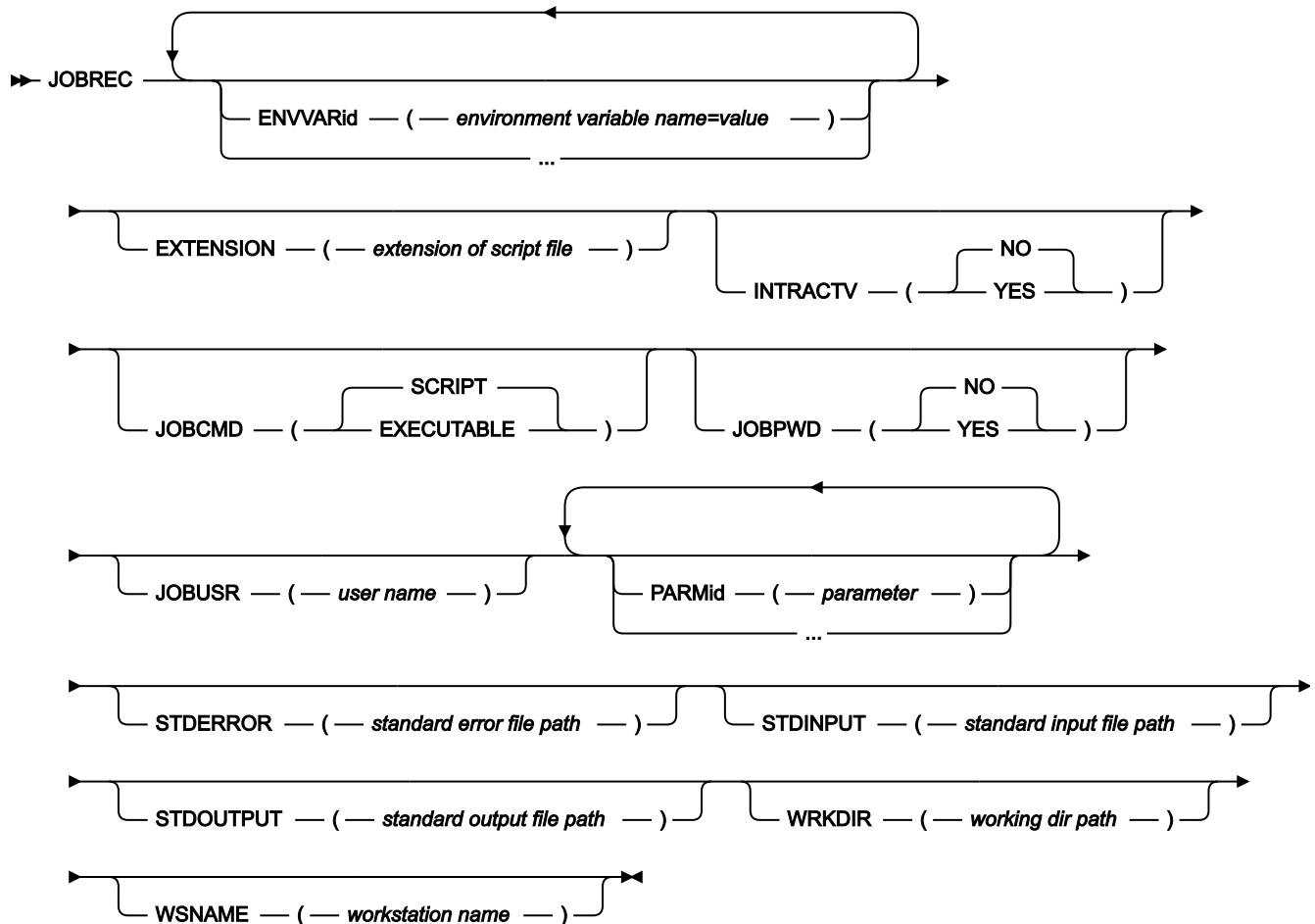
If you set JOBCMD to (*script*) or (*executable*), you can choose between the following native job types:

- Script
- Executable

This section provides detailed information about the syntax diagram for the script and executable job types. The following syntax rules apply:

- The JOBCMD keyword is required for executable and optional for script job types.

Format



Parameters

ENVVARid(*environment variable name=value*)

The environment variable that the task launcher must set before submitting the job. You can specify a series of ENVVARid keywords, where *id* is a unique numeric value, including null and zero.

Use the following format, which is case sensitive:

```
ENVVARid(environment variable name=value)
```

Where:

environment variable name

The name of the environment variable.

value

The value associated with the environment variable.

EXTENSION(extension of script file)

The extension (bat, vbs, etc.) of the script file launched by the job. This keyword is valid only when JOBCMD is set to SCRIPT.

INTRACTV(YES|NO)

Specifies if a Windows™ job runs interactively on the Windows™ desktop. This keyword is valid only for jobs running on Windows™ IBM Z Workload Scheduler Agent workstations.

JOBCMD(EXECUTABLE|SCRIPT)

The content of the job definition member. Valid values are:

SCRIPT

The job definition member contains the command or script to be run. You can add the EXTENSION keyword to specify the file extension.

EXECUTABLE

The job definition member invokes the job that is to be run, which is stored locally on the IBM Z Workload Scheduler Agent workstation. If you did not set any value for the WRKDIR keyword, you must specify the fully qualified path to the job.

JOBPWD(YES|NO)

Specifies that the user name set in JOBUSR or by using the job-submit exit EQQUX001 is associated with a password. If you set JOBPWD to YES, IBM Z Workload Scheduler searches for the user password in the USRPSW keyword of the USRREC statement (for details, see *IBM Z Workload Scheduler: Customization and Tuning*).

Typically, the password is required for users who schedule jobs to run on Windows™ workstations. Set JOBPWD to NO if the user works with UNIX™ workstations.

JOBUSR(user name)

The user name for running the script or executable, if required.

If the user schedules jobs to run on Windows™ workstations, ensure that a user password is also defined (see the JOBPWD keyword).

- If you are defining a Windows™ domain user, use the following format:

```
JOBUSR(domainName\user1)
```

- If you are defining a Windows™ user in the *username@internet_domain* format, use the following format:

```
JOBUSR('administrator@mywindow.com')
```

To specify the user name, you can also use the job-submit exit EQQUX001. This user name overrides the value specified for JOBUSR. In turn, the value specified for JOBUSR overrides the value specified for USRNAM in the USRREC statement.

PARMid(*parameter*)

The parameter to be used when running the script or command. You can specify a series of PARM*id* keywords, where *id* is a unique numeric value, including null and zero.

STDERROR(*standard error file path*)

The fully qualified name of the file where the standard error must be redirected.

STDINPUT(*standard input file path*)

The fully qualified name of the file to be used as standard input.

STDOUTPUT(*standard output file path*)

The fully qualified name of the file to be used as standard output.

WRKDIR(*working dir path*)

The fully qualified path to the directory where the script or command is to be run.

WSNAME(*workstation_name*)

The name of the workstation from which user name and password must be retrieved. User name and password can be specified in the statement, using the JOBUSR and JOBPWD keywords, or can be associated to a workstation with the USRREC statement. With the WSNAME keyword, you can specify a workstation where the user name and password are stored, which is different from the workstation where the job runs.

Example

Examples

The following is an example of a JOBRECE statement to invoke the script ALLOCATE.BAT stored locally on the IBM Z Workload Scheduler Agent workstation:

```
//JOBREC
WRKDIR(C:\)
STDERROR(D:\MYDIR\ERROR.TXT)
STDINPUT(D:\MYDIR\INPUT.TXT)
STDOUTPUT(D:\MYDIR\OUTPUT.TXT)
STDOUTPUT(D:\MYDIR\NEW.TXT)
JOBCMD(EXECUTABLE)
PARM(300)
PARM2(ELEMENT)
//END JOBREC
D:\MYDIR\ALLOCATE.BAT
```

The following is an example of a JOBRECE statement to run the command DIR on the directory C:\PROGRAM on the IBM Z Workload Scheduler Agent workstation:

```
//JOBREC
WRKDIR(C:\PROGRAM)
JOBCMD(SCRIPT)
```

```
EXTENSION(vbs)
//END JOBREC
DIR
```

Using variable substitution

The variable substitution mechanism that is used in IBM Z Workload Scheduler native job types works the same way when scheduling jobs on IBM Z Workload Scheduler Agents. The variable substitution mechanism changes when you define job types with advanced options, such as Java™ jobs that run a Java™ class, or file transfer jobs that transfer files to and from a server, to name a few. These types of jobs are created and edited using the Dynamic Workload Console.

When you define a variable for job types with advanced options, including those defined in a JOBREC using the JOBTYP keyword, the variable must be preceded by the \$ (dollar) sign and enclosed in {} (curly brackets), as follows:

```
${variable_name}
```

If you want to nest variables, use the following syntax:

```
${variable_1_name ${variable_2_name}}
```

In this case, variable_2 is resolved first and provides the value for variable_1.

With job types with advanced options you also have the option to define and use variables locally on the IBM Z Workload Scheduler Agents. To define a variable on the agent, use the [param on page 49](#) command. After defining the variable, to add it within a job definition so that it is resolved locally on the agent at run time, use the following syntax:

```
${Agent:variable_name}
```

User passwords can be similarly parameterized for local resolution. For details, see [Defining variables and passwords for local resolution on page 49](#).

Dependent variables and promptable variables are not supported in job types with advanced options.

For more information, see the section about variable substitution in *Managing the Workload*.

Defining file dependencies to check for file changes

You can use the **filewatch** utility to check for file system changes on files and directories, for example, when you want to make sure that a file exists before running a job that processes that file. By defining a job that runs the **filewatch** utility, you can implement file dependency, that is, a relationship between a file and an operation in which specific activity on the file determines the starting of the operation.



Note: The filewatch utility is not available for Limited Fault-Tolerant Agent for IBM i systems. For these systems, see the *Limited Fault-Tolerant Agent IBM i* manual.

To run the **filewatch** utility you can use variable substitution, which occurs when:

- The controller submits the job associated to a JOBLIB member. For details about the scheduler job tailoring, see *IBM Z Workload Scheduler: Managing the Workload*.
- The IBM Z Workload Scheduler Agent submits the job. For details about submitting a job, see [Defining a job in the JOBLIB data set on page 74](#).

Syntax

filewatch -v | -u | -?

filewatch -c[ondition] condval -f[ilename] file_path -dea[dline] deadline

[-i[nterval] interval]

[-dat[abase] log_extension]

[-r[eturncode] rc]

[-t[race] trace level]

The arguments are not positional. You can use an abbreviated format for all the arguments. Generally you can truncate the arguments to any position following the first character, except for `deadline` and `database` that require at least three characters.

-v

Returns the command version and exits.

-u

Returns command usage information and exits.

-?

Same as **-u**

-condition

The condition to be checked. Valid values are:

wcr | waitCreated

Waits until the file exists. If the file already exists, **filewatch** exits immediately. If `-filename` argument specifies a directory, the process waits until the directory exists and contains a new file.

wmr | waitModificationRunning

Waits until the file size or modification time changes. If `-filename` argument specifies a directory, the process waits until the size or earlier file modification time changes, when a file is created, modified, or deleted.

wmc | waitModificationCompleted

Checks that the file size or modification time stopped changing, meaning that **filewatch** waits for three search intervals without any change. If `-filename` argument specifies a directory, the process checks the size or the earlier file modification time change, for example if the number of directory files and the earlier file modification time does not change within three search intervals.

wmrc | waitModificationRunningCompleted

Waits until the file size or modification time changes and stops changing, meaning that, after the first change, **filewatch** waits for three search intervals without any further change. If `-filename` argument specifies a directory, the process checks the size or the earlier file modification time change, for example if the number of directory files and the earlier file modification time does not change within three search intervals.

wdl | waitDelete

Stops running when the file is deleted. If `-filename` argument specifies a directory, the process waits until a file is deleted from the directory.

-filename

The file path to be processed. You can embed blank or special characters, by using double quotation marks. Wildcard characters are not supported. To include more files in the monitoring process, you can store those files in a specific directory and use a file path specifying that directory. When **filewatch** is used on a Windows™ server, and the file path specifies a directory, the path name must not include a trailing forward slash "/" character.

-deadline

The deadline period, expressed in seconds. The allowed formats are:

- An integer in the range 0 to 31536000 (the upper value corresponds to one year). To have **filewatch** performing an indefinite loop, specify 0.
- *hh:mm:ss*, in the range 00:00:01 to 24:00:00, to select a time within the same day when **filewatch** started.

-interval

The file search interval, expressed in seconds. Specify an integer in the range:

- 5–3600 when specifying **wcr** or **wdl** as condition value.
- 30–3600 otherwise.

The default is 60.

-database

Optional extension of the log database, that is the database where **filewatch** stores the file status. If you specify this value, **filewatch** updates the `TWA_home/TWS/filewatchdb.log_extension` database, otherwise `TWA_home/TWS/filewatchdb` is updated.

For details about the log database, see [Maintaining the filewatch log database on page 106](#).

-returncode

The exit return code, if the condition is not successfully checked by the deadline. Specify an integer in the range 0 to 256. The returncode value is ignored if you specify 0 as deadline value. The default is 4.

-trace

Trace level for internal logging and traces. Possible values are:

0

To receive error messages only.

1

Indicates the *fine* level, to receive the most important messages with the lowest volume.

2

Indicates the *finer* level, to activate entry and exit traces.

3

Indicates the *finest* level, to receive the most detailed tracing output.

The default value is 0.

You find the trace output in the log of the job that run **filewatch**.

Maintaining the filewatch log database

The log database is used to store **filewatch** activity data. This log is located in the `TWA_home/TWS` directory. The default name is `filewatchcdb`.

When processing a file, **filewatch** uses it to store records with information about:

- File name.
- File status. The allowed values are Exist, Created, Running, Completed, RunningCompleted, or Deleted.
- Date and time of file creation at the entry update time.
- Date and time of the last file modification at the entry update time.
- Date and time of the last entry update.



Note: The log contains only one entry for each file.

To maintain the log database, use the **filewatchcdb** utility.

Syntax

filewatchcdb -v | -u | -?

filewatchcdb -c[ondition] condval

[-filename] file_path

[-dat[abase] log_extension]

The arguments are not positional.

Arguments

-v

Returns the command version and exits.

-u

Returns command usage information and exits.

-?

Same as **-u**

-condition

The condition to be checked. Valid values are:

bld | build

Rebuild the log database.

dlt | delete

Delete the log database record corresponding to the filename argument value.

gls | getLastStatusChange

Gets and returns the log database record corresponding to the filename argument value.

-filename

Use this argument to delete the entry corresponding to the specified value. You can embed blank or special characters, by using double quotation marks. Use * as wildcard character. This argument is required if you specify dlt (delete) or gls (getLastStatusChange) as condition value.

-database

Optional extension of the log database to be accessed. If you specify this value, **filewatchdb** accesses the `TWA_home/TWS/filewatchdb.log_extension` database, otherwise `TWA_home/TWS/filewatchdb` is accessed.

Checking for file changes with filemonitor

You can use the **filemonitor** utility to monitor for file changes (files that were either created or modified) within a time interval. This could be useful when, for example, you want to make sure that a file exists before running a job that processes that file. By defining a job that runs the **filemonitor** utility, you can implement file dependency, that is, a relationship between a file and an operation in which specific activity on the file determines the starting of the operation.



Note:

1. The **filemonitor** utility is available on the z-centric agents with IBM Workload Scheduler V9.4, Fix Pack 1 installed.
2. To issue **filemonitor** from the command line, ensure that you set up the environment with the following command from `<TWA_home>`:

**Windows**

```
twc_env.cmd
```

UNIX and Linux

```
../twc_env.sh
```

3. To issue **filemonitor** from a job, use the command `filemonitorlauncher`. The same arguments valid for **filemonitor** apply to `filemonitorlauncher`.

Syntax**filemonitor -V | -U**

```
filemonitor {-path path_to_monitor | -epath path_to_monitor}
[-exitOnPathToMonitorNotFound]
-event {fileCreated | fileModified} [-modificationCompletedTime seconds]
[-repositoryName repository_name]
[-repositoryPath repository_path]
[-recursive]
[-outputFile output_filename]
[-scanInterval scan_interval]
[-maxEventsThreshold max_events]
[-minFileSize min_file_size]
[-timeout seconds | time_of_the_day]
[-preserveEventsOnDelete seconds]
```

filemonitor -reset

```
[-repositoryName repository_name]
[-repositoryPath repository_path - generateEventsOnFirstScan]
```

Arguments

Note: If you set the same argument more than once, the last value specified is applied and no error message is reported.

-V

Displays the command version and exits.

-U

Displays command usage information and exits.

-path *path_to_monitor*

The path where the files to be processed are located. To include more files in the monitoring process, store all the files in the directory set with the **-path** argument.

The following syntax rules apply:

- Paths containing blank or special characters must be specified within double quotation marks.
- Wildcard characters question mark (?) and asterisk (*) are supported.

Universal Naming Convention (UNC) paths are also supported with the following syntax types:

- `\\server_name\share_name\directory_name\...`
- `\\?\UNC\server_name\share_name\directory_name`
- `\\?\path_name`

-epath *path_to_monitor*

The path where the files to be processed are located, always specified with slashes (/) as separators. Backslashes (\) are not allowed as separators, even if you are indicating a Windows path. To include more files in the monitoring process, store all the files in the directory set with the **-epath** argument.

The following syntax rules apply:

- Paths containing blank or special characters must be specified within double quotation marks.
- Wildcard characters question mark (?) and asterisk (*) are supported.
- Any character other than backslash (\), question mark (?), asterisk (*), square brackets ([]) or a backslash at the end of the value (\) is intended exactly as it is written. For example, `my\path` is not equivalent to `mypath`.
- Use the syntax `[class description]` to indicate a single character as follows:

[*range_of_characters*]

A range of characters separated by a minus sign (-). For example, `A-B` or `1-9`.

[*list_of_characters*]

A string of characters. For example, `ABC` or `1aX`.

- The characters exclamation mark (!) and caret (^) are used to reverse the sense. For example, `[!A-Z]` matches a single character that is not equivalent to any letter from A to Z. `[!F]` matches any character that is not F.

For example:

- `-epath /mypath/myp?th/e[!1].txt`
- `-epath /mypath/my[1-9]path/e[A-Z].txt`
- `-epath c:/mypath/p?th/e[!1].txt`

[-exitOnPathToMonitorNotFound]

Optionally, specify this argument to have the command exit if the specified path is not found.

-event {fileCreated | fileModified} [-modificationCompletedTime seconds]

The event type to be monitored. This argument is required when you specify **-path**.

Supported types are **fileCreated** and **fileModified**. For both types, you can set the

-modificationCompletedTime, which is the time interval, in seconds, that is used to determine when the event is sent.

-event fileCreated

As soon as the file is created, the event, `FileCreated`, is sent.

-event fileModified

As soon as the file is modified, the event, `ModificationCompleted`, is sent.

-event fileCreated -modificationCompletedTime seconds

When a file is created, the event is not sent immediately, but only after the interval of time specified by **-modificationCompletedTime***seconds* has elapsed, and during which no subsequent changes were made to the file, which includes the file being deleted and recreated with the same name.

-event fileModified -modificationCompletedTime seconds

When a file is modified, the event is not sent immediately, but only after the interval of time specified by **-modificationCompletedTime***seconds* has elapsed, and during which no subsequent changes were made to the file.

-repositoryName repository_name

Optionally, specify a database where to log the status of the retrieved files. The default is `filemonitor.db`.

-repositoryPath repository_path

The path to the **filemonitor** database. The default is:

Windows

```
<TWA_home>\TWS\stdlist\JM\filemonitor
```

UNIX and Linux

```
<TWA_DATA_DIR>/TWS/stdlist/JM/filemonitor
```

Paths containing spaces must be enclosed within double quotes. Wildcard characters are not allowed.

-generateEventsOnFirstScan

All the files retrieved during the first scan performed by **filemonitor** are considered as created or modified, and can generate events. This argument is available only if you specify the **repositoryPath** argument.

-recursive

Optional. The monitoring process includes subdirectories.

-outputFile *output_filename*

Optional. An output file where to store the retrieved events. Ensure that the directory where the output file is to be created is already existing. The command output is also printed to standard output and stored in the job properties, if you launch the **filemonitor** command from a job. Wildcard characters are not allowed.

-scanInterval *scan_interval*

Optional. A period of time in seconds between two consecutive checks on the files being created or modified. The default value is 300 seconds. The supported range is 1-3600 seconds.

-maxEventsThreshold *max_events*

Optional. The maximum number of events to be returned. The default value is **1**. If you specify **all**, all events are returned.

-minFileSize *min_file_size*

Optional. The minimum size in bytes that files must reach to be included in the scan. The default value is **0**.

-timeout *seconds* | *hh:mm:ss*

Optional. You can specify the maximum period, in seconds, that **filemonitor** runs or set the time of the current day until which **filemonitor** runs. The time refers to the local time of the workstation where **filemonitor** is running.

If you do not specify this parameter, **filemonitor** runs indefinitely.

-preserveEventsOnDelete

Optional. Returns events on the specified file, also if the file was deleted in the meantime. If you do not specify this argument, when a file is deleted all events preceding the file deletion, if any, are discarded.

-reset

Resets the information collected. With this argument you can optionally specify a **-repositoryName** and **-repositoryPath**.

Configuring trace properties for filemonitor

To configure the trace properties for **filemonitor**, edit the [FileMonitor.Logging] section in the `FileMonitor.ini` file located in the following path, and restart the **filemonitor** utility.

Windows

```
<TWA_Home>\TWS\ITA\cpa\config
```

UNIX and Linux

```
<TWA_DATA_DIR>\TWS\ITA\cpa\config
```

The section containing the trace properties is named:

```
[FileMonitor.Logging.cclog]
```

FileMonitor.trhd.fileName

The name of the trace file.

FileMonitor.trhd.maxFileBytes

The maximum size that the trace file can reach. The default is 1024000 bytes.

FileMonitor.trhd.maxFiles

The maximum number of trace files that can be stored. The default is 3.

FileMonitor.trfl.level

Determines the type of trace messages that are logged. Change this value to trace more or fewer events, as appropriate, or on request from Software Support. Valid values are:

DEBUG_MAX

Maximum tracing. Every trace message in the code is written to the trace logs.

INFO

All *informational*, *warning*, *error* and *critical* trace messages are written to the trace. The default value.

WARNING

All *warning*, *error* and *critical* trace messages are written to the trace.

ERROR

All *error* and *critical* trace messages are written to the trace.

CRITICAL

Only messages which cause the agent to stop are written to the trace.

The output trace (by default, `FileMonitor_trace.log`) is provided in XML format, and is located in `<TWA_Home>/TWS/stdlist/JM`.

Return Codes

0

The operation completed successfully.

4

Filemonitor stopped running, because timeout expired. No results were returned.

8

Filemonitor cannot run because the timeout is set to a time that is already passed.

12

Filemonitor stopped running, because the timeout expired without being able to access to the repository.

-1

An error occurred. Search the trace log (by default, `FileMonitor_trace.log`) for additional details.

Comments

If one or more files were created or modified in between subsequent invocations, the modifications are detected. However, files already detected in a previous run are not listed again in subsequent invocations. Wildcard characters are supported in both file names and directory names.

Example

In the following example, the **filemonitor** command checks every 2 minutes for all files created in the `c:\temp\logs` path and having a minimum size greater than 1024 bytes. The check is performed on all sub folders and the results are stored in `c:`

`\backup\logs\reports.txt`:

```
filemonitor -path "C:\temp\logs" -event fileCreated -recursive
-outputFile "C:\backup\logs\reports.txt"
-scanInterval 120 -maxEventsThreshold all -minFileSize 1024
```

Promoting critical jobs

An overview of the mechanism that is behind the promotion of critical jobs, or of their predecessors, running on IBM Z Workload Scheduler Agents, including pools and dynamic pools.

The promotion of a critical job is the process whereby the execution of a job in the critical path, that is about to miss its deadline, is provided with more resources. Thus, it is redirected to a more powerful or faster dynamic agent, to idle workstations in a pool or dynamic pool, to an agent allocated to the execution of critical work.

As soon as IBM Z Workload Scheduler determines that a job flagged as critical is not going to meet its deadline, it automatically updates the definition of the job, and of as many of its predecessors as necessary (that is, all the jobs in the critical path of the critical job), with the `tws.job.promoted` and `tws.job.resourcesForPromoted` promotion variables. These variables set the dynamic workload broker component to perform the load balancing required to guarantee that the job is run on time.

If the critical job is to be processed in a dynamic pool, the variables must be added manually to the job definition. For details, see the section about promoting jobs scheduled on dynamic pools in the *IBM Workload Scheduler: User's Guide and Reference*.

To control the priority allocation of system resources to the jobs in the critical network that must be promoted, you must configure the [PromotedNice on page 32](#) or [PromotedPriority on page 32](#) property of the `native job launcher` section in the `JobManager.ini` file of your IBM Z Workload Scheduler Agents.

When a [native on page 74](#) job is promoted, the `TWS_PROMOTED_JOB` environment variable is automatically added with value set to YES in the job definition. You can possibly take advantage of the value (YES or NO) of this variable in the script/program run by the native job, to implement a prioritization mechanism.

Killing IBM Z Workload Scheduler Agent operations

The `kill` action ends an operation in the current plan that has already started. You can kill a running IBM Z Workload Scheduler Agent operation from any of the following interfaces:

- [ISPF on page 114](#)
- [PIF on page 114](#)
- [BCIT on page 114](#)
- [OCL on page 114](#)
- [Dynamic Workload Console on page 115](#)

This action can be taken only on **STARTED** operations that are in the **EXECUTING** status, so that their job ID is known.

The operation number (except for the OCL instructions, where the application number is the required parameter) is required to identify the operation that is to be killed.

ISPF

About this task

To kill an IBM Z Workload Scheduler Agent operation, perform the following steps:

1. Select option 3 (OPERATIONS) from the Modifying the Current® Plan panel (fast path 5.3).
2. Specify the list criteria for the operation you want to kill.

You see the Modifying Operations in the Current® Plan panel (EQQMOPRL). This panel lists the operations in the current plan that fulfill the list criteria you specified.

3. Enter the **K** (Kill) row command next to the row that lists the operation.

The IBM Z Workload Scheduler Agent notifies the controller when the operation is killed.

PIF

To kill an IBM Z Workload Scheduler Agent operation, use the following request:

```
MODIFY CPOP OPNO=operation_number OPCMD=KJ
```

For more detailed information, see *Developer's Guide: Driving IBM Z Workload Scheduler*.

BCIT

To kill an IBM Z Workload Scheduler Agent operation, use the following instruction:

```
ACTION=MODIFY,RESOURCE=CPPOP,OPNO=operation_number,OPCMD=KJ
```

For more detailed information, see *Developer's Guide: Driving IBM Z Workload Scheduler*.

OCL

To kill an IBM Z Workload Scheduler Agent operation, use the following instruction:

```
KILLJOB APPL(application_number) OPNO(operation_number)
```

For more detailed information, see *Developer's Guide: Driving IBM Z Workload Scheduler*.

Dynamic Workload Console

About this task

To kill an IBM Z Workload Scheduler Agent operation, perform the following steps:

1. Click **IBM Z Workload Scheduler→;Workload→;Monitor→;Monitor Jobs**.
2. Select **All Jobs in plan** or another predefined task name.
3. Choose an engine name, or specify connection properties, and click **OK**.
4. Select a job and click **More Actions→;Kill**.

Chapter 9. Administering an IBM i dynamic environment

On overview on how to administer the IBM Workload Scheduler IBM i dynamic environment.

To begin scheduling jobs with advanced options on IBM i agents, the agents must be configured.

Scheduling on IBM i

About this task

When scheduling a job on IBM i systems, the job launches a native command that can be either a system or a user command. The native command consists of SBMJOB system command, which launches a batch job. The native command starts one or more batch jobs. The batch jobs can be monitored only if they are started by the native command. The IBM i Agent Monitor can monitor a maximum of 130 batch jobs.

You can specify the name of the queue where the monitoring agent component runs by using the **MonitorQueueName** property in the native job launcher section of the `JobManager.ini` file. If you do not specify this property, the default queue (QBATCH) is used.

For more information, see the section about configuring properties of the native job launcher [NativeJobLauncher] in the *IBM Workload Scheduler: Administration Guide*.

Use the Dynamic Workload Console to check whether an IBM i job is waiting for a reply to a message. An IBM i job that is waiting for a message reply is in the SUSP (suspended) status. This status indicates that the job is running while waiting for input. When the input is received, the job status changes to EXEC (executing).

For more information about job statuses, see the section about status description and mapping for distributed jobs in the *Dynamic Workload Console User's Guide*.

If an IBM i job is waiting for a reply, you can view the message text and the related reply. This information is written into the correspondent IBM® Workload Scheduler job log so that the IBM® Workload Scheduler operator knows the exact message the IBM i job is waiting for.

When an IBM i job is waiting for a reply to a message, you can reply to the message directly from the **Monitor Workload** of the Dynamic Workload Console. The job in SUSP (suspended) status requires your attention on additional information to be displayed. A pop-up window shows the message that is waiting for your reply. Reply to the message in the pop-up window, then select one of the following actions:

Forward action

To forward your reply. A message in the pop-up window confirms that your reply was sent successfully.

Cancel action

To cancel your reply. The pop-up window is closed.

You can even define standard rules to automate the reply to the waiting messages. When defining an IBM i job, by using the Dynamic Workload Console specify the list of messages for which you want to set up an automated reply. For each message, specify:

Message Id

Message identifier.

Message Text

Text of the message.

Message Reply

An automated reply that you want to send.

For more information, see the section about defining an IBM i job in *User's Guide and Reference*.

The agent joblog and TWSASPOOLS environment variable

About this task

By default, all information about the running of jobs is stored in the agent joblog. Most of this information usually consists of spool files. To select the spool file types that you want included in the agent joblog, use the **TWSASPOOLS** system variable, which works at IBM i agent level for any job to be submitted.

The **TWSASPOOLS** system variable forces the IBM i agent to either ignore all spool files or include one or more of them.

On the IBM i agent, create a new system level environment variable named TWSASPOOLS and set it to a list of the spool file types that are to be included. The list must begin with the SPOOLS: token.

For example, to force the IBM i agent to ignore all spool files, create the TWSASPOOLS variable as follows.

```
ADDENVVAR ENVVAR(TWSASPOOLS) VALUE(SPOOLS:) LEVEL(*SYS)
```

where the list after the SPOOL: token is empty. In this case, any agent joblog report for the IBM i agent is limited to the activity report that the Agent Monitor produces to trace its submission and monitoring action, and to the IBM i joblog of the Agent Monitor, which is always added at the end of the agent joblog.

To allow the IBM i agent to include only the QPRINT and the QPJOBLOG spool file types, that is, any spool files produced by printf instructions inside any ILE-C program and any produced joblog, create the TWSASPOOLS as follows:

```
ADDENVVAR ENVVAR(TWSASPOOLS) VALUE('SPOOLS: QPRINT QPJOBLOG') LEVEL(*SYS)
```

If the TWSASPOOLS variable already exists, change it as follows:

```
CHGENVVAR ENVVAR(TWSASPOOLS) VALUE('SPOOLS: QPRINT QPJOBLOG') LEVEL(*SYS)
```

If any VALUE parameter is set to an incorrect string, the IBM i agent ignores the TWSASPOOLS environment variable option. You can create and change the TWSASPOOLS environment variable while with the IBM i agent active, but no workload activity must be running.

Child job monitoring on IBM i agents

About this task

When you submit a command on an IBM i agent, the command might start one or more batch jobs. The IBM i agent monitors these batch jobs, which are referred to as child jobs.

When searching and monitoring any child jobs that are started, the IBM i agent uses a high percentage of its processing time.

If you know that your job scheduling does not start any child jobs or you have no interest in monitoring child jobs, you can instruct the IBM i agent to not search and monitor child jobs, and hence improve the performance of the agent.

You can exclude child job monitoring either at the agent level for all the commands or at the job definition level for a single command. If you want child job monitoring only for some specific submitted commands, you can set this option at the job definition level for a single command.

You can perform one or both of the following procedures to exclude or include child job monitoring:

Exclude child jobs from job monitoring at the agent level

By default child jobs are monitored. You can exclude child jobs from job monitoring for all submitted commands by creating the `TWS_NOCHILDS` system environment variable using the following IBM i system command:

```
ADDENVVAR ENVVAR(TWS_NOCHILDS) LEVEL(*SYS)
```

If the IBM i agent finds the `TWS_NOCHILDS` on the IBM i system, it does not monitor child jobs for any submitted command.

Exclude or include child jobs from job monitoring at the job definition level

You can exclude or include child jobs from job monitoring for a specific job by using `:NOCHILDS` or `:CHILDS` as ending tokens of the command string for the specific command.

- If you add the `:NOCHILDS` end token at the end of the native command you are submitting, the IBM i agent ignores any child jobs that are started by the command.
- If you add the `:CHILDS` end token at the end of the command you are submitting, the IBM i agent finds and monitors all the child jobs that are started by the command.



Note: The setting at job definition level overrides the setting at agent level.

Example

Examples

To monitor any child jobs that are started when the `PAYROLL` program is run, define the following command in the job definition:

- If the `TWS_NOCHILDS` system variable is defined on the IBM i system:

```
CALL PGM(MYLIB/PAYROLL) :CHILDS
```

- If the `TWS_NOCHILDS` system variable is not defined on the IBM i system:

```
CALL PGM(MYLIB/PAYROLL)
```

To not monitor any child jobs that are started when `MYSCHEDULE` program is run, define the following command in the job definition:

- If the `TWS_NOCHILDS` system variable is not defined on the IBM i system:

```
CALL PGM(MYLIB/MYSCHEDULE) :NOCHILDS
```

- If the `TWS_NOCHILDS` system variable is defined on the IBM i system:

```
CALL PGM(MYLIB/MYSCHEDULE)
```

Information about child job monitoring on IBM i agent joblog

About this task

If you include the child job monitoring on IBM i agent, as described in the section [Child job monitoring on IBM i agents on page 117](#), you can see information related to child job monitoring on the IBM i agent joblog.

Otherwise the information about the child job monitoring is not present in the joblog.

Example

Examples

This example shows the information related to child job monitoring included at job level for the `IBMI35C` job on the agent `D400`. The joblog shows the final status of 4 child jobs started from the submitted native command:

```
=====
=                               JOBLOG HEADER
=
= Occurrence name : AS400PGMCHILD
= Occurrence IA   : 2012/10/29 21:23
= Job name       : IBMI35C
= Work station   : D400
= Operation number: 002
= Start time     : 2012/10/31 10:23
= End time       : 2012/10/31 10:23
= Process ID     : 375602
= Duration       : 0.00.10
= Status         : ERROR
= Return code    : 30
= Hostname       : NC117025.ROMELAB.IT.IBM.COM
=
=====

The Dynamic Agent submitter-monitor job is qualified as:
  JobName=DYNAMICMON JobUser=ZOSDYN JobNumber=375602
Here follows the user command string
  <CALL PGM(MINERMA/SBM4JOBS) :CHILDS>
2012/10/31 10:23:27.893 - Dynamic Agent job submitted the User Command
  CALL PGM(MINERMA/SBM4JOBS)
```

```

The FOLLOWING 4 JOBS STARTED under the submitted User Command
  JobName=ZOSDYN      JobUser=ZOSDYN      JobNumber=375607
  JobName=ZOSDYN      JobUser=ZOSDYN      JobNumber=375609
  JobName=ZOSDYN      JobUser=ZOSDYN      JobNumber=375611
  JobName=ZOSDYN      JobUser=ZOSDYN      JobNumber=375612
Message CPF1241 (Success) received on MsgQueue ZOSDYN      QUSRSYS
  for the job ZOSDYN      ZOSDYN      375607
Message CPF1240 (Abend) received on MsgQueue ZOSDYN      QUSRSYS
  for the job ZOSDYN      ZOSDYN      375609
Message CPF1241 (Success) received on MsgQueue ZOSDYN      QUSRSYS
  for the job ZOSDYN      ZOSDYN      375611
Message CPF1241 (Success) received on MsgQueue ZOSDYN      QUSRSYS
  for the job ZOSDYN      ZOSDYN      375612
*** END codes gathered by the Monitor job ***
> END Status Code (Status): 0
> PROGRAM Return Code (Prc): 0
> USER Return Code (Urc): 0
  Urc was retrieved through SYSAPI

```

```

2012/10/31 10:23:37.904 - Dynamic Agent job ended monitoring the User Command
*** Return Code for submitted Command is 30 ***
*** job(s) started under User Command abended ***

```

This example shows the joblog for the IBMI35C job on the agent P400 when child job monitoring is excluded at job level:

```

=====
=                                JOBLOG HEADER                                =
=
= Occurrence name : AS400PGMNOCHILD
= Occurrence IA   : 2012/11/07 03:49
= Job name       : IBMI35NC
= Work station   : P400
= Operation number: 002
= Start time     : 2012/11/07 16:48
= End time       : 2012/11/07 16:48
= Process ID     : 491329
= Duration       : 0.00.05
= Status         : COMPLETED
= Return code    : 0
= Hostname       : NC117025.ROMELAB.IT.IBM.COM
=
=====

The Dynamic Agent submitter-monitor job is qualified as:
  JobName=DYNAMICMON JobUser=ZOSDYN JobNumber=491329
Here follows the user command string
  <CALL PGM(MINERMA/SBM4JOBS) :NOCHILDS>
2012/11/07 16:48:50.631 - Dynamic Agent job submitted the User Command
  CALL PGM(MINERMA/SBM4JOBS)
As per user choice, NO job started under the submitted command will be monitored
*** END codes gathered by the Monitor job ***
> END Status Code (Status): 0
> PROGRAM Return Code (Prc): 0
> USER Return Code (Urc): 0
  Urc was retrieved through SYSAPI

2012/11/07 16:48:55.685 - Dynamic Agent job ended monitoring the User Command

```



```
*** Return Code for submitted Command is 0 ***
*** User Command ended successfully ***
```

The agent return code retrieval

About this task

The IBM i programming model was originally based on an early object orientation model in which programs communicated through message passing, rather than using return codes. The introduction of the Integrated Language Programming (ILE) model led to the definitions of common areas to exchange data as return codes in the same job environment: the user return codes and the system end codes.

For information about user return codes, see [Controlling the job environment with the user return code on page 121](#).

When the IBM i agent verifies that a submitted command or job is completed, it assigns a return code to the job based on the job status of the completed job. The return code is set depending on the completion message of the command or job. If the command or job completes successfully, the return code is set to 0. If the command or job does not complete successfully, the return code is set to the value of the severity of the message related to the exception that caused the abnormal end of the job. The IBM i agent can also set the return code to the value of the user return code when it is returned by the submitted command. If retrieved, the user return code is used as the value to set the return code.

The return code value assigned to the job is included in the IBM i agent joblog for the job and sent back to the scheduler user interface (WEB UI or z/OS ISPF panels) as return code, for compatibility reasons with agents on other operating systems.

Controlling the job environment with the user return code

About this task

With the introduction of the IBM i ILE model, it is possible to retrieve a value returned by a called program inside the same job.

When the Agent Monitor verifies that a submitted command is completed, it retrieves the following end of job codes using an IBM i System API:

End status code or <Status> (0 if successful)

It indicates if the system issued a controlled cancellation of the job. Possible values are:

1

the subsystem or the job itself is canceled.

0

the subsystem or the job itself is not canceled.

blank

the job is not running.

Program return code or <Prc> (0000 if successful)

It specifies the completion code of the last program (such as a data file utility program, or an RPG or COBOL program, invoked by the job).

If the job includes no program, the program return code is 0.

User return code or <Urc> (0000 if successful)

It specifies the user-defined return code set by ILE high-level language constructs. For example, the return code of a program written in C language.

It represents the most recent return code set by any thread within the job.

If the submitted command is a call to a user ILE program returning a value on exiting, this value is found in the Urc end of job code.

You can decide how to control the job environment of your submitted jobs by preparing the commands to be submitted as CALLs to your ILE programs, where the internal flow is controlled and the end status is decided through proper exit values. If a user program ends in error for an incorrect flow control, without returning a value, the Agent Monitor does not set the Return Code as user return code (Urc), but follows the criteria described in [The agent return code retrieval on page 121](#).

The following example shows an ILE C user program where two batch jobs are launched and a value of 10 is returned to the caller, regardless of the completion status of the batch jobs.

```
=====
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
void main(int argc, char *argv[])
{
    int    EnvVarRC=0;
    printf("issuing SBMJOB CMD(CALL MYLIB/DIVBY0)...\n");
    system("SBMJOB CMD(CALL MYLIB/DIVBY0)");
    printf("issuing SBMJOB CMD(WRKACTJOB OUTPUT(*PRINT))...\n");
    system("SBMJOB CMD(WRKACTJOB OUTPUT(*PRINT)) LOG(4 0 *SECLVL)");
    exit(10);
    return;
}
=====
```

Alternative method to set the user return code

About this task

In some IBM i environments, the system API retrieving the user return code (Urc) from the Agent Monitor code does not retrieve the correct value for Urc. It is therefore not recommended that you use any IBM i system APIs to retrieve the user return code. To receive a value returned by a called program, it is better to provide, instead, a parameter to receive the value.

Even if the Agent Monitor can retrieve the user return code using system API, an alternative user return code retrieval method was implemented in the Agent Monitor code. The alternative retrieval method has the following logic. The `USERRC` job

environment variable is created and set to the *INI* value before submitting the user command. When the command ends, the Agent Monitor retrieves its user return code using the system APIs, but it also verifies if the `USERRC` job environment variable was updated at user program level. If a value different from *INI* is found, this is considered as the user return code and the value retrieved using the system APIs is ignored because the user program modified the value of `USERRC` job environment variable.

The change of the `USERRC` variable at user program level requires the `USERRC` value change before exiting from the application user code. In the ILE C case, you can do this using the `putenv` statement, where the user return code is set to be returned.

The following example shows how the user code returns the user return code using the IBM i agent reserved job environment variable `USERRC`. This code was obtained from the code of the example in [Controlling the job environment with the user return code on page 121](#) by replacing the `exit` with the `putenv` statement.

```
=====
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
void main(int argc, char *argv[])
{
    int    EnvVarRC=0;
    printf("issuing SBMJOB CMD(CALL MYLIB/DIVBY0)...\n");
    system("SBMJOB CMD(CALL MYLIB/DIVBY0)");
    printf("issuing SBMJOB CMD(WRKACTJOB OUTPUT(*PRINT))...\n");
    system("SBMJOB CMD(WRKACTJOB OUTPUT(*PRINT)) LOG(4 0 *SECLVL)");
    EnvVarRC = putenv("USERRC=10");
    return;
}
=====
```

Configuring the agent on IBM i systems

An overview on how to configure the agent on IBM i systems.

The configuration settings of the agent are contained in the `JobManager.ini` file and in the `JobManagerGW.ini` file (for the path of these files, see the section about installation paths in *IBM Workload Scheduler: Planning and Installation*).

The configuration files are made up of many different sections. Each section name is enclosed between square brackets and each section includes a sequence of `variable = value` statements.

You can customize properties for the following:

- Log properties
- Trace properties when the agent is stopped. You can also customize traces when the agent is running using the procedure described in [Configuring trace properties when the agent is running on page 25](#).
- Native job executor
- Java™ job executor
- Resource advisor agent
- System scanner

On IBM i systems, the log messages are written in the following file:

```
TWA_DATA_DIR>/stdlist/JM/JobManager_message.log
```

On IBM i systems, the trace messages are written in the following files:

```
<TWA_DATA_DIR>/TWS/stdlist/JM/ITA_trace.log
<TWA_DATA_DIR>/TWS/stdlist/JM/JobManager_trace.log
<TWA_DATA_DIR>/TWS/stdlist/JM/javaExecutor0.log
```

Not all the properties in the `JobManager.ini` file and in the `JobManagerGW.ini` file can be customized. For a list of the configurable properties, see the following sections:

- [Configuring log message properties \[JobManager.Logging.cclog\] on page 22.](#)
- [Configuring trace properties when the agent is stopped \[JobManager.Logging.cclog\] on page 23.](#)
- [Configuring common launchers properties \[Launchers\] on page 28.](#)
- [Configuring properties of the native job launcher \[NativeJobLauncher\] on page 30.](#)
- [Configuring properties of the Java job launcher \[JavaJobLauncher\] on page 33.](#)
- [Configuring properties of the Resource advisor agent \[ResourceAdvisorAgent\] on page 34.](#)
- [Configuring properties of the System scanner \[SystemScanner\] on page 36](#)



Note: In the `JobManager.ini` file and in the `JobManagerGW.ini` file you must refer to Java 64 bit version.

Configuring log message properties [JobManager.Logging.cclog]

About this task

To configure the logs, edit the [JobManager.Logging.cclog] section in the `JobManager.ini` file. This procedure requires that you stop and restart the IBM Workload Scheduler agent

The section containing the log properties is named:

```
[JobManager.Logging.cclog]
```

You can change the following properties:

JobManager.loggerhd.fileName

The name of the file where messages are to be logged. the default value is

On Windows operating systems

```
TWA_home\stdlist\JM\JobManager_message.log
```

On UNIX operating systems

```
$(TWA_DATA_DIR)/stdlist/JM/JobManager_message.log
```

JobManager.loggerhd.maxFileBytes

The maximum size that the log file can reach. The default is **1024000** bytes.

JobManager.loggerhd.maxFiles

The maximum number of log files that can be stored. The default is **3**.

JobManager.loggerhd.fileEncoding

By default, log files for the agent are coded in UTF-8 format. If you want to produce the log in a different format, add this property and specify the required codepage.

JobManager.loggerfl.level

The amount of information to be provided in the logs. The value ranges from 3000 to 7000. Smaller numbers correspond to more detailed logs. The default is **3000**.

JobManager.ffdc.maxDiskSpace

Exceeding this maximum disk space, log files collected by the first failure data capture mechanism are removed, beginning with the oldest files first.

JobManager.ffdc.baseDir

The directory to which log and trace files collected by the ffdc tool are copied. The default directory is

On Windows operating systems

<TWA_home>\stdlist\JM\JOBMANAGER-FFDC

On UNIX operating systems

\$(TWA_DATA_DIR)/stdlist/JM/JOBMANAGER-FFDC

JobManager.ffdc.filesToCopy

Log and trace files (JobManager_message.log and JobManager_trace.log) collected by the ffdc tool located in <TWA_home>\TWS\stdlist\JM. The files are available in the following paths:

On Windows operating systems

- TWA_home\TWS\stdlist\JM\JobManager_message.log
- TWA_home\TWS\stdlist\JM\JobManager_trace.log

On UNIX operating systems

- \$(TWA_DATA_DIR)/stdlist/JM/JobManager_message.log
- \$(TWA_DATA_DIR)/stdlist/JM/JobManager_trace.log

When a message is logged (JobManager.ffdc.triggerFilter = JobManager.msgIdFilter) that has an ID that matches the pattern "AWSITA*E" (JobManager.msgIdFilter.msgIds = AWSITA*E), which corresponds to all error messages, then the log and trace files (JobManager.ffdc.filesToCopy = "/opt/IBM/TWA_<TWS_user>/TWS/stdlist/JM/JobManager_message.log" "/opt/IBM/TWA_<TWS_user>/TWS/stdlist/JM/JobManager_trace.log") are copied (JobManager.ffdc.className = ccg_ffdc_filecopy_handler) to the directory JOBMANAGER-FFDC (JobManager.ffdc.baseDir = /opt/IBM/TWA_<TWS_user>/TWS/stdlist/JM/JOBMANAGER-FFDC). If the files

copied exceed 10 MB (`JobManager.ffdc.maxDiskSpace = 10000000`), then the oldest files are removed first (`JobManager.ffdc.quotaPolicy = QUOTA_AUTODELETE`).

Configuring trace properties when the agent is stopped [JobManager.Logging.cclog]

How to configure the trace properties when the agent is stopped.

To configure the trace properties when the agent is stopped, edit the [JobManager.Logging] section in the `JobManager.ini` file and then restart the IBM Workload Scheduler agent.

The section containing the trace properties is named:

```
[JobManager.Logging.cclog]
```

You can change the following properties:

JobManager.trhd.fileName

The name of the trace file. the default path is as follows:

On Windows operating systems

```
TWA_home/TWS/stdlist/JM/JobManager_trace.log
```

On UNIX operating systems

```
$(TWA_DATA_DIR)/stdlist/JM/JobManager_trace.log
```

JobManager.trhd.maxFileBytes

The maximum size that the trace file can reach. The default is 10240000 bytes.

JobManager.trhd.maxFiles

The maximum number of trace files that can be stored. The default is 5.

JobManager.trfl.level

Determines the type of trace messages that are logged. Change this value to trace more or fewer events, as appropriate, or on request from IBM Software Support. Valid values are:

DEBUG_MAX

Maximum tracing. Every trace message in the code is written to the trace logs.

INFO

All *informational*, *warning*, *error* and *critical* trace messages are written to the trace. The default value.

WARNING

All *warning*, *error* and *critical* trace messages are written to the trace.

ERROR

All *error* and *critical* trace messages are written to the trace.

CRITICAL

Only messages which cause the agent to stop are written to the trace.

The output trace (JobManager_trace.log) is provided in XML format.

After installing the z-centric agent or dynamic agent on Windows 2012, the JobManager_trace.log might not be created. In this case, perform the following procedure:

1. Stop the agent.
2. Create a backup copy of JobManager.ini, and edit the original file by changing the row:

```
JobManager.trhd.className = ccg_multiproc_filehandler
```

to

```
JobManager.trhd.className = ccg_filehandler
```

3. Restart the agent.

Configuring trace properties when the agent is running

Use the **twstrace** command to set the trace on the agent when it is running.

Using the **twstrace** command, you can perform the following actions on the agent when it is running:

- [See command usage and verify version on page 25.](#)
- [Enable or disable trace on page 26.](#)
- Set the traces to a specific level, specify the number of trace files you want to create, and the maximum size of each trace file. See [Set trace information on page 26.](#)
- [Show trace information on page 26.](#)
- Collect trace files, message files, and configuration files in a compressed file using the command line. See [Collect trace information on page 27.](#)
- Collect trace files, message files, and configuration files in a compressed file using the Dynamic Workload Console. See the section about retrieving agent traces from the Dynamic Workload Console in *Troubleshooting Guide*.

You can also configure the traces when the agent is not running by editing the [JobManager.Logging] section in the JobManager.ini file as described in [Configuring the agent on page 18](#). This procedure requires that you stop and restart the agent.

twstrace command

Use the **twstrace** command to configure traces, and collect logs, traces, and configuration files (ita.ini and jobManager.ini) for agents. You collect all the information in a compressed file when it is running without stopping and restarting it.

See command usage and verify version

To see the command usage and options, use the following syntax.

Syntax

twstrace -u | -v

Parameters

-u

Shows the command usage.

-v

Shows the command version.

Enable or disable trace

To set the trace to the maximum or minimum level, use the following syntax.

Syntax

twstrace -enable | -disable

Parameters

-enable

Sets the trace to the maximum level. The maximum level is **1000**.

-disable

Sets the trace to the minimum level. The minimum level is **3000**.

Set trace information

To set the trace to a specific level, specify the number of trace files you want to create, and the maximum size the trace files can reach, use the following syntax.

Syntax

twstrace [-level <level_number>] [-maxFiles <files_number>] [-maxFileBytes <bytes_number>]

Parameters

-level <level_number>

Sets the trace level. Specify a value in the range from 1000 to 3000, which is also the default value. Note that if you set this parameter to 3000, you have the lowest verbosity level and the fewest trace messages. To have a better trace level, with the most verbose trace messages and the maximum trace level, set it to **1000**.

-maxFiles <files_number>

Specify the number of trace files you want to create.

-maxFileBytes <bytes_number>

Set the maximum size in bytes that the trace files can reach. The default is **1024000** bytes.

Show trace information

To display the current trace level, the number of trace files, and the maximum size the trace files can reach, use the following syntax.

Syntax

twstrace -level | -maxFiles | -maxFileBytes

Parameters

-level

See the trace level you set.

-maxFiles

See the number of trace files you create.

-maxFileBytes

See the maximum size you set for each trace file

Example

Sample

The example shows the information you receive when you run the following command:

```
twstrace -level -maxFiles -maxFileBytes
```

```
AWSITA176I The trace properties are: level="1000",  
max files="3", file size="1024000".
```

Collect trace information

To collect the trace files, the message files, and the configuration files in a compressed file, use the following syntax.

Syntax

twstrace -getLogs [-zipFile <compressed_file_name>] [-host <host_name>] [-protocol {http | https}] [-port <port_number>] [-iniFile <ini_file_name>]

Parameters

-zipFile <compressed_file_name>

Specify the name of the compressed file that contains all the information, that is logs, traces, and configuration files (ita.ini and jobManager.ini) for the agent. The default is **logs.zip**.

-host <host_name>

Specify the host name or the IP address of the agent for which you want to collect the trace. The default is **localhost**.

-protocol http|https

Specify the protocol of the agent for which you are collecting the trace. The default is the protocol specified in the `.ini` file of the agent.

-port <port_number>

Specify the port of the agent. The default is the port number of the agent where you are running the command line.

-iniFile <ini_file_name>

Specify the name of the `.ini` file that contains the SSL configuration of the agent for which you want to collect the traces. If you are collecting the traces for a remote agent for which you customized the security certificates, you must import the certificate on the local agent and specify the name of the `.ini` file that contains this configuration. To do this, perform the following actions:

1. Extract the certificate from the keystore of the remote agent.
2. Import the certificate in a local agent keystore. You can create an ad hoc keystore whose name must be **TWSClientKeyStore.kdb**.
3. Create an `.ini` file in which you specify:
 - **0** in the **tcp_port** property as follows:

```
tcp_port=0
```

- The port of the remote agent in the **ssl_port** property as follows:

```
ssl_port=<ssl_port>
```

- The path to the keystore you created in Step 2 on page 130 in the **key_repository_path** property as follows:

```
key_repository_path=<local_agent_keystore_path>
```

Configuring common launchers properties [Launchers]

About this task

In the `JobManager.ini` file, the section containing the properties common to the different launchers (or executors) is named:

```
[Launchers]
```

The following properties are available:

BaseDir

The installation path of the IBM Workload Scheduler agent. Do not modify this value.

CommandHandlerMinThreads

Indicates the maximum number of commands that can be run on the agent concurrently. Limits to the number of jobs vary depending on the resources of your workstation, however consider that operations on comdhandler are usually short. The default is **20**. Usually, there is no need to modify this setting, even if you

plan a very high workload on the agent. You might want to change it if many commands are run concurrently on the agent, for example, many concurrent requests to retrieve job logs.

CommandHandlerMaxThreads

Indicates the maximum number of commands that can be run on the agent concurrently. Limits to the number of jobs vary depending on the resources of your workstation, however consider that operations on comdhandler are usually short. The default is **100**. Usually, there is no need to modify this setting, even if you plan a very high workload on the agent. You might want to change it if many commands are run concurrently on the agent, for example, many concurrent requests to retrieve job logs.

CpaHeartBeatTimeSeconds

The polling interval in seconds used to verify if the **agent** process is still up and running. If the agent process is inactive the product stops also the **JobManager** process. The default is **30**. Modify only if you use dynamic pools with CPU-based requirements or optimization policies. With a lower value, the agent reacts quickly to CPU modifications, but this might cause unstable values in case of CPU spikes. Lower values causes a higher use of resources on the agent.

DirectoryPermissions

The access rights assigned to the agent for creating directories when running jobs. The default is **0755**. Supported values are UNIX-format entries in hexadecimal notation.

DownloadDir

The name of the directory where the fix pack installation package or upgrade image for IBM Workload Scheduler fault-tolerant agents or dynamic agents is downloaded during the centralized agent update process. If not specified, the following default directory is used:

On Windows operating systems:

`TWA_home\TWS\stdlist\JM\download`

On UNIX operating systems:

`TWA_DATA_DIR/TWS/stdlist/JM/download`

The centralized agent update process does not apply to z-centric agents.

ExecutorsMaxThreads

Specifies the maximum number of jobs the dynamic agent can run concurrently. For example, to allow the dynamic agent to run a maximum of 500 jobs concurrently, set this parameter to **500**. The default is **400**.

ExecutorsMinThreads

Specifies the minimum number of jobs the dynamic agent can run concurrently. For example, to allow the dynamic agent to run a minimum of 500 jobs concurrently, set this parameter to **500**. The default is **38**. Modify if the number of expected concurrent jobs is much higher than 38. The agent dynamically allocates more threads if necessary, until it reaches the value specified in **ExecutorsMaxThreads**.

FilePermissions

The access rights assigned to the agent for creating files when running jobs. The default is **0755**. Supported values are UNIX-format entries in hexadecimal notation.

MaxAge

The number of days that job logs are kept (in path *TWA_home/TWS/stdlist/JM*) before being deleted. The default is **30**. Possible values range from a minimum of 1 day.

MaxMetricsSizeKBytes

The maximum combined size in kilobytes of both metrics and output properties. This property is not present by default in the `JobManager.ini` file, but can be manually added to help optimize CPU and disk space usage. If you encounter a message similar to the one below, configuring this property might be beneficial:

```
AWSITA447E Failed to add the value for the specified metric or property:
storage capacity for metrics and properties is insufficient.
```

If this property is not present in the `JobManager.ini` file, the default value is 16,384 KB.

NotifierMaxThreads

Notifier threads are in charge of notifying the dynamic workload broker of each status change in a job. This parameter specifies the maximum number of job status changes that can be notified to the dynamic workload broker.

NotifierMinThreads

Notifier threads are in charge of notifying the dynamic workload broker of each status change in a job. This parameter specifies the minimum number of job status changes that can be notified to the dynamic workload broker. The default value is **3**. Modify this parameters only in case of unexpected errors and after consulting with software support team.

SpoolDir

The path to the folder containing the jobstore and outputs. The default is:

On Windows operating systems

`TWA_home/TWS/stdlist/JM`

On UNIX operating systems

`$(TWA_DATA_DIR/stdlist/JM`

StackSizeBytes

The size of the operating system stack in bytes. The default is **DEFAULT**, meaning that the **agent** uses the default value for the operating system. Do not modify this parameter unless instructed to do so by the software support team. Incorrect values can cause the agent to crash.

Configuring properties of the native job launcher [NativeJobLauncher]

About this task

In the `JobManager.ini` file, the section containing the properties of the native job launcher is named:

```
[NativeJobLauncher]
```

You can change the following properties:

AllowRoot

Applies to UNIX™ systems only. Specifies if the root user can run jobs on the agent. It can be `true` or `false`. The default is `false`. This property does not apply to IBM i, use the `AllowQSECOFR` option instead

AllowQSECOFR

Applies to IBM i systems only. Specifies if QSECOFR user can run jobs on the agent. It can be `true` or `false`. The default is `true`. Add a line like `AllowQSECOFR = false` to the `JobManager.ini` file to deny job execution to QSECOFR.

CheckExec

If `true`, before launching the job, the agent checks both the availability and the execution rights of the binary file. The default is `true`.

DefaultWorkingDir

Specifies the working directory of native jobs. You can also specify the value for the working directory when creating or editing the job definition in the Graphical Designer. When specified in the Graphical Designer, this value overrides the value specified for the **DefaultWorkingDir** property. If you do not specify any working directories, the `<TWS_home>\bin` directory is used.

JobUnspecifiedInteractive

Applies to Windows™ operating systems only. Specifies if native jobs are to be launched in interactive mode. It can be `true` or `false`. The default is `false`.

KeepCommandTraces

Set to `true` to store the traces of the method invocation for actions performed on a job definition, for example, when selecting from a picklist. These files are stored in the path `/opt/IBM/TWA_<TWS_user>/TWS/stdlist/JM/r3batch_cmd_exec`. The default setting is `false`.

KeepJobCommandTraces

Set to `true` to store the traces of the method invocation for actions performed on a job instance, for example, viewing a spool list. These files are stored in the `.zip` file of the job instance. The default setting is `true`.

LoadProfile

Applies to agents on Windows servers only. Specifies if the user profile is to be loaded. It can be `true` or `false`. The default is `true`.

MonitorQueueName

Specifies the name of the queue where the IBM i jobs are monitored. If you do not specify this property, the default queue (QBATCH) is used.

PortMax

The maximum range of the port numbers used by the task launcher to communicate with the Job Manager.
The default is 0, meaning that the operating system assigns the port automatically.

PortMin

The minimum range of the port numbers used by the task launcher to communicate with the Job Manager. The default is 0, meaning that the operating system assigns the port automatically.

PostJobExecScriptPathName

The fully qualified path of the script file that you want to run when the job completes. By default, this property is not present in the `JobManager.ini` file. If you do not specify any file path or the script file doesn't exist, no action is taken.

This property applies to dynamic agent and z/OS agent. For details about running a script when a job completes, see *User's Guide and Reference*.

PromotedNice

Used in workload service assurance. This property is not supported on the Agent for z/OS.

For UNIX and Linux operating systems only, assigns the priority value to a critical job that needs to be promoted so that the operating system processes it before others. Applies to critical jobs or predecessors that need to be promoted so that they can start at their critical start time.

Boundary values vary depending upon each specific platform, but generally lower values correspond to higher priority levels and vice versa. The default is -1.

Be aware that:

- The promotion process is effective with negative values only. If you set a positive value, the system runs it with the -1 default value.
- An out of range value (for example -200), prompts the operating system to automatically promote the jobs with the lowest allowed nice value.
- Overusing the promotion mechanism (that is, defining an exceedingly high number of jobs as mission critical and setting the highest priority value here) might overload the operating system, negatively impacting the overall performance of the workstation.

PromotedPriority

Used in workload service assurance. This property is not supported on the Agent for z/OS.

For Windows operating systems only, sets to this value the priority by which the operating system processes a critical job when it is promoted. Applies to critical jobs or predecessors that need to be promoted so that they can start at their critical start time. Valid values are:

- High
- AboveNormal (the default)

- Normal
- BelowNormal
- Low Or Idle

Note that if you set a lower priority value than the one non-critical jobs might be assigned, no warning is given.

RequireUserName

When `true`, requires that you add the user name in the JSDL job definition.

When `false`, runs with the user name used by job manager, that is:

- `TWS_user` on UNIX™ and Linux™ systems
- The local system account on Windows™ systems

The default is `false`.

RunExecutablesAsIBMiJobs

If you set this property to `true`, you can define IBM i jobs as generic jobs without using the XML definition. Generic jobs are automatically converted to IBM i jobs. As a side effect, generic jobs cannot be run when this parameter is enabled (`RunExecutablesAsIBMiJobs=true`). There is no default value because this property is not listed in the `JobManager.ini` file after the agent installation.

If you set this property to `true`, ensure that the user you used to install the agent has been granted the `*ALLOBJ` special authority.

RunInteractiveJobOnInvalidSession

Applies only to native and executable jobs starting interactive programs running on dynamic agents installed on Windows operating systems. Interactive programs run only if the job user has an active session open when the job runs. If there is no active session for the job user, the job behavior is defined by this property, as follows. Set the property to `true` to enable jobs to start interactive programs even if there is no active session for the job user. Set the property to `false` to prevent jobs from starting interactive programs if there is no active session for the job user.

ScriptSuffix

The suffix to be used when creating the script files. It is:

`.cmd`

For Windows™

`.sh`

For UNIX™

VerboseTracing

Enables verbose tracing. It is set to `true` by default.

Configuring properties of the Java™ job launcher [JavaJobLauncher]

About this task

In the `JobManager.ini` file, the section containing the properties of the Java™ job launcher is named:

```
[JavaJobLauncher]
```

You can change the following property:

JVMOptions

The options to provide to the Java™ Virtual Machine used to start job types with advanced options. Supported keywords for establishing a secure connection are:

- `https.proxyHost`
- `https.proxyPort`

Supported keywords for establishing a non-secure connection are:

- `Dhttp.proxyHost`
- `Dhttp.proxyPort`

For example, to set job types with advanced options, based on the default JVM http protocol handler, to the unauthenticated proxy server called with name `myproxyserver.mycompany.com`, define the following option:

```
JVMOptions = -Dhttp.proxyHost=myproxyserver.mycompany.com -Dhttp.proxyPort=80
```

Configuring properties of the Resource advisor agent [ResourceAdvisorAgent]

About this task

In the `JobManager.ini` and `JobManagerGW.ini` files, the section containing the properties of the Resource advisor agent is named:

```
[ResourceAdvisorAgent]
```

You can change the following properties:

BackupResourceAdvisorUrls

The list of URLs returned by the IBM Workload Scheduler master in a distributed environment or by the dynamic domain manager either in a z/OS or in a distributed environment. The agent uses this list to connect to the master or dynamic domain manager.

CPUScannerPeriodSeconds

The time interval that the Resource advisor agent collects resource information about the local CPU. The default value is every 10 seconds.

FullyQualifiedHostname

The fully qualified host name of the agent. It is configured automatically at installation time and is used to connect with the master in a distributed environment or with the dynamic domain manager in a z/OS or in a distributed environment. Edit only if the host name is changed after installation.

NotifyToResourceAdvisorPeriodSeconds

The time interval that the Resource advisor agent forwards the collected resource information to the Resource advisor. The default value is every 119 seconds.

ResourceAdvisorUrl**JobManager.ini**

The URL of the master in a distributed environment, or of the dynamic domain manager in a z/OS or in a distributed environment, that is hosting the agent.

This URL is used until the server replies with the list of its URLs. The value is

`https://$(tdwb_server):$(tdwb_port)/JobManagerRESTWeb/JobScheduler/resource`, where:

\$(tdwb_server)

is the fully qualified host name of the master in a distributed environment or of the dynamic domain manager either in a z/OS or in a distributed environment.

\$(tdwb_port)

is the port number of the master in a distributed environment or of the dynamic domain manager either in a z/OS or in a distributed environment.

It is configured automatically at installation time. Edit only if the host name or the port number are changed after installation, or if you do not use secure connection (set to `http`). If you set the port number to zero, the resource advisor agent does not start. The port is set to zero if at installation time you specify that you will not be using the master in a distributed environment or the dynamic domain manager either in a z/OS or in a distributed environment.

In a distributed environment, if **-gateway** is set to either `local` or `remote`, then this is the URL of the dynamic agent workstation where the gateway resides and through which the dynamic agents communicate. The value is

`https://$(tdwb_server):$(tdwb_port)/ita/JobManagerGW/JobManagerRESTWeb/JobScheduler/resource`, where:

\$(tdwb_server)

The fully qualified host name of the dynamic agent workstation where the gateway resides and through which the dynamic agent communicates with the dynamic workload broker.

`$(tdwb_port)`

The port number of the dynamic agent workstation where the gateway resides and through which the dynamic agent communicates with the dynamic workload broker.

JobManagerGW.ini

In a distributed environment, if **-gateway** is set to `local`, then **ResourceAdvisorUrl** is the URL of the master or dynamic domain manager. The value is

`https://$(tdwb_server):$(tdwb_port)/JobManagerRESTWeb/JobScheduler/resource`, where:

`$(tdwb_server)`

The fully qualified host name of the master or dynamic domain manager.

`$(tdwb_port)`

The port number of the master or dynamic domain manager.

ScannerPeriodSeconds

The time interval that the Resource advisor agent collects information about all the resources in the local system other than CPU resources. The default value is every 120 seconds.

The resource advisor agent, intermittently scans the resources of the machine (computer system, operating system, file systems and networks) and periodically sends an update of their status to the master or dynamic domain manager either in a z/OS or in a distributed environment.

The CPU is scanned every `CPUScannerPeriodSeconds` seconds, while all the other resources are scanned every `ScannerPeriodSeconds` seconds. As soon as one of the scans shows a significant change in the status of a resource, the resources are synchronized with the master in a distributed environment or the dynamic domain manager either in a z/OS or in a distributed environment. The following is the policy followed by the agent to tell if a resource attribute has significantly changed:

- A resource is added or deleted
- A string attribute changes its value
- A CPU value changes by more than `DeltaForCPU`
- A file system value changes by more than `DeltaForDiskMB` megabytes
- A Memory value changes by more than `DeltaForMemoryMB` megabytes

If there are no significant changes, the resources are synchronized with the IBM Workload Scheduler master in a distributed environment or with the dynamic domain manager either in a z/OS or in a distributed environment every `NotifyToResourceAdvisorPeriodSeconds` seconds.

Configuring properties of the System scanner [SystemScanner]

About this task

In the `JobManager.ini` file, the section containing the properties of the System scanner is named:

```
[SystemScanner]
```

You can change the following properties:

CPUSamples

The number of samples used to calculate the average CPU usage. The default value is 3.

DeltaForCPU

The change in CPU usage considered to be significant when it becomes higher than this percentage (for example, DeltaForCPU is 20 if the CPU usage changes from 10 percent to 30 percent). The default value is 20 percent.

DeltaForDiskMB

The change in use of all file system resources that is considered significant when it becomes higher than this value. The default value is 100 MB.

DeltaForMemoryMB

The change in use of all system memory that is considered significant when it becomes higher than this value. The default value is 100 MB.

Configuring to schedule job types with advanced options

About this task

You can define job types with advanced options by using the related configuration files. The options you define in the configuration files apply to all job types with advanced options of the same type. You can override these options when defining the job by using the Dynamic Workload Console or, if you are in a distributed environment, the **composer** command.

Configuration files are available on each dynamic agent in TWA_home/TWS/JavaExt/cfg for the following job types with advanced options:

Table 4. Configuration files for job types with advanced options

Job type	File name	Keyword
- Database job type - MSSQL Job	DatabaseJobExecutor.properties	Use the <code>jdbcDriversPath</code> keyword to specify the path to the JDBC drivers. Define the keyword so that it points to the JDBC jar files directory, for example: <pre>jdbcDriversPath=c:\\mydir\\jars\\jdbc</pre> The JDBC jar files must be located in the specified directory or its subdirectories. Ensure you have list permissions on the directory and its sub subdirectories.

Table 4. Configuration files for job types with advanced options (continued)

Job type	File name	Keyword
		 Note: For the MSSQL database, use version 4 of the JDBC drivers.
Java™ job type	JavaJobExecutor.properties	Use the <code>jarPath</code> keyword to specify the path to the directory where the jar files are stored. This includes all jar files stored in the specified directory and all sub directories.
J2EE job type	J2EEJobExecutorConfig.properties	For more information about the J2EE job type, see the topic about configuring to schedule J2EE jobs in the <i>IBM Workload Scheduler: Administration Guide</i> .

Customizing the SSL connection between IBM i agents and the Z controller when using your certificates

Customizing the SSL connection between IBM i agents and the Z controller when using your certificates.

About this task

The communication between IBM i agents and the Z controller to which they are registered to is, by default, in http. If you are using your own certificates, to enable an https communication you must customize the agent certificates and the configuration file by performing the following steps:

1. Generate a random file.
2. Generate a PEM file containing the private key of the agent and call it `ita_prv<suffix>.pem`. The IBM Workload Scheduler default PEM file is called **ita_prvtws.pem**.
3. Save the password of the agent private key in a stash file (`.sth` file).
4. Generate another PEM file and call it `ita_pub<suffix>.pem`. It must contain the certificate for the agent private key.
5. Create a copy of the file created in Step 4 on page 140 and call it `ita_cert<suffix>.pem`.
6. Generate another PEM file and call it `ita_ca_cert<suffix>.pem`. This file must contain the certificate of both the agent and the Z controller or the dynamic domain manager to which the agent is connected.
7. Open the `ita.ini` agent configuration file and set the values appropriate for your environment in the following properties:

```
password_file=<stash_file_fullpath>
random_file=<random_file_fullpath>
cert_label=<label_agent_private_key>
key_db_name=<suffix>
key_repository_dir=<directory_ita_*<suffix>.pem>
tcp_port=0
ssl_port=<ssl_port_value>
```

Where:

stash_file_fullpath

Specify the fully qualified path to the stash file that contains the agent private key password. This is the file that you created in the [Step 3 on page 140](#). The default value is `/opt/IBM/TWA_<TWS_user>/TWS/ITA/cpa/ita/cert/password.sth`.

random_file_fullpath

Specify the fully qualified path to the random file. This is the file that you created in the [Step 1 on page 140](#). The default value is `/opt/IBM/TWA_<TWS_user>/TWS/ITA/cpa/ita/cert/TWS.rnd`.

label_agent_private_key

Specify the label of the agent private key. The default is **client**.

suffix

Specify the suffix you used in the names of all the files that you generated. The default product value is **twc**.

directory_ita_*<suffix>.pem

Specify the directory that contains all the .pem files that you generated. The default directory is `/opt/IBM/TWA_<TWS_user>/TWS/ITA/cpa/ita/cert`.

tcp_port_value

Specify **0** as TCP/IP port value.

ssl_port_value

Specify the *tcp_port_value*. For example, if the TCP/IP port value was 31114, specify 31114.

8. Stop the IBM i agent by using the following command:

```
ShutDownLwa
```

9. Start the IBM i agent by using the following command:

```
StartUpLwa
```

After you complete the procedure, depending on the SSL storing certificate method you use, import the certificates in a RACF KEYRING or in a keystore created in the UNIX System services. Depending on the method you use refer either to the RACF or the Unix System services documentation.

Chapter 10. Monitoring and recovering jobs

This chapter contains the following sections.

IBM Z Workload Scheduler Agent job status notifications

Notifications about the change of status of running jobs are returned to the HTTP server subtask by the Tivoli® agent. [Table 5: Job status change notifications on page 142](#) shows the possible status changes for jobs running on IBM Z Workload Scheduler Agents.

Table 5. Job status change notifications

Notification	New operation status	Explanation
Submitted (POST executed)	S	Post executed (from local HTTP client)
Submitted	S	Agent launched the job
Canceled	E (+ CAN)	Job canceled by command
Executing	S	Job started execution
Failed_Execution	E (+ error code)	Job ended in error
Succeeded_Execution	C	Job completed successfully
Failed_Submission	E (+ OSUB)	Job submission failed
Unknown_Execution	E (+ CCUN)	Agent lost track of the job

Setting up and running automatic recovery for IBM Z Workload Scheduler Agent jobs

IBM Z Workload Scheduler supports the use of automatic recovery statements in IBM Z Workload Scheduler Agent scripts.

When you create or edit an IBM Z Workload Scheduler Agent script, start the lines where you specify the automatic recovery statements with `/** OPC`, `/**%OPC`, or `/**>OPC`. These lines are removed before the script is downloaded onto the IBM Z Workload Scheduler Agents.

You can run the following recovery actions on IBM Z Workload Scheduler Agent jobs:

- Restart the current occurrence at the failed operation, with or without JCL changes.
- Restart the current occurrence at another operation.
- Add occurrences of special recovery applications. Make the restart of the failed occurrence dependent on the completion of the recovery occurrences.
- Release a dependent occurrence.

Data set cleanup is not available.

Automatic-recovery-control statement for IBM Z Workload Scheduler Agent scripts

Each automatic-recovery-control statement describes an error situation and the recovery actions for it.

Use the following rules to create RECOVER statements:

- Each statement must begin in a new 80-byte logical record.

The symbols `/**%OPC` must appear in bytes 1 to 7 and be followed by at least one blank, where:

`/**%OPC RECOVER`

Identifies a RECOVER statement.

`/**%OPC`

Identifies a RECOVER continuation statement.

The automatic recovery function also inserts informational statements in the JCL:

`/** OPC`

Identifies a message statement.

`/**>OPC`

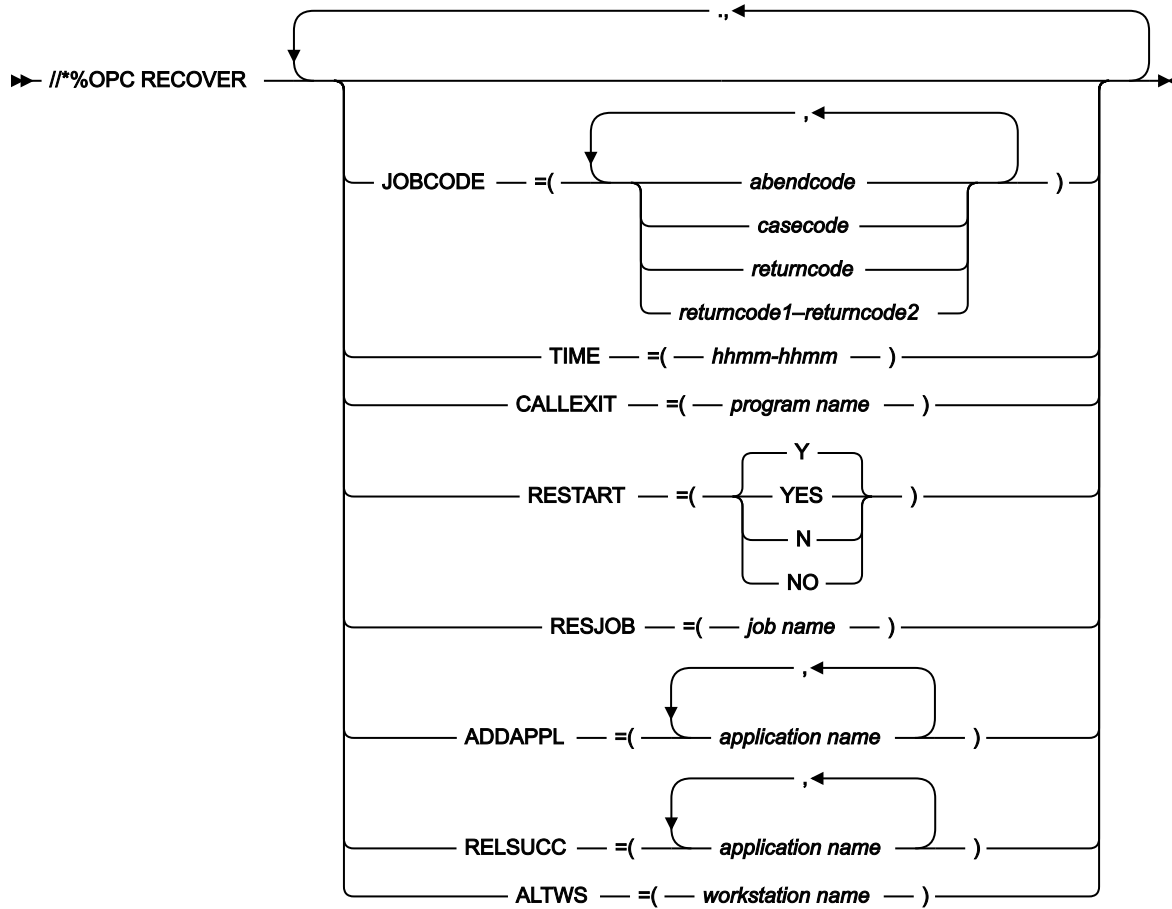
Identifies a comment statement.

- You cannot use variables anywhere in the RECOVER statement.
- The parameters are optional; you can code them in any sequence.
- Each parameter consists of a keyword followed by an equals sign and variable information.
- Parameters are separated by commas.
- You cannot code the same keyword more than once on the same statement.
- If you code only one parameter value, you do not need to enclose it in parentheses; for example, `JOBCODE=PCHK`.
- Bytes 72 to 80 are ignored by the automatic recovery function.
- When the total length of fields on a control statement exceeds 71 bytes, continue the statement using the following continuation conventions:
 - Interrupt the field after a complete or partial parameter, including the comma that follows it, before byte 72.
 - Code the identifying continuation characters `/**%OPC` followed by at least one blank in bytes 1 to 7 of the statement that follows.
 - Continue the interrupted operand in any position from bytes 9 to 16.
- Limit the RECOVER statement to one card (avoid continuation).

RECOVER statement syntax

The syntax of the RECOVER statement follows. These are the only RECOVER parameters that apply to the IBM Z Workload Scheduler Agent workstation.

For a complete list of the RECOVER parameters, see *Managing the Workload*.



Statement parameters

All the parameters for the RECOVER statement are optional. For IBM Z Workload Scheduler Agent jobs, the available RECOVER statement categories are Selection and Recovery action.

Selection parameters

Selection parameters for error situations managed by the recover statement.

These selection parameters specify the error situations that the RECOVER statement manages:

JOBCODE

Restricts the RECOVER statement to be valid only for those job completion codes and return codes specified.

TIME

Restricts the RECOVER statement to be valid only in the time range specified.



Note: Review the EXCLUDECC and EXCLUDERC parameters of the AROPTS statement, which specify codes for which no automatic recovery is done. For information about AROPTS, see *Customization and Tuning*.

Recovery action parameters

The following recovery action parameters specify the actions to take for recovery:

RESTART

Specifies if the occurrence is to be restarted.

RESJOB

Specifies the name of the job from which the occurrence must be rerun.

ADDAPPL

Specifies an application or a list of applications to be added as occurrences in the current plan.

RELSUCC

Specifies the application ID of a successor occurrence, or a list of IDs.

ALTWS

Specifies the name of an alternate workstation on which to run the operation.

Selection parameters

When a job ends in error, this information is available:

- Job name
- Abend code, if job abended
- Return code, if job did not abend

The selection parameters use this information:

JOBCODE

The code can be an abend code, an error code set by IBM Z Workload Scheduler or JCC, a case code, a return code, or a return code range. The values are those given as the error code on the Handling Operations Ended in Error panel in the Modify Current® Plan panel.

The JOBCODE keyword values are:

Sxxx

Specifies a system abend code.

Uxxx

Specifies a user abend code.

xxxx

Specifies a case code or an error code set by IBM Z Workload Scheduler, either directly or by using the job completion checker.

n

Specifies a return code.

x-y

Specifies a return code range, where *x* and *y* represent positive decimal values.

The codes specified in the JOBCODE parameter are tested against the job code as set by IBM Z Workload Scheduler when the job ends.

You can also specify abend codes in generic form. Therefore, an asterisk (*) can represent any character, or any group of characters, in those positions where it is placed. A code can contain more than one asterisk only if each asterisk is separated from the next by another character.



Note: The JOBCODE=* notation covers all possible user and system abend codes; it does not cover return codes. To cover all possible return codes, specify a return code range in the form:

```
JOBCODE=x-y
```

where *x* and *y* represent positive decimal values.

A return code cannot be greater than 4095. To specify a range of values, set *x* to the lower value and *y* to the higher value. To specify all return codes greater than or equal to a certain value, set *x* to that value and *y* to 4095.

TIME

The time is specified in the form *hhmm*, where *hh* is the hour from 00 to 24, and *mm* is the minute from 00 to 60. This is the time when the recovery is automatic.

For example:

TIME=0700-1600

No recovery actions occur between 4 p.m. and 7 a.m.

TIME=2200-0800

IBM Z Workload Scheduler does automatic recovery only between 22.00 and 8.00.

TIME=0000-2400

Recovery can be automatic at any time.

TIME=0000-0000

Recovery is not started unless there is manual intervention.

The recovery actions for a job that remains in the ended-in-error list can be manually started up later. Such requests override any TIME value specified.

Default: The recovery specification is for the time range specified by the STARTIME and ENDTIME keywords of the AROPTS automatic recovery options statement. For more details, see *Customization and Tuning*.

Action parameters

The following parameters specify the action that IBM Z Workload Scheduler must take when the recovery statement is invoked.

RESTART

RESTART=Y causes the job to be rerun, either from the failing operation or, if you specified RESJOB, from an earlier operation within the occurrence.

RESTART=N prevents the restart. It can be used with ADDAPPL when the recovery actions are to be done by a separate application. It can also be used to select cases for which no recovery should be performed or when testing the recovery procedure. The operation remains in ended-in-error status.

Default: RESTART=Y.

RESJOB

The RESJOB parameter handles failures at an occurrence level, not within the failing job itself.

The occurrence of the application is rerun from the first preceding computer workstation operation, whose name matches the job name specified in the RESJOB parameter. If the job name specified cannot be found in a computer operation preceding the failed operation in the same occurrence, no automatic recovery occurs and the job remains in the error handling list with an extended status code indicating an automatic recovery error.

The indicated operation must be a predecessor to the failed operation or be the failed operation itself.



Note: External successors cannot be handled automatically. Therefore the set of operations selected for rerun that can be completed at failure time must not have any external successors.

The rerun operation must be defined on an automatically reporting computer workstation.

The parameter is ignored if RESTART=N is specified.

Default: Rerun from the failed operation.

ADDAPPL

Here you specify a list of applications. Occurrences of these applications are added to the current plan if this RECOVER statement is invoked.

- If RESTART=N is specified, the applications are independent of the failed occurrence.
- If RESTART=Y is specified, the recovery applications are added to the current plan as predecessors to the failing operation or, if RESJOB is specified, to the operation where restart is being attempted.

Added applications are independent of each other. A maximum of 40 application occurrences can be added.

For example, suppose a database update fails. A rerun of the failed job is necessary but must be postponed to a later time. However, a database restore job must be run to repair the database before the online users require the database. This recovery situation can be specified as:

```
//*%OPC RECOVER JOBCODE=CHK,RESTART=N,ADDAPPL=A301RORG
```

This means that, on case code SCHK, do not rerun the failing operation. However, add an occurrence of the application A301RORG to the current plan, without a dependency to the failing operation and leave the failing operation in ended-in-error status.

Default: No application occurrences are added.



Note:

1. When automatic recovery adds an occurrence to the current plan, input arrival and deadline times are not taken from the application description. Instead the occurrence is given an input arrival of the time the add is run, according to the time on the z/OS® system where the controller is started and the deadline is set for 8 hours after input arrival. If an occurrence of that application already exists with this input arrival time, then one minute is added to the time until a time is reached when the occurrence can be included. If the added occurrence includes time-dependent operations with specific input arrival times, then the operations are started at the specified time.
2. An occurrence that has been added to the current plan by automatic job recovery does not become the predecessor to an occurrence that is added later by daily planning, even if normal dependency criteria are met.

RELSUCC

This parameter specifies which external successors to the failing operation are allowed to run even if their predecessor operation has ended in error.

The external successors to the failing operation are checked, and the dependencies between the failed operation and the specified successors are deleted at recovery time.

The effect is that this predecessor (the failed operation) is reported as complete to the external successor, and the successor-predecessor chaining is removed. The external successor becomes ready if its other predecessors are completed. The dependency does not exist when the failed occurrence is rerun.

Even if one successor is released, other successors might be waiting for the failed occurrence to complete. These might be successors not yet in the current plan. Assume that W is a weekly application and D is a daily application that is dependent on W. If W fails and there is a RECOVER statement causing the release of that day's D, the occurrence of D the next day also waits for W to complete, but without any automatic release.

You can specify a maximum of 40 application IDs.

Default: None.

ALTWS

Specifies the name of an alternate workstation on which to run the operation. The ALTWS parameter overrides the alternate workstation defined in the workstation description. You can use this parameter, for example, with the TIME parameter to specify alternate workstations for an operation, depending on the time of day.

Default: None.

Chapter 11. Running a script when a job completes

In many scenarios, when a job completes, you might want to run one or more actions, by using the information related to the job completion. For this purpose, you can write a script file and store it in a directory of the agent file system. The script is run every time that a job completes, either successfully or unsuccessfully. The script runs with the same credentials as the agent user that is running the job.



Note: The agent user must be authorized to access the script file and its directory.

To provide IBM Workload Scheduler with the path of the script file, you must modify the `JobManager.ini` file as follows:

1. Locate the `JobManager.ini` file on the local agent instance where the script will run. The file is located in the `TWA_home/TWS/ITA/cpa/config` directory on the agent.
2. In the `[NativeJobLauncher]` section of the file, define the value of the **PostJobExecScriptPathName** property, with the fully qualified path of the script file that you want to run when a job completes.
3. Save the changes to the file.

If you do not specify any file path or the script file doesn't exist, no action is taken when the job completes. For details about customizing the **PostJobExecScriptPathName** property, see *Administration Guide*.

The following job variables can be used in the script:

- `JOB_ID`
- `JOB_ALIAS`
- `JOB_SPOOL_DIR`
- `JOB_STATUS`
- `JOB_RETURN_CODE`
- `JOB_DURATION`
- `JOB_START_TIME`
- `JOB_END_TIME`

The script is run for any of the following job final statuses:

- `SUCCEEDED_EXECUTION`
- `UNKNOWN`
- `CANCELLED`
- `FAILED_EXECUTION`

In the `JobManager_message.log`, you are notified via a message if the job started successfully, or if any error prevented the job from starting. To analyze the output of the script execution, you can check the `out.log` file in the `post_script` subdirectory of the job `SpoolDir`.

Chapter 12. Viewing the job logs from the host

An overview on how to view or retrieve the job logs issued in the z-centric environment and how to attach a header with job run information on top of each log.

There are two ways of retrieving the job logs:

- You can manually request to browse the log of a specific job (on demand) or have automatically displayed the logs of jobs that ended in error. To browse the log you can use any of the available user interfaces (ISPF dialogs, Dynamic Workload Console, or programming interfaces).

To specify the job log retrieval policy (on demand, or automatic for ended-in-error), define the `JOBLOGRETRIEVAL` keyword in the `HTTPOPTS` initialization statement. See *IBM Z Workload Scheduler: Customization and Tuning*.

- You can activate the [Output collector on page 153](#) started task to have the logs of all jobs, independently of their outcome, sent to the JES spool to be picked up by an external output management product.

On all the job logs originated from the z-centric environment you can customize IBM Z Workload Scheduler so that a [header on page 151](#) is added automatically at the top of the job log with the information necessary to classify the output (occurrence name, occurrence IA, job name, workstation name, operation number, start time, end time) and information about the run (process ID, return code, duration, status, hostname).



Remember: to have the job log header mechanism functioning for workload run on dynamic agents, the dynamic agents must be running at least version 8.6.0.1 of IBM Workload Scheduler.

Customizing the job log header

A sample job log header template is downloaded on IBM Z Workload Scheduler at installation time. You can customize it to fit it to your preferences.

When you customize the `JLOGHDRTEMPL` keyword of the `HTTPOPTS` initialization statement, you command the system to add a header containing run-related information to every job log released in the z-centric environment.

The header format can be personalized. The information displayed in the header is added depending on how you customize the header template sample that comes with the installation of IBM Z Workload Scheduler and which predefined variables that describe the job and its run are featured in the header.

When you create the sample job JCL with the `EQQJOBS` installation aid, the following job log header template sample is added to the samples library that you specified in the `EQQJOBS` panels:

```
=====
=                                JOBLOG  HEADER
=
= Occurrence name : ${occname}
= Occurrence IA   : ${IAYYYY}/${IAMM}/${IADD}  ${IAhh}:${Iamm}
= Job name       : ${jobname}
= Work station   : ${wsname}
= Operation number: ${opernum}
```

```

= Start time      : ${YYYYY}/${SMM}/${SDD} ${Shh}:${Smm}
= End time       : ${EYYYY}/${EMM}/${EDD} ${Ehh}:${Emm}
= Process ID     : ${processid}
= Duration       : ${duration}
= Status        : ${status}
= Return code    : ${returncode}
= Hostname      : ${hostname}
=
=====

```

You can then edit and copy the sample to a PARM library that must be pointed to by the JLOGHDRTEMPL keyword of the HTTPOPTS initialization statement.

The properties of the predefined variables provided in the sample are described in the following table. When you customize the sample, you can select the variables that you deem useful or retain them all.

Table 6. The predefined variables of the sample job log header template.

Variable	Name	Length	Value is...
Occurrence name	occname	16	sent at job submission
Occurrence input arrival time	IAYYYY IAMM IADD IAhh IAmm		sent at job submission
Job name	jobname	8	sent at job submission
Workstation	wsname	4	sent at job submission
Operation number	opernum	2	sent at job submission
Start time	SYYYY SMM SDD Shh Smm		local on agent
End time	EYYYY EMM EDD Ehh Emm		local on agent
Process ID	processid		local on agent
Duration	duration		local on agent
Status	status		local on agent
Return code	returncode		local on agent; NOERROR actions are ignored and not truncated to 4 characters
Hostname	hostname		local on agent; hostname of the agent where the job ran

The syntax for the job log header variables is consistent with the syntax used for variable substitution in the z-centric environment, that is:

```

${variable_name}

```




Important: The template sample is in the IBM-37 codepage. The brace characters ({ and }) are mapped differently depending on the codepage. If you use a different codepage to view the template, you are very likely to need to correct the variables syntax.

Only simple variable substitution is allowed, no compound variables are supported. If the variable name is incorrect, the variable is left unresolved.

You can also use variables to compose the run dates. The dates are based on the local time of the controller.

Collecting job logs with Output collector

An overview on the mechanism that automatically retrieves the logs of jobs and dynamic jobs run on IBM Z Workload Scheduler Agents and copies them to the JES spool so that they can be processed by an external output management product.

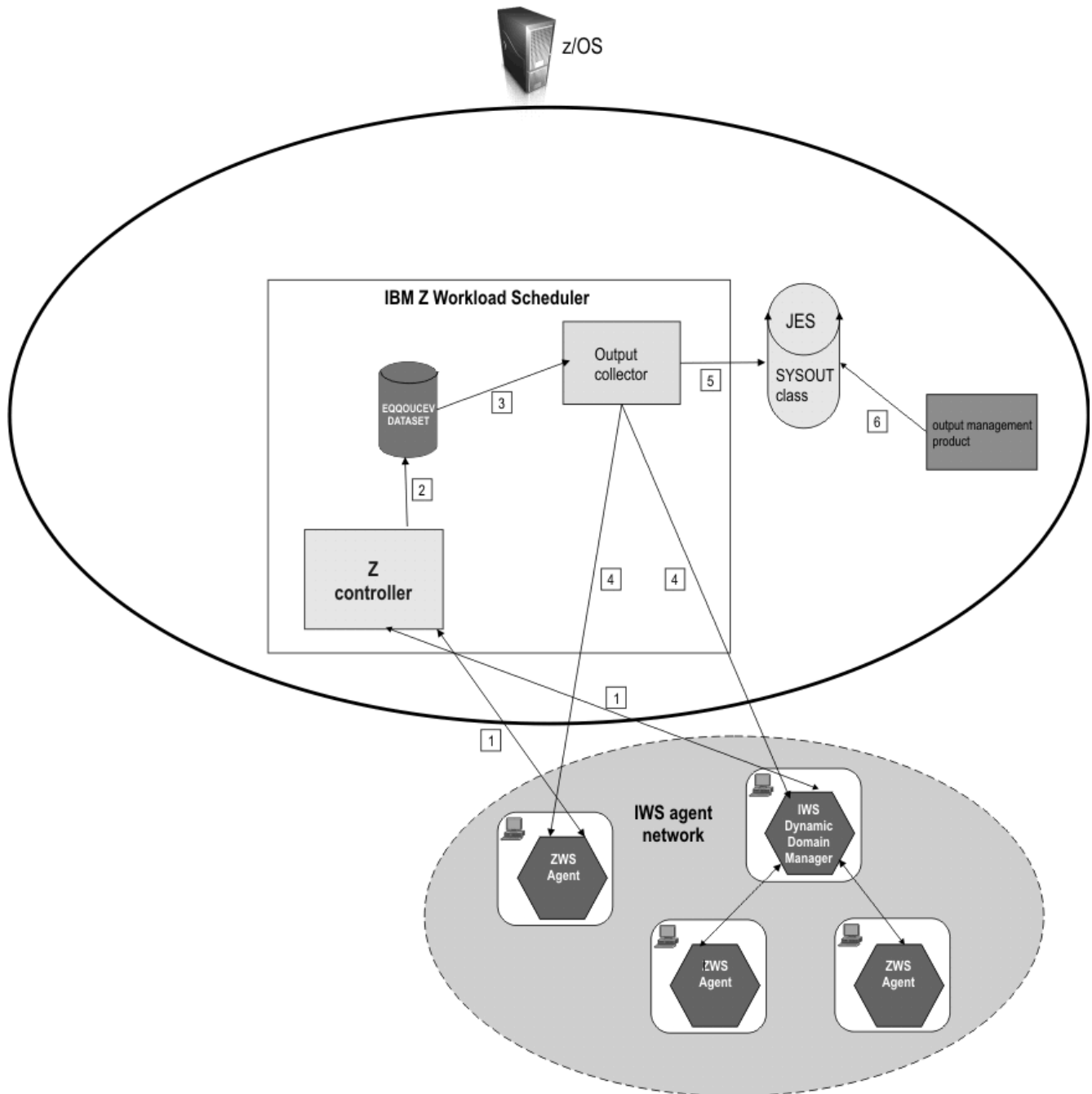
The standard log management process for jobs running in the z-centric environment is to be requested manually, with an exception for the logs of jobs that ended in error that can be received semi-automatically. In addition to this, or as an alternative, you can activate the Output collector started task, which automatically retrieves the log of every job run in the z-centric environment and sends it to the JES spool so that it can be processed by an external output management product.

Every time a job or a dynamic job completes or terminates on an IBM Z Workload Scheduler Agent, the Output collector started task receives an event from the IBM Z Workload Scheduler controller (which manages all communication with the agents and the dynamic domain managers in the z-centric environment). The event contains the information necessary for Output collector to identify the job and where it run. The output collector then retrieves the job log from the agent, or the dynamic domain manager if the job is dynamic, and copies it to a SYSOUT in JES (using a specific SYSOUT class) to make it available to an output management product.

Activation of this feature is optional. If you activate it, it automatically collects the logs of all jobs run in the z-centric environment, regardless of whether they complete successfully or terminate in error. If you do not activate it, you can still configure your system to either request logs manually or to receive those of jobs ended in error.

The following figure describes the job log retrieval process with Output collector.

Figure 12. Job log retrieval with Output collector.



Where:

1. The controller submits and tracks jobs and gets the job logs upon request.
2. Every time a job completes or terminates, the controller writes an event in the EQQOUCV data set. The event contains identification of the job and the name of the agent that ran it.
3. Output collector reads the event in EQQOUCV.
4. Based on the information found in the event record, Output collector retrieves the job log from the agent (or the dynamic domain manager if the job is dynamic).

5. Output collector copies the job log to a SYSOUT in JES using a specific SYSOUT class.
6. The external output management product can get the job log for analysis, accounting, and other operations.

The controller and Output collector use the EQQOUCV and EQQOUCKP data sets to share the information concerning completed or terminated jobs. The communication process is based upon events. Every time a job completes or terminates, the controller queues an event for Output collector with the information necessary to identify the job and the agent that run it in a new record in EQQOUCV. Output collector reads the record, checkpoints it in EQQOUCKP, dispatches it to the proper thread, and marks the event as processed moving to the next-to-read index in the data set header. EQQOUCKP is used to checkpoint the incoming requests to prevent their loss in case of unplanned closures.

To write a job log in the JES spool, Output collector allocates a SYSOUT data set with DDNAME equal to the job name, writes the job log in it, and then closes it. This implies that all the SYSOUTS will have the started task job name and job id and will differ only in the DDNAME and job log header.



Attention: The following event may take place while Output collector is processing more job instances with the same name at the same time, even if on different agents: if Output collector is retrieving a particularly sizable log from an agent and the agent goes offline, Output collector stops the retrieval until the link with the agent is established again. As soon as the agent is online again, Output collector resumes retrieving the log. The anomaly is that, if in this lapse of time Output collector is requested to retrieve any other logs with the same name, perhaps from other agents (remember that all logs are named after their jobs), the blocked state is extended also to these requests. As soon as conditions are set for resuming the retrieval of the first log, the other logs are also processed. This does not apply to other job logs having different names: Output collector in this time span continues to process them normally.

Activating and configuring Output collector

This is a summary about how to activate and configure the Output collector started task.

To activate the Output collector, run the EQQJOBS installation aid and go to panel EQQJOBSC of option 1 - Create sample job JCL. In the EQQJOBSC panel you specify that Output collector will be running and you enter the names of the SYSOUT class and of the WRITER task that Output collector uses to copy the logs to the JES spool. This creates the sample started task procedure and initial parameters for running Output collector and allocates the data sets (EQQOUCV and EQQOUCKP) used for receiving job log information from the controller. See *IBM Z Workload Scheduler: Planning and Installation* for a detailed description.

To fully configure Output collector, you must act on the following initialization statements in the controller:

HTTPOPTS

Add the following keywords:

OUTPUTCOLLECTOR

Specify that the feature will be running.

JLOGHDRTEMPL

Specify the name of the EQPPARM library member that contains the job log header. This keyword is not exclusive to the Output collector, but can be used for all the job logs produced in the z-centric environment.

These keywords are independent of the **JOBLOGRETRIEVAL** keyword, which enables users to request job logs manually through the usual IBM Z Workload Scheduler interfaces.

ROUTOPTS

To work with the Output collector task, it is required that at least one IBM Z Workload Scheduler Agent (z-centric) destination is defined in the keyword **HTTP** or **HTTPS**.

OPCOPTS

Optionally, add the **OUTCOL** keyword to specify the name of the Output collector started task to the controller. In a sysplex environment this enables the standby controller to continue the interaction with Output collector.

OUCOPTS

Add this initialization statement and its keywords to define:

- The names of the JES SYSOUT class and of the WRITER task used by Output collector (if they were not defined with EQQJOBS).
- The name of the EQPPARM library member that contains the OPCOPTS, ROUTOPTS, and HTTPOPTS statements.
- The number of threads (range) that Output collector can open.
- The maximum number of job logs that Output collector can retrieve concurrently from an agent.
- The name of the controller for which Output collector is started.

In a sysplex configuration the Output collector started task must reside in the same image where the controller is.

For a detailed description of the HTTPOPTS, OPCOPTS, and OUCOPTS statements, see *IBM Z Workload Scheduler: Customization and Tuning*.

Refreshing destinations from Output collector

You can use the MODIFY operator commands to display and refresh HTTP/HTTPS destinations also from Output collector.

You can refresh or display IBM Z Workload Scheduler Agent [destinations on page 39](#) while Output collector is running, without having to stop and restart Output collector to make the changes effective. To do so, run the TSO commands:

- `/F procname, DSPDEST` to display
- `/F procname, RFRDEST` to add, modify, or delete

where in *procname* you can enter the JCL procedure name for Output collector instead of the one for the controller.

If you run `/F procname, RFRDEST` with the *procname* for the controller, the refresh operation is run on the controller and propagated also to Output collector.

Be aware that any changes you make using the procedure name for the controller apply also to Output collector, while they are not extended to the controller if you use the procedure name for Output collector.

Chapter 13. Troubleshooting

Troubleshooting agent problems

The following problems could be encountered:

On AIX operating systems the concurrent submission of one hundred or more jobs on the same agent can result in a core dump or in a resource temporarily unavailable message

On AIX operating systems, the concurrent submission of one hundred or more jobs on the same agent can result in a memory dump or in a resource temporarily unavailable message

On AIX operating systems if you submit concurrently one hundred or more jobs on the same agent you can receive a core memory dump or the following message:

```
resource temporarily unavailable
```

Cause and solution:

This problem is due to insufficient memory and the process number per user allocated to run the jobs concurrently. To solve this problem, verify the value of the following configuration settings and change them as follows:

Ulimit settings

The submission of a significant number of Java jobs requires a large amount of memory. Change the value for data, stack, and memory limits according to the number of jobs you want to submit. The submission of a significant number of native jobs requires a high number of file descriptors and processes. Change the values for `nofiles` and `processes` according to the number of jobs you want to submit. The following example gives possible setting values to submit 100 jobs concurrently:

```
time(seconds)      unlimited
file(blocks)       2097151
data(kbytes)       131072
stack(kbytes)      32768
memory(kbytes)     32768
coredump(blocks)   2097151
nofiles(descriptors) 4000
threads(per process) unlimited
processes(per user) unlimited
```

Process number per user

To submit a high number of jobs concurrently you must have a high value for the **maxuproc** setting. Use the **lsattr -E -l sys0 -a maxuproc** command to verify the number of concurrent processes that a user can create. Use the **chdev -l sys0 -a maxuproc=<value>** command to change the value for the **maxuproc** setting. For example, to submit 100 jobs concurrently use the following command:

```
chdev -l sys0 -a maxuproc=500
```

Appendix A. Configuring to schedule job types with advanced options

About this task

In addition to defining job types with advanced options using the Dynamic Workload Console or the [JOBREC on page 75](#) statement, you can use the related configuration files. The options you define in the configuration files apply to all job types with advanced options of the same type. You can override these options when defining the job using the Dynamic Workload Console or the [JOBREC on page 75](#) statement.

Configuration files are available on each dynamic agent in `TWA_home/TWS/JavaExt/cfg` for the following job types with advanced options:

Table 7. Configuration files for job types with advanced options

Job type	File name	Keyword
Database job type MSSQL Job	DatabaseJobExecutor.properties	Use the <code>jdbcDriversPath</code> keyword to specify the path to the JDBC drivers. Define the keyword so that it points to the JDBC jar files directory, for example: <pre>jdbcDriversPath=c:\mydir\jars\jdbc</pre> The JDBC jar files must be located in the specified directory or its subdirectories. Ensure you have list permission for the directory and its sub subdirectories.  Note: The MSSQL database requires version 4 of the JDBC drivers for both Database and MSSQL Job job types.
Java™	JavaJobExecutor.properties	Use the <code>jarPath</code> keyword to specify the path to the directory where the jar files are stored. This includes all jar files stored in the specified directory and all sub directories.

Logging information about job types with advanced options

You can use the `logging.properties` file to configure the logging process for job types with advanced options, with the exception of the Executable and Access Method job types.

The `logging.properties` file is located on the IBM Z Workload Scheduler Agent, under `TWA_home/TWS/JavaExt/cfg/logging.properties`.

Following installation, this file is as follows:

```
# Specify the handlers to create in the root logger
# (all loggers are children of the root logger)
```

```
# The following creates two handlers
handlers = java.util.logging.ConsoleHandler, java.util.logging.FileHandler

# Set the default logging level for the root logger
.level = INFO

# Set the default logging level for new ConsoleHandler instances
java.util.logging.ConsoleHandler.level = INFO

# Set the default logging level for new FileHandler instances
java.util.logging.FileHandler.level
    = ALL
java.util.logging.FileHandler.pattern
    = C:\TWA_home\TWS\JavaExt\logs\javaExecutor%g.log
java.util.logging.FileHandler.limit
    = 10000000
java.util.logging.FileHandler.count
    = 10

# Set the default formatter for new ConsoleHandler instances
java.util.logging.ConsoleHandler.formatter = java.util.logging.
SimpleFormatter
java.util.logging.FileHandler.formatter = java.util.logging.
SimpleFormatter

# Set the default logging level for the logger named com.mycompany
com.ibm.scheduling = INFO
```

You can customize:

- The logging level (from INFO to WARNING, ERROR, or ALL) in the following keywords:

.level

Defines the logging level for the internal logger.

com.ibm.scheduling

Defines the logging level for the job types with advanced options. To log information about job types with advanced options, set this keyword to ALL.

- The path where the logs are written, specified by the following keyword:

java.util.logging.FileHandler.pattern

Appendix B. Managing return codes

By default, all nonzero error codes are considered as an error. Optionally, you can define the error codes that, for job-tracking purposes, are not considered as errors. To do this, set the following parameters:

NOERROR

Defines a list of error codes that, for job-tracking purposes, are treated as normal completion codes. When setting this value on an IBM Z Workload Scheduler Agent, beware that the maximum length allowed for a return code is 4 digits. If the IBM Z Workload Scheduler Agent returns an error code higher than 9999, it is truncated at the fourth rightmost digit. For example, 12345 is considered as 2345.

For negative error codes, specify a five-character sequence starting with the minus (-) symbol, for example -0008.

ERRRES

Defines a list of error codes that, for job-tracking purposes, result in an automatic reset of an operation. The operation is reset to status A (arriving) and contains the message Error, automatically reset in its operation details panel. An error code can be either of the following:

- 4-digit job or started-task return code (*nnnn*)
- System abend code (*Sxxx*)
- User abend code (*Uxxx*)
- IBM Z Workload Scheduler-defined code

The HIGHRC parameter is supported for positive RC on IBM Z Workload Scheduler Agents and agents. Jobs ending with negative RC are always considered in error. The negative RC are handled through NOERROR. For a detailed description of NOERROR and ERRRES, see *IBM Z Workload Scheduler: Customization and Tuning*.

The following is a list of the return codes for each job type with advanced options:

Database jobs:

```
RC = 0 -> Job completed successfully
RC = -1 -> SQL statement was run with an exit code different from 1
RC = -2 -> MSSQL Job error
RC = -3 -> SQL statement did not run because of an error in the statement
```

File transfer jobs:

```
RC = 0 -> The file transfer completed successfully

RC = -1 -> The file transfer is not performed. The job fails with the following
error code: AWKFTE007E
```

Explanation: An error occurred during the file transfer operation

Possible reasons: Remote file not found or permission denied

```
RC = -2 -> The file transfer is not performed. The job fails with the following
error code: AWKFTE020E
```

Explanation: Only for SSH or Windows protocols. An error was returned

while attempting to convert the code page

Possible reasons: For SSH or Windows protocols, the code page is automatically detected and converted. In this case, there is an error in the code page of the file to be transferred, which is not compliant with the code page of the local system

RC = -3 -> The file transfer is not performed. The job fails with the following error code: AWKFTE015E

Explanation: An error occurred during the file transfer operation

Possible reasons: Local file is not found

RC = -4 -> The file transfer is performed with the default code page. The job fails with the following error code: AWKFTE023E

Explanation: The specified codepage conversion has not been performed. File transfer has been performed with default code pages

Possible reasons: The specified code page is not available

IBM i jobs:

Return code = user return code when retrieved

Return code = 0 -> job completed successfully

Return code > -1 -> job completed unsuccessfully

Web services jobs:

RC = 0 -> Job completed successfully

RC = -1 -> The server hostname contained in the Web Service URL is unknown

RC = -2 -> Web Service invocation error

Appendix C. Modifying the workstation from fault-tolerant agent to z-centric agent

About this task

The fault-tolerant agent workstations are no longer supported and must be modify to IBM Z Workload Scheduler Agent (z-centric) workstations, as described in the following procedure.



Note:

- For jobs whose [JOBREC on page 75](#) includes an RCONDSUC parameter, see [Managing return codes on page 161](#) for information about how to perform return code management for jobs scheduled on an IBM Z Workload Scheduler Agent.
- To modify a fault-tolerant agent workstation that uses non-centralized job scripts to an IBM Z Workload Scheduler Agent workstation, you can follow the same procedure. Additionally, you must convert the information contained in the EQQSCLIB data set into information for the JOBLIB data set, by using the appropriate [JOBREC on page 75](#) statement keywords.

On the Z controller:

1. Define the IBM Z Workload Scheduler Agent destination by setting the ROUTOPTS statement. For details, refer to [Defining destinations with the ROUTOPTS statement on page 39](#).
2. Optionally, customize the HTTP connection with the HTTPOPTS statement. For details, refer to [Customizing the HTTP connection with the HTTPOPTS statement on page 42](#).
3. From the MODIFYING GENERAL INFORMATION ABOUT A WORK STATION panel, modify the workstation database definition by setting Z-CENTRIC AGENT to ☒. For details, refer to [Modifying on page 56](#).
4. To modify one fault-tolerant agent at a time, comment out the CPUREC statement related to the fault-tolerant workstation you are modifying.

To modify all the fault-tolerant agents at once, delete the TPLGYPRM parameter in the BATCHOPT statement.

5. For your changes to be effective, run a DAILY PLAN batch job.

Results

Message EQQ0371I is issued in EQQMLOG:

```
Z-CENTRIC AGENT INFO HAS BEEN CHANGED FOR WORKSTATION WSNNAME
```

If on the workstation modified there were operations with status S (started), the status is changed to I (interrupted) and message EQQ0372W is issued in EQQMLOG:

```
STATUS OF OPERATION ADID IA OPNO HAS BEEN CHANGED FROM STARTED TO INTERRUPTED  
AS A CONSEQUENCE OF WORKSTATION WSNNAME CHANGE
```

The DP batch job ends with RC=4.

After you have modified all the fault-tolerant agents to z-centric agents:

1. Delete any statements related to the end-to-end with fault tolerance environment from the Z controller and server.
2. From the Z controller started task, delete the following DD names:
 - EQQTWSIN
 - EQQTWSOU
 - EQQSCLIB
 - EQQTWSCS
3. From the server started task, delete the following DD names:
 - EQQTWSIN
 - EQQTWSOU
 - EQQTWSCS

Appendix D. Integrating with SAP and PeopleSoft

Integrating with SAP and PeopleSoft

Access methods are used to extend the job scheduling functions of IBM Workload Scheduler to other systems and applications. They run on:

Extended agents (applies only to IBM Workload Scheduler)

They are logical workstations related to an access method hosted by a physical IBM Workload Scheduler workstation (not another extended agent). More than one extended agent workstation can be hosted by the same IBM Workload Scheduler workstation and use the same access method. The extended agent runs on fault-tolerant agents defined using a standard IBM Workload Scheduler workstation definition, which gives the extended agent a name and identifies the access method. The access method is a program that is run by the hosting workstation whenever IBM Workload Scheduler submits a job to an external system.

Jobs are defined for an extended agent in the same manner as for other IBM Workload Scheduler workstations, except that job attributes are dictated by the external system or application.

Information about job running execution is sent to IBM Workload Scheduler from an extended agent using the job `stdlist` file. A method options file can specify alternate logins to launch jobs and check *opens* file dependencies. For more information, see the *User's Guide and Reference*.

A physical workstation can host a maximum of 255 extended agents.

dynamic agents and IBM Z Workload Scheduler agents (z-centric)

They communicate with external systems to start the job and return the status of the job. To run access methods on external applications using dynamic agents, you define a job of type **access method**.

Access methods are available on the following systems and applications.

- SAP
- z/OS
- Custom methods
- unixssh
- unixrsh

The UNIX™ access methods included with IBM Workload Scheduler, are described in the related section in *Administration Guide*.

If you are working with dynamic agents, for information about defining IBM Workload Scheduler workstations, see the section that explains how to define workstations in the database in *User's Guide and Reference*. For information about writing access methods, see the section about the access method interface in *User's Guide and Reference*.

More information about access methods is found in *Scheduling Job Integrations with IBM Workload Automation*.

Access method interface

The interface between IBM Workload Scheduler and an access method consists of information passed to the method on the command line, and of messages returned to IBM Workload Scheduler in **stdout**.

Method command line syntax

The IBM Workload Scheduler host runs an access method using the following command line syntax:

```
methodname -t task options -- taskstring
```

where:

methodname

Specifies the file name of the access method. All access methods must be stored in the directory:

```
TWS_home/methods
```

-t task

Specifies the task to be performed, where *task* is one of the following:

LJ

Launches a job.

MJ

Manages a previously launched job. Use this option to resynchronize if a prior **LJ** task ended unexpectedly.

CF

Extended agents only. Checks the availability of a file. Use this option to check file `opens` dependencies.

GS

Extended agents only. Gets the status of a job. Use this option to check job `follows` dependencies.

options

Specifies the options associated with the task. See [Task options on page 166](#) for more information.

taskstring

A string of up to 255 characters associated with the task. See [Task options on page 166](#).

Task options

The task options are listed in [Table 8: Method command task options on page 167](#). An X means that the option is valid for the task.

Table 8. Method command task options

Task	-c	-n	-p	-r	-s	-d	-l	-o	-j	-q	-w	-S	Task String
LJ	X	X	X	X	X	X	X	X	X			X	<i>ljstring</i>
MJ	X	X	X	X	X	X	X	X	X				<i>mjstring</i>
CF	X	X	X							X			<i>cfstring</i>
GS	X	X	X	X		X					X		<i>gsstring</i>

-c *xagent,host,master*

Specifies the names of the agent, the host, and the master domain manager separated by commas.

-n *nodename*

Specifies the node name of the computer associated with the agent, if any. This is defined in the extended agent's workstation definition **Node** field.

-p *portnumber*

Specifies the TCP/IP port number associated with the agent, if any. This is defined in the agent workstation definition **TCP Address** field.

-r *currentrun,specificrun*

Specifies the current run number of IBM Workload Scheduler and the specific run number associated with the job separated by a comma. The current and specific run numbers might be different if the job was carried forward from an earlier run.

-s *jstream*

Specifies the name of the job's job stream.

-d *scheddate,epoch*

Specifies the job stream date (*yymmdd*) and the epoch equivalent, separated by a comma.

-l *user*

Specifies the job's user name. This is defined in the job definition **Logon** field.

-o *stdlist*

Specifies the full path name of the job's standard list file. Any output from the job must be written to this file.

-j *jobname,id*

Specifies the job's name and the unique identifier assigned by IBM Workload Scheduler, separated by a comma. The name is defined in the job definition **Job Name** field.

-q *qualifier*

Specifies the qualifier to be used in a test command issued by the method against the file.

-w timeout

Specifies the amount of time, in seconds, that IBM Workload Scheduler waits to get a reply on an external job before sending a SIGTERM signal to the access method. The default is 300.

-S new name

Specifies that the job is rerun using this name in place of the original job name. Within a job script, you can use the `jobinfo` command to return the job name and run the script differently for each iteration.

-- ljstring

Used with the **LJ** task. The string from the **Script File** or **Command** field of the job definition.

-- mjstring

Used with the **MJ** task. The information provided to the IBM Workload Scheduler by the method in a message indicating a job state change **%CJ** (for additional details on messages indicating job state change, see [Method response messages on page 168](#)) following to an **LJ** task. Usually, this identifies the job that was launched. For example, a UNIX® method can provide the process identification (PID) of the job it launched, which is then sent by the IBM Workload Scheduler as part of an **MJ** task.

-- cfstring

Used with the **CF** task. For a file opens dependency, the string from the **Opens Files** field of the job stream definition.

-- gsstring

Used with the **GS** task. Specifies the job whose status is checked. The format is as follows:

followsjob[jobid]

where:

followsjob

The string from the **Follows Sched/Job** list of the job stream definition.

jobid

An optional job identifier received by IBM Workload Scheduler in a **%CJ** response to a previous **GS** task.

Method response messages

Methods return information to IBM Workload Scheduler in messages written to **stdout**. Each line starting with a percent sign (%) and ending with a new line is interpreted as a message. The messages have the following format:

%CJ *state [mjstring | jobid]*

%JS *[cputime]*

%RC *rc*

%UT *[errormessage]*

where:

CJ

Changes the job state.

state

The state to which the job is changed. All IBM Workload Scheduler job states are valid except HOLD and READY. For the **GS** task, the following states are also valid:

ERROR

An error occurred.

EXTRN

Status is unknown.

mjstring

A string of up to 255 characters that IBM Workload Scheduler will include in any **MJ** task associated with the job.

jobid

A string of up to 64 characters that IBM Workload Scheduler will include in any **GS** task associated with the job.

JS [*cputime*]

Indicates successful completion of a job and provides its elapsed run time in seconds.

RC *rc*

rc is a number that is interpreted by IBM Workload Scheduler as the return code of the extended agent job. The return code is taken into account only if a return code condition was specified in the definition of the extended agent job. Otherwise, it is ignored and the successful completion of the extended agent job is indicated by the presence of message **%JS [*cputime*]**. Likewise, if the method does not send the **%RC** message, then the successful completion of the extended agent job is indicated by the presence of message **%JS [*cputime*]**.

UT [*errormessage*]

Indicates that the requested task is not supported by the method. Displays a string of up to 255 characters that IBM Workload Scheduler will include in its error message.

Method options file

For extended, agents, and IBM Z Workload Scheduler Agent you can use a method options file to specify login information and other options.

An options file is a text file located in the methods directory of the IBM Workload Scheduler installation, containing a set of options to customize the behavior of the access method. The options must be written one per line and have the following format (with no spaces included):

```
option=value
```

All access methods use two types of options files:

Extended agents

Global options file

A common configuration file created by default for each access method installed, whose settings apply to all the extended agent workstations defined for that method. When the global options file is created, it contains only the **LJuser** option, which represents the operating system user ID used to launch the access method. You can customize the global options file by adding the options appropriate to the access method.

Local options file

A configuration file that is specific to each extended agent workstation within a particular installation of an access method. The name of this file is `XANAME_accessmethod.opts`, where:

XANAME

Is the name of the extended agent workstation. The value for *XANAME* must be written in uppercase alphanumeric characters. Double-byte character set (DBCS), Single-byte character set (SBCS), and Bidirectional text are not supported.

accessmethod

Is the name of the access method.

If you do not create a local options file, the global options file is used. Every extended agent workstation, except for z/OS, must have a local options file with its own configuration options.

For example, if the installation of the access method includes two extended agent workstations, CPU1 and CPU2, the names of the local options files are respectively `CPU1_accessmethod.opts` and `CPU2_accessmethod.opts`.

IBM Workload Scheduler reads the options file, if it exists, before running an access method. For extended agents, if the options file is modified after IBM Workload Scheduler is started, the changes take effect only when it is stopped and restarted.

IBM Z Workload Scheduler Agents and agents

Global options file

A common configuration file created by default for each access method installed, whose settings apply to all the agent workstations defined for that method. When the global options file is created, it contains only the **LJuser** option, which represents the operating system user ID used to run the access method. You can customize the global options file by adding the options appropriate to the access method.

The name of the global options file is `accessmethod.opts`, where access method is the name of the method you are creating.

Local options file

A configuration file that is specific to each access method. The name of this file is

`optionsfile_accessmethod.opts`,

In a distributed environment:

- If you are defining a job to run the access method by using the Dynamic Workload Console it is the options file you specified in the **Create new > Job definition > ERP > Access Method** XA Task tab.
- If you are defining the access method by using **composer** it is the options file you specified in the **target** attribute of the job definition.

If you do not create a local options file, the global options file is used.

In a z/OS environment:

- If you are defining a job to run the access method by using the Dynamic Workload Console it is the options file you specified in the **Create new > ERP > Access Method** XA Task tab.
- If you are defining the access method by using the **JOBREC** statement it is the name of the workstation where the access method runs.

If you do not create a local options file, the global options file is used.

If you do not specify an option in the `options_file_accessmethod.opts` file the product uses the value specified for that option in the global option file. If you do not specify them neither in the `options_file_accessmethod.opts` file nor in the global option file the product issues an error message.

The options file must have the same path name as its access method, with an `.opts` file extension. For example, the Windows® path name of an options file for a method named `netmeth` is

```
TWS_home\methods\netmeth.opts
```

IBM Workload Scheduler reads the options file, if it exists, before running an access method.

The options recognized by IBM Workload Scheduler are as follows:

LJuser=username

Specifies the login to use for the **LJ** and **MJ** tasks. The default is the login from the job definition. See [Launch job task \(LJ\) on page 172](#) and [Manage job task \(MJ\) on page 173](#).

CFuser=username

Extended agents only. Specifies the login to use for the **CF** task. The default for UNIX® is **root**, and for Windows® is the user name of the account in which the product was installed. See the topic about check file tasks (CF) extended agents in the *IBM Workload Scheduler: User's Guide and Reference*.

GSuser=username

Specifies the login to use for the **GS** tasks. The default for UNIX® is **root**, and for Windows® is the user name of the account with which IBM Workload Scheduler was installed. See Get status task (GS) extended agents only.

GStimeout=seconds

Specifies the amount of time, in seconds, IBM Workload Scheduler waits for a response before killing the access method. The default is **300** seconds.

nodename=node_name

Specifies the host name or IP address if required by the method you are defining. For the **unixssh** access method, the host name or IP address to connect to the remote engine.

PortNumber=port_number

Specifies the port number if required by the method you are defining. For the **unixssh** access method, the port to connect to the remote engine.

For IBM Z Workload Scheduler Agents and agents, you can specify the node name and port number also in the JobManager.ini file.

If you do not specify them in the *options_file_accessmethod.opts* file the product uses the value specified in the global option file. If you do not specify them neither in the *options_file_accessmethod.opts* file nor in the global option file the product uses the value specified in the *option_file* stanza of the JobManager.ini file.



Note: If the extended agent host is a Windows® computer, these users must be defined as IBM Workload Scheduler user objects.

Running methods

The following subsections describe the interchange between IBM Workload Scheduler and an access method.

Launch job task (LJ)

About this task

The **LJ** task instructs the extended agent method to launch a job on an external system or application. Before running the method, IBM Workload Scheduler establishes a run environment. The **LJuser** parameter is read from the method options file to determine the user account with which to run the method. If the parameter is not present or the options file does not exist, the user account specified in the **Logon** field of the job's definition is used. In addition, the following environment variables are set:

HOME

The login user's home directory.

LOGNAME

The login user's name.

PATH

For UNIX®, it is set to `/bin:/usr/bin`. For Windows®, it is set to `%SYSTEM%\SYSTEM32`.

TWS_PROMOTED_JOB

Set to `YES`, when the job (a mission-critical job or one of its predecessors) is promoted.

TZ

The time zone.

If the method cannot be run, the job is placed in the FAIL state.

Once a method is running, it writes messages to its **stdout** that indicate the state of the job on the external system. The messages are summarized in [Table 9: Launch job task \(LJ\) messages on page 173](#).

Table 9. Launch job task (LJ) messages

Task	Method Response	IBM Workload Scheduler Action
LJ and MJ	%CJ <i>state</i> [<i>mjstring</i>]	Sets job state to <i>state</i> . Includes <i>mjstring</i> in any subsequent MJ task.
	%JS [<i>cputime</i>]	Sets job state to SUCC.
	Exit code=non-zero	Sets job state to ABEND.
	%UT [<i>errormessage</i>] and Exit code=2	Sets job state to ABEND and displays <i>errormessage</i> .

A typical sequence consists of one or more **%CJ** messages indicating changes to the job state and then a **%JS** message before the method exits to indicate that the job ended successfully. If the job is unsuccessful, the method must exit without writing the **%JS** message. A method that does not support the **LJ** task, writes a **%UT** message to **stdout** and exits with an exit code of 2.

Manage job task (MJ)

About this task

The **MJ** task is used to synchronize with a previously launched job if IBM Workload Scheduler determines that the **LJ** task ended unexpectedly. IBM Workload Scheduler sets up the environment in the same manner as for the **LJ** task and passes it the *mjstring*. See [Launch job task \(LJ\) on page 172](#) for more information.

If the method locates the specified job, it responds with the same messages as an **LJ** task. If the method is unable to locate the job, it exits with a nonzero exit code, causing IBM Workload Scheduler to place the job in the ABEND state.

Killing a job

About this task

While an **LJ** or **MJ** task is running, the method must trap a SIGTERM signal (signal **15**). The signal is sent when an operator issues a **kill** command from IBM Workload Scheduler console manager. Upon receiving the signal, the method must attempt to stop (**kill**) the job and then exit without writing a **%JS** message.

Access methods

Access methods are used to extend the job scheduling functions of IBM Workload Scheduler to other systems and applications. They run on extended agents, dynamic agents, and IBM Z Workload Scheduler agents. They enable communication between external systems (SAP R/3) and IBM Workload Scheduler and launch jobs and return the status of jobs.

For information about the supported versions of the access methods, run the [Data Integration](#) report and select the **Supported Software** tab.

PeopleTools 8.61 and later require connecting using TLS. To continue working with an unsecure connection without enabling TLS, modify the psagent script by adding the **TM_ALLOW_NOTLS** option and setting it to **yes**.

Installing and configuring the access methods

The access methods documented in this guide are packaged with IBM Workload Scheduler and are automatically installed with the product on dynamic and fault-tolerant agents.



Important: In order to be entitled to use the access methods and plug-ins, you must have purchased at least one of the following offerings: IBM Workload Scheduler, IBM Workload Scheduler for Applications, or IBM Z Workload Scheduler Agent. See the 10.2.6 Quick Start Guide available from [IBM Fix Central](#). For information about the supported versions of the plug-ins and access methods, open the [Data Integration](#) report and select the **Supported Software** tab.

For details about installing an IBM Workload Scheduler dynamic or fault-tolerant agent, see IBM® Workload Scheduler Planning and Installation.

To use any of the access methods on supported agents, you create an options file, which configures the access method and defines the workstation and the jobs that extend the scheduling capability to external systems or applications.

Setting options for the access methods

An options file is a text file located in the `methods` directory of the IBM Workload Scheduler installation, containing a set of options to customize the behavior of the access method. The options must be written one per line and have the following format (with no spaces included):

```
option=value
```

All access methods use two types of options files: a global options file and one or more local options files. The names of the local options files are generically referred to as `XA_Unique_ID_accessmethod.opts` on extended agents and `DYNAMIC_AGENT_FILE_accessmethod.opts` on dynamic agents. The file names specified for the local options files for both types of agents must respect the following rules:

- Both `XA_Unique_ID` and `DYNAMIC_AGENT_FILE` in the file name must be uppercase alphanumeric characters. See specific requirements about `XA_Unique_ID` in [XA_Unique_ID on page 177](#).
- Double-byte character set (DBCS), single-byte character set (SBCS), and bidirectional text are not supported. For information about acceptable values for the extended agent workstation name, See [Table 10: How to complete the extended agents definition on page 179](#).

Dynamic agents and IBM Z Workload Scheduler agents

Global options file

A common configuration file created by default for each access method installed, whose settings apply to all the dynamic agent workstations defined for that method. When the global options file is created, it contains only the **LJuser** option, which represents the operating system user ID used to launch the access method. You can customize the global options file by adding the options appropriate for the access method.

The name of the global options file is `accessmethod.opts`, which, depending on your operating system, corresponds to:

For PeopleSoft

`psagent.opts`

For SAP

`r3batch.opts`

Local options file

One or more configuration files that are specific to each access method. The name of this file is `optionsfile_accessmethod.opts` and they are saved to the path `TWA_DATA_DIR/methods`.

In a distributed environment

- If you are defining a job to run the access method by using the Dynamic Workload Console, it is the options file you specify in the **New > Job definition > ERP > Access Method XA Task** tab.
- If you are defining the SAP job to run the access method by using the Dynamic Workload Console, it is the options file you specify in the **New > Job definition > ERP > SAP Job on Dynamic Workstations XA Task** tab.
- If you are defining the job to run the access method by using **composer**, it is the options file you specify in the **target** attribute of the job definition.

If you do not create a local options file, the global options file is used.

If you do not specify an option in the `optionsfile_accessmethod.opts` file, while the access method is running, the product uses the values specified for that option in the global options file.

If you do not specify options either in the `optionsfile_accessmethod.opts` or in the global option file, the product issues an error message.

If the SAP access method is installed for `AGENT1` workstation, with unique identifier, `S4HANAR3BW`, but you have two external SAP systems on which to schedule jobs, then in the `TWA_DATA_DIR/methods` directory, you create the following options files:

- `SAP1_S4HANAR3BW_r3batch.opts`
- `SAP2_S4HANAR3BW_r3batch.opts`

Each file contains the options specific to each external SAP system, for example, the connection information.

For pools and dynamic pools containing n agents, you must create an options file for the dynamic pool and copy it in the `TWA_DATA_DIR/methods` of each agent of the pool so that all members of the pool have a local options file with the same name. Then you must create another options file for the specific agent in the same directory. For example, if the SAP access method is installed for `AGENT1` and `AGENT2` which belong to the dynamic pool `DYN_POOL`, create in the `TWA_DATA_DIR/methods` directory of each agent the following options files:

AGENT1

- `FILE0PTS_AGENT1_r3batch.opts`
- `FILE0PTS_DYN_POOL_r3batch.opts`

AGENT2

- `FILE0PTS_AGENT2_r3batch.opts`
- `FILE0PTS_DYN_POOL_r3batch.opts`

Extended agents

All access methods use two types of options file:

Global options file

A common configuration file created by default for each access method installed, whose settings apply to all the extended agent workstations defined for that method. When the global options file is created, it contains only the **LJuser** option, which represents the operating system user ID used to launch the access method. You can customize the global options file by adding the options appropriate to the access method.

The name of the global options file is `accessmethod.opts`, which, depending on your operating system, corresponds to:

For PeopleSoft

`psagent.opts`

For SAP`r3batch.opts`**For custom access methods**`netmth.opts`**Local options file**

A configuration file that is specific to each extended agent workstation within a particular installation of an access method. The name of this file is `XA_Unique_ID_accessmethod.opts`, where:

XA_Unique_ID

The unique identifier of the workstation in the plan. Since the folder feature was introduced, workstations can be defined in folders, therefore, the workstation name alone, is not sufficient to identify a workstation in the plan, but instead, the name and folder combination is mapped to a unique identifier. In the `localopts` file, the value of the `this_cpu` option is the unique identifier of the workstation. You can also verify the unique identifier for a workstation by submitting the `composer list` command with the `;showid` filter. Should this result in blank, `XA_Unique_ID` corresponds to the workstation name. You can also retrieve the unique identifier submitting the `conman showcpus` command with the `;showid` filter. For example, if the installation of the `r3batch` access method includes two extended agent workstations, with unique identifiers `S4HANAR3BW` and `07756YBX76Z6AFX2`, then the names of the local options files are `S4HANAR3BW_r3batch.opts` and `07756YBX76Z6AFX2_r3batch.opts`.

accessmethod

Is the name of the access method.

If you do not create a local options file, the global options file is used. Every extended agent workstation, except for z/OS®, must have a local options file with its own configuration options.

The options files must be located in the `TWA_DATA_DIR/methods` directory. They are read when the supported agent is started. Options are specific to each access method. For details about how to configure each access method, see the following sections:

PeopleSoft

[Configuring the PeopleSoft access method on page 188.](#)

SAP

[Configuring the SAP access method on page 229.](#)

Option value inheritance

This property is currently available for r3batch only. Local options files can inherit existing values from the same options in the global options file `r3batch.opts`. For an access method, the options are listed twice; once as global options and once

as local options. If the local options file does not contain a value for the option, then the value for that option in the global options file is used. Otherwise the option value in the local options file is always used.

For example, you might want to define the same value for the **Ljuser** option and a different value for the **retrieve_joblog** option. To do this, you define the **Ljuser** option value in the `r3batch.opts` file. Then you define a different value for the **retrieve_joblog** option in each local options file. This results in the following actions when launching the SAP job:

- The value for the **Ljuser** option is extracted from the `r3batch.opts` file.
- The value for the **retrieve_joblog** option is taken from each local options file.

Defining supported agent workstations

A workstation definition is required for each entity of an access method through which IBM Workload Scheduler schedules and launches jobs.

Creating a workstation using the Dynamic Workload Console

About this task

How to create a workstation definition for supported agents using the Dynamic Workload Console.

Dynamic agents

The agents are automatically registered to the IBM Workload Scheduler network.

Extended agents

To define an extended agent workstation using the Dynamic Workload Console, perform the following steps:

1. From the Dynamic Workload Console portfolio, click **Administration > Workload Environment Design > Create Workstations**.
2. Select an engine from the list and click **Create Workstation**.
3. In the Workstations properties panel, specify the attributes for the extended agent workstation you are creating. For all the details about available fields and options, see the online help by clicking the "?" in the top-right corner. In the workstation definition, specify the access method and other properties, as shown in [Table 10: How to complete the extended agents definition on page 179](#). For further information about the workstation definition properties, see the section about workstation definition in *IBM Workload Scheduler User's Guide and Reference*.
4. To assign the workstation to an existing domain or to create a new domain, click **Assign to Domain**.
5. Click **Save**.

The following table shows how to complete some specific fields of the workstation properties panel for **extended agents**.

Table 10. How to complete the extended agents definition

Field	Description by Access Method	
	PeopleSoft	SAP
Name	The name for the extended agent workstation. For all access methods the name must start with a letter and can contain alphanumeric characters, dashes, and underscores. The maximum length is 16 characters. Workstation names must be unique and cannot be the same as workstation class and domain names. Double-byte character set (DBCS), single-byte character set (SBCS), and bidirectional text are not supported. If a workstation name contains these characters and, as a result, the options file name contains the same name, the workstation cannot be validated by the SAP system. For all the access methods, the name of the options file associated with the extended agent workstation must contain the unique identifier of the extended agent workstation, <i>XA_Unique_ID_<access-method>.opts</i>. That is, if the unique identifier for the extended agent workstation named <i>/SAPBUS/S4HANAR3BW</i> is <i>07756YBX76Z6AFX2</i>, then the options file name must be <i>07756YBX76Z6AFX2_r3batch.opts</i>. For information about retrieving the unique identifier for an extended agent workstation, see UNIQUE_ID on page 177.	
TCP Port	Any number other than 0.	
Access Method	psagent	r3batch



Note: In UNIX™ the name is case sensitive and must be lowercase.

Creating a workstation using the command line

You can define supported agents workstations also using the **composer** command line of IBM Workload Scheduler.

Dynamic agents

The following example shows a definition for a dynamic agent workstation named *LINUX248* that uses the secure protocol *https* to connect to the Broker server.

```
CPUNAME LINUX248
DESCRIPTION "This workstation was automatically created."
OS UNIX
NODE linux248.romelab.it.abc.com SECUREADDR 31114
```

```

TIMEZONE Europe/Rome
FOR MAESTRO HOST NC118003_DWB
  AGENTID "FD640FCA740311E18C4EE96D727FA991"
  TYPE AGENT
  PROTOCOL HTTPS
END

```

Extended agents

The following example shows a definition for a z/OS extended agent workstation named *MVSCPU* that uses the `mvsjes` access method.

```

cpuname MVSCPU description "zOS extended agent"
  os other
  node mvsesa36.rome.abc.com
  tcpaddr 5000
  domain masterdm
  for maestro
    type x-agent
    host ROCIOUS
    access mvsjes
end

```

For details about defining workstations with **composer**, see the *IBM Workload Scheduler User's Guide and Reference*.

Defining workstations for end-to-end scheduling

About this task

How to create a workstation definition for end-to-end environment.

Scheduling in an end-to-end environment means that in IBM Z Workload Scheduler you are scheduling and monitoring jobs that are physically running on IBM Workload Scheduler workstations.

Extended agents

Extended agent workstations must be defined as fault-tolerant workstations in IBM Z Workload Scheduler.

A fault-tolerant workstation is the IBM Z Workload Scheduler definition of an existing IBM Workload Scheduler agent in the distributed network. The IBM Workload Scheduler agent is where the job associated with the fault-tolerant workstation actually runs in the distributed network.

To define the extended agent workstation in IBM Z Workload Scheduler, you must:

1. Define the workstation in the CPUREC initialization statement. For an example, see [Creating the CPUREC statement for extended agents on page 181](#).
2. Add the same workstation definition to the database using ISPF or the Dynamic Workload Console. For a description of how to define the workstation using the Dynamic Workload Console, see *Dynamic Workload Console User's Guide*. For an example, see [Defining the workstation with ISPF on page 182](#).

IBM Z Workload Scheduler agents

To define the agent workstation with z-centric capability in IBM Z Workload Scheduler, add the workstation definition to the database using ISPF or the Dynamic Workload Console. For further information, see *Scheduling End-to-end with z-centric Capabilities*.

Creating the CPUREC statement for extended agents

This section is valid only for Extended agents. Create the CPUREC statement for the workstation in the TOPOLOGY initialization statement. The TOPOLOGY initialization statement is used to define parameters related to the topology of the connected IBM Workload Scheduler network. Such a network topology statement is made up of one or more (one for each domain) DOMREC statements that describe the topology of the distributed network, and by several CPUREC statements, one for each fault-tolerant workstation.

The following example shows a CPUREC statement for an SAP extended agent workstation named R3XA. The extended agent is hosted by an IBM Workload Scheduler agent named TWSA, which is also the domain manager of DOMAIN1.

```
*****TPLGINFO MEMBER *****

/*****/
/* DOMREC: Domain definition */
/*****/
DOMREC    DOMAIN(DOMAIN1)
          DOMMNGR(TWSA)
          DOMPARENT(MASTERDM)

/*****/
/* CPUREC: Extended agent workstation definition */
/*****/
CPUREC    CPUNAME(R3XA)
          CPUOS(OTHER)
          CPUNODE(NODE1)
          CPUDOMAIN(DOMAIN1)
          CPUHOST(TWSA)
          CPUTYPE(XAGENT)
          CPUACCESS(r3batch)
          CPUUSER(TWSuser)
          CPUTZ('Europe/Rome')

/*****/
/* CPUREC: Domain manager workstation definition */
/*****/
CPUREC    CPUNAME(TWSA)
          CPUNODE(NODE1)
          CPUAUTOLINK(ON)
          CPUDOMAIN(DOMAIN1)
          CPUTYPE(FTA)
          CPUUSER(TWSuser)
          CPUTZ('Europe/Rome')
```

The following keywords define R3XA as an extended agent:

CPUACCESS

The extended agent access method. For SAP, it is r3batch.

CPUHOST

The name of the IBM Workload Scheduler workstation hosting the extended agent. It cannot be another standard agent or extended agent.

CPUTYPE

The workstation type. For an extended agent, it must be `XAGENT`.



Note: The `CPUREC` statement does not exist for an IBM Workload Scheduler for z/OS agent workstation.

For further information about `CPUREC` for extended agents, see *Customization and Tuning*.

Defining the workstation with ISPF

About this task

This section shows the ISPF definition for extended agents and agents with z-centric capability.

Extended agents

In ISPF, define the workstation as `computer automatic` and then set the *FT Work station* field to **Y**. The `CPUREC` statement with the three keywords described in [Creating the CPUREC statement for extended agents on page 181](#) provides the extended agent specification.



Note: Make sure you write the `CPUREC` statement before making the ISPF or Dynamic Workload Console definition, because they have no effect without the `CPUREC` statement.

Figure 13. Defining an Extended Agent workstation

```

----- CREATING GENERAL INFORMATION ABOUT A WORK STATION -----
Command ==>

Enter the command R for resources  A for availability or M for access method
above, or enter data below:

WORK STATION NAME  ==> R3XA
DESCRIPTION        ==> Extended agent for R/3 access method-----
WORK STATION TYPE  ==> C      G General, C Computer, P Printer
REPORTING ATTR     ==> A      A Automatic, S Manual start and completion
                                C Completion only, N Non reporting
FT Work station    ==> Y      FT Work station, Y or N
PRINTOUT ROUTING   ==> SYSPRINT The ddname of daily plan printout data set
SERVER USAGE       ==> N      Parallel server usage C , P , B or N

Options:
SPLITTABLE         ==> N      Interruption of operation allowed, Y or N
JOB SETUP          ==> N      Editing of JCL allowed, Y or N
STARTED TASK, STC  ==> N      Started task support, Y or N
WTO                ==> N      Automatic WTO, Y or N
DESTINATION        ==> ----- Name of destination
Defaults:
TRANSPORT TIME     ==> 00.00   Time from previous work station HH.MM
DURATION           ==> ----- Duration for a normal operation HH.MM.SS

```

IBM Z Workload Scheduler agents

For detailed information and examples about the ISPF definition of IBM Z Workload Scheduler agents with z-centric capabilities, see *Scheduling End-to-end with z-centric capabilities*.

Defining jobs for supported agents

To run and monitor a PeopleSoft, SAP, or z/OS job with IBM Workload Scheduler, the supported agents, or access method require an IBM Workload Scheduler job definition, where you specify the external job you want to schedule, the workstation (also defined in IBM Workload Scheduler) on which it is to run, and any recovery actions. To define the job, use either of the following methods:

- Dynamic Workload Console.
- IBM Workload Scheduler **composer** command line.

If you are scheduling in an end-to-end environment, to define the job, use either of the following methods:

- Dynamic Workload Console.
- IBM Z Workload Scheduler ISPF dialogs. You must also create a member in the `SCRIPTLIB` with a `JOBREC` statement for the job.

Jobs defined for supported agents are added to job streams and scheduled in the same way as any other job in IBM Workload Scheduler and IBM Z Workload Scheduler.

Defining jobs with the Dynamic Workload Console

About this task

How to create a job definition for supported agents using the Dynamic Workload Console.

Steps for defining a job for supported agents.

To define jobs, follow these steps:

1. From the Dynamic Workload Console, click the **Design** menu and select Workload Designer.
2. Specify an engine name, either distributed or z/OS. The Workload Designer window opens. Job types and characteristics vary depending on whether you select a distributed or a z/OS engine.
3. Click **Create new** and select **Job definition**.
4. Select the category and type of job you want to create.
 - For SAP jobs, **ERP > SAP Job on XA Workstations** or **SAP Job on Dynamic Workstations**. See [Creating SAP Standard R/3 jobs from the Dynamic Workload Console on page 254](#).
 - For z/OS and PeopleSoft, **ERP > Access Method**.
5. In the properties panel, specify the attributes for the job definition you are creating. For all the details about available fields and options, see the online help specific for the item.
6. Click **Save** to save the job definition in the database.



Note: The access method for SAP provides supplementary features if you use the alternative steps described in [Creating SAP Standard R/3 jobs from the Dynamic Workload Console on page 254](#). You can create native SAP Standard jobs on a remote SAP system directly from the Dynamic Workload Console.

Defining jobs using the command line

You can also define jobs using the **composer** command line of IBM Workload Scheduler.

Dynamic agents

The following example describes an IBM Workload Scheduler job named `DYN_JOB_R3_0001` defined in the folder name `SAPJOBS`, that runs on a dynamic agent workstation named `NC112015_1`. The IBM Workload Scheduler launches a job in an SAP environment named `JOB_APPS_93`.

```
NC112015_1#/SAPJOBS/DYN_JOB_R3_0001
TASK
<?xml version="1.0" encoding="UTF-8"?>
<jSDL:jobDefinition
  xmlns:jSDL="http://www.abc.com/xmlns/prod/scheduling/1.0/jSDL"
  xmlns:jSDLxa="http://www.abc.com/xmlns/prod/scheduling/1.0/jSDLxa"
    name="r3">
  <jSDL:application name="r3" plugin="xajob">
    <jSDLxa:xajob accessMethod="r3batch" target="NW73LIN_r3batch">
      <jSDLxa:taskString>/ -job JOB_APPS_93 -i 14514200 -c c
        -flag ENABLE_APPL_RC </jSDLxa:taskString>
    </jSDLxa:xajob>
  </jSDL:application>
</jSDL:jobDefinition>
RECOVERY STOP
```

Extended agents

The following example describes an IBM Workload Scheduler job named `psjob2` that runs on a PeopleSoft extended agent workstation with unique identifier named `XAPS002`. IBM Workload Scheduler logs on to UNIX operating system as `psjobs` and launches a job under PeopleSoft. The PeopleSoft process is named `XRFWIN`. If recovery is needed, IBM Workload Scheduler runs job `recov2` and then continues processing.

```
XAPS002#/myspsjobs/psjob2
streamlogon psjobs
scriptname
  -process XRFWIN -type 'SQR Report' -runcontrol 1 -runlocationdescr PSNT
description "peoplesoft job #2"
recovery continue after recov2
```

The arguments of `scriptname` differ by application. For details, see:

- [Task string parameters for PeopleSoft jobs on page 196](#).
- [Task string to define SAP jobs on page 262](#).

For more information about using the **composer** command line to define jobs, see *User's Guide and Reference*.

Defining jobs for end-to-end scheduling

Extended agents

Extended agent jobs scheduled to run in an end-to-end environment cannot be defined using the Dynamic Workload Console or the IBM Workload Scheduler command line, but must be added to the `SCRIPTLIB` of IBM Z Workload Scheduler.

In the OPERATIONS ISPF panel of IBM Z Workload Scheduler, extended agent jobs are defined like any other job, but with the specific attribute for a job defined on an extended agent workstation. The following example shows the definition of a job named `SAPJOB`. This is the IBM Z Workload Scheduler job that drives the running of on SAP R/3 job (named `BAPRINT46B` as shown in the next example). It shows as an extended agent job because the associated workstation is an extended agent workstation named `R3XA`.

Figure 14. Defining an Extended Agent job for end-to-end scheduling

```

-----Row 1 to 1 of 1
Command ==> Scroll ==> PAGE

Enter/Change data in the rows, and/or enter any of the following
row commands:
I(nn) - Insert, R(nn),RR(nn) - Repeat, D(nn),DD - Delete
S - Select operation details, J - Edit JCL
Enter the TEXT command above to include operation text in this list, or,
enter the GRAPH command to view the list graphically.

Application          : APLL1          FTW appl

Row Oper      Duration  Job name  Internal predecessors      Morepreds
cmd  ws   no.  HH.MM.SS  SAPJOB_____ -IntExt-
'''  R3XA  001  00.00.01  SAPJOB_____      0    0
***** Bottom of data *****

```

For each job, create a member in the `SCRIPTLIB` of IBM Z Workload Scheduler with details about the job in a `JOBREC` statement. A `SAPJOB` member was created for the job of the previous example. It contains a `JOBREC` statement like this:

```

JOBREC
  JOBCMD('/-job BAPRINT46B -user MAESTRO -i 14160001 -c C')
  JOBUSR(twsila)

```

The string in `JOBCMD` is read and interpreted by the access method before running the job. The job of this example, `BAPRINT46B`, was previously defined on SAP R/3 and assigned with an ID of 14160001, that was manually written in `JOBCMD`.

The following example is for a PeopleSoft job. The entire string that follows the `JOBCMD` keyword must be enclosed within quotation marks ("), because for PeopleSoft jobs single quotes are already used in the string.

```
JOBREC
  JOBCMD("/ -process XRFWIN -type 'SQR Report' -runcontrol IWS")
  JOBUSR(PsBuild)
```

IBM Z Workload Scheduler agents

For information about the jobs definition for agent with z-centric capabilities, see *Scheduling End-to-end with z-centric capabilities*.

The arguments of `JOBCMD` differ by application. For details, see:

- [Task string parameters for PeopleSoft jobs on page 196.](#)
- [Task string to define SAP jobs on page 262](#) or [Defining SAP jobs dynamically on page 282.](#)

Submitting jobs

About this task

To submit jobs on the supported agent workstation, perform the following steps:

1. Verify that the application system to which the job belongs and the related database is up and running.
2. Launch the job. For details, see:

Dynamic agents

- *IBM Workload Scheduler User's Guide and Reference* for **conman** command line.
- *Dynamic Workload Console User's Guide* for Dynamic Workload Console.

Extended agents

- *IBM Workload Scheduler User's Guide and Reference* for **conman** command line.
- *Dynamic Workload Console User's Guide* for Dynamic Workload Console.

IBM Z Workload Scheduler agents

- *IBM Z Workload Scheduler: Scheduling End-to-end with z-centric Capabilities* for **ISPF** panel.
- *Dynamic Workload Console User's Guide* for Dynamic Workload Console.

Access method for PeopleSoft

What you need and what you can do with Access method for PeopleSoft.

Using Access method for PeopleSoft you can run and monitor PeopleSoft jobs from the IBM Workload Scheduler environment. These jobs can be run as part of a schedule or submitted for ad-hoc job processing. PeopleSoft extended agent or dynamic agent jobs can have all of the same dependencies and recovery options as other IBM Workload Scheduler jobs. PeopleSoft jobs must be defined in IBM Workload Scheduler to be run and managed in the IBM Workload Scheduler environment.

For information about the supported versions of the plug-ins and access methods, run the [Data Integration](#) report and select the **Supported Software** tab.

Features

Look at the tasks you can perform by using Access method for PeopleSoft.

Using Access method for PeopleSoft, you can perform the following tasks:

- Use IBM Workload Scheduler standard job dependencies on PeopleSoft jobs.
- Schedule PeopleSoft jobs to run on specified days, times, and in a prescribed order.

- Define inter-dependencies between PeopleSoft jobs and IBM Workload Scheduler jobs that run on different applications such as SAP and Oracle E-Business Suite.
- Define inter-dependencies between PeopleSoft jobs and jobs that run on different operating systems.

Roles and responsibilities

Here you can see the roles and responsibilities of all the actors involved in the process model, and the tasks they perform.

In a typical enterprise, different users contribute to the implementation and operation of the product. [Table 11: Roles and responsibilities in Access method for PeopleSoft on page 187](#) describes the roles and responsibilities of all those involved in the process model, showing the tasks they perform.

Table 11. Roles and responsibilities in Access method for PeopleSoft

User role	User task
IBM Workload Scheduler configurator	Defining the configuration options on page 188
IBM Workload Scheduler developer	• Defining PeopleSoft jobs in IBM Workload Scheduler on page 196 • Configuring the job status mapping policy on page 198
PeopleSoft administrator	• Creating a batch processing ID in PeopleSoft on page 192 • Configuring the ITWS_PSXA PeopleSoft project on page 192 • Uploading the PeopleSoft project on page 193

Scheduling process for the PeopleSoft supported agents

IBM Workload Scheduler can launch and monitor jobs in the PeopleSoft process scheduler using a PeopleSoft extended agent or dynamic agent workstation. The PeopleSoft supported agent (extended agent or dynamic agent) is defined in a standard IBM Workload Scheduler workstation definition. This definition is a logical workstation name and specifies the access method as `psagent`. The access method is used to communicate job requests to the PeopleSoft process scheduler.

To launch a PeopleSoft job, IBM Workload Scheduler runs the `psagent` method, passing it information about the job. An options file provides the method with the path, the executable, and other information about the PeopleSoft process scheduler and application server used to launch the job. The supported agent can then access the PeopleSoft process request table and make an entry in the table to launch the job. Job progress and status information is written to the job's standard list file.

For extended agents, there is no need to install Database connectivity on fault-tolerant agents hosting PeopleSoft extended agents because the method currently uses the PeopleSoft 3-tier architecture. You must configure at least one PeopleSoft Application Server for the supported agent to work. The application server must be active to successfully submit jobs to the PeopleSoft process scheduler.

PeopleSoft job tracking in IBM Workload Scheduler

A PeopleSoft job is a collection of processes that run together as a single unit. IBM Workload Scheduler jobs can be defined in one of the following ways:

- As PeopleSoft jobs, that is, as a collection of PeopleSoft processes. In this case, the status of the PeopleSoft job is tracked, not the status of the individual processes within the job.
- As PeopleSoft processes. In this case, the status of the individual process is tracked and IBM Workload Scheduler schedules can be defined to create complex inter-dependencies and recovery options between PeopleSoft processes.

Security

Security for the PeopleSoft jobs is handled by standard IBM Workload Scheduler security.

Configuring the PeopleSoft access method

This section provides detailed reference information about the PeopleSoft options and how to define them in the options file.

Defining the configuration options

The IBM Workload Scheduler installation process creates a default global options file for the `psagent` access method, named `psagent.opts`. You can also create the following local files in the path:

On UNIX operating systems

`psjoa.jar`

`TWA_DATA_DIR/methods`

On Windows operating systems

`TWA_home\methods`

Extended agent

`XA_Unique_ID_psagent.opts` where `XA_Unique_ID` is the unique identifier for the extended agent workstation. For more details about how to identify the unique ID, see [UNIQUE_ID on page 177](#).

Dynamic agent

`DYNAMIC_AGENT_FILE_psagent.opts` where `DYNAMIC_AGENT_FILE` is any text string. This string does not necessarily correspond to the name of the dynamic agent workstation since the dynamic agent can have more than one `.opts` file associated. For more information, see [Setting options for the access methods on page 174](#).

To edit both options file, you can use any text editor. On dynamic workstations, you can edit the options files from the job definition panels in the Dynamic Workload Console. For examples of options files for this access method, see [PeopleSoft options file example on page 190](#).

[Table 12: Psagent access method options on page 189](#) describes the options for the `psagent` access method. Option names are case insensitive. Before you use a manually-created options file, check that all the option names are written correctly, otherwise they will be ignored.

Table 12. Psagent access method options

Option	Description
CHECKINTERVAL	(Optional) Specifies the frequency (in seconds) with which the psagent monitors a submitted process up to completion. The default is 120.
LJUSER	(Optional) Specifies the ID of the IBM Workload Scheduler user that runs the psagent to launch jobs (LJ tasks). This user must be a valid IBM Workload Scheduler user on the IBM Workload Scheduler hosting workstation.
PS_DISTSTATUS	(Optional) Determines whether the distribution status of the PeopleSoft request is taken into account when determining the status of the IBM Workload Scheduler job. Values are 0 (not taken into account) or 1 (taken into account - the default value).
PSFT_DOMAIN_PWD	(Optional) Specifies the encrypted password (case-sensitive) of the PeopleSoft domain used for the connection to the PeopleSoft application server. For details about how to encrypt the password, see Encrypting PeopleSoft operator passwords on page 191.
PSFT_OPERATOR_ID	(Mandatory) Specifies the PeopleSoft operator ID used for the connection to the PeopleSoft application server.
PSFT_OPERATOR_PWD	(Mandatory) Specifies the encrypted password (case-sensitive) of the PeopleSoft operator ID used for the connection to the PeopleSoft application server. For details about how to encrypt the password, see Encrypting PeopleSoft operator passwords on page 191.
PSJOAPATH	(Optional) Specifies the full path name of the psjoa.jar file, containing both the path and the psjoa.jar filename. If this option is not set, the following default path name is used: TWA_DATA_DIR/methods/psjoa.jar TWA_home\methods\psjoa.jar Ensure that you identify the version of the psjoa.jar file that corresponds to the version of PeopleSoft that you are using and you have access to the file.
RUNLOCATION	(Optional) Specifies the default PeopleTools process server that processes the requests.
SERVER_NAME_LIST	(Mandatory) Specifies the list of application servers that the psagent tries to connect to. It is a comma-separated list of addresses in the format:

Table 12. Psagent access method options (continued)

Option	Description
	<pre>server:port [,server:port] ...</pre> where: server Specifies the host name or TCP/IP address of the server port Specifies the port number the server is listening on.
TWS_MAX_WAIT_TIME	(Optional) Specifies the maximum time that the supported agent waits (timeout) after a failed operation on the PeopleSoft application server before retrying the operation. The default is 10 seconds.
TWS_MIN_WAIT_TIME	(Optional) Specifies the minimum time that the supported agent waits (timeout) after a failed operation on the PeopleSoft application server before retrying the operation. The default is 5 seconds.
TWS_RETRY	(Optional) The maximum number of times that the supported agent attempts to re-run a failed operation on the PeopleSoft application server. The default is 5.
TWSXA_INLINE_CI	(Optional) Specifies the name of the component interface that the <code>psagent</code> invokes to submit jobs to PeopleSoft. The default is <code>ITWS_PROCESSREQUEST</code>. If you use this default, you must perform the customization steps described in Configuring the ITWS_PPSXA PeopleSoft project on page 192. If you do not plan to schedule jobs containing in-line variables, and you do not want to perform the additional customization steps, you must replace the default value with <code>PROCESSREQUEST</code>. This is the component interface invoked by previous versions of the access method; it does not allow the use of in-line variables.
TWSXA_SCHED_METH	(Optional) Specifies the name of the PeopleSoft method invoked by the component interface specified in <code>TWSXA_INLINE_CI</code>. Both <code>ITWS_PROCESSREQUEST</code> and <code>PROCESSREQUEST</code> use the default method <code>Schedule</code>. If you are using either of these component interfaces, leave the default. If you are using a different component interface, specify the name of the method called by your component interface, respecting the case of the PeopleSoft object name.

PeopleSoft options file example

Below is a sample options file. It can help you determine your specific site requirements although your options file might be different.

Remember to save the file in the following directory:

```
TWA_DATA_DIR/methods
```

```
TWA_home\methods
```

Example

```
LJuser=TwsUsr
CheckInterval=120
PSFT_OPERATOR_ID=PSHC
PSFT_OPERATOR_PWD=*****
SERVER_NAME_LIST=9.87.120.36:9000
```

If you create the options file manually, you must encrypt the PeopleSoft operator password, as described in [Encrypting PeopleSoft operator passwords on page 191](#).

Encrypting PeopleSoft operator passwords

When you add or change the PeopleSoft operator password using the Dynamic Workload Console, the password is automatically encrypted and securely stored in the file. For added security, it is displayed on the screen as a series of asterisks.

When you add or change the PeopleSoft user password using a text editor, run the **pwdcrypt** command to encrypt the password before writing it in the file, as follows:

```
pwdcrypt password
```

The program returns the password in encrypted format that you can then copy and paste into the options file.

Connecting to more than one PeopleSoft application server

It might be necessary for the **psagent** method to connect to more than one PeopleSoft application server. For example, a single installation of PeopleSoft might have a TEST, DEMO, and PRODUCTION environment, each with a separate application server. This requires that the **psagent** method uses a separate connect string for each application server.

To support this, you can set up multiple PeopleSoft extended agent workstations that connect to the same method but use different options files. When a workstation starts the method, it first looks for the options file with extended agent workstation unique identifier prepended to **psagent.opts**. For example, a PeopleSoft extended agent with unique identifier **ps847system** would have the following options file:

```
PS847SYSTEM_psagent.opts
```

The **psagent** method searches first for an options file with the extended agent workstation unique identifier, and then for the default **psagent.opts** file. This allows the user to set up an extended agent for each PeopleSoft application server.

To connect to only one application server, use the default name for the options file, **psagent.opts**.



Note: In case you specify some connection properties in your local option files, make sure that the same properties are commented out in your global option file, with the exception of the global property **LJuser**. This action is needed to avoid that warning messages related to duplicate properties are displayed in the job log.

Creating a batch processing ID in PeopleSoft

Create an operator ID in PeopleSoft dedicated to batch scheduling. This operator ID must be granted authority to use the Component Interface in the PeopleTools environment. All the jobs submitted by IBM Workload Scheduler should use this operator ID.

Configuring the ITWS_PSXA PeopleSoft project

About this task

The configuration steps described in this section are necessary to enable IBM Workload Scheduler to schedule PeopleSoft jobs that have in-line variables in their definitions.

The `ITWS_PROCESSREQUEST` component interface works around some limitations of the PeopleSoft APIs when invoked from a batch environment. Because of these limitations, IBM Workload Scheduler cannot schedule jobs defined with in-line bind variables. With current PeopleSoft APIs, data that is stored in the PeopleSoft database and referred to by a `runcontrol ID` parameter that is used to retrieve a `runcontrol data record`, needs to be loaded into the `Component Buffer` before scheduling the API invocation. This cannot be done from a batch environment. Therefore, when invoking the PeopleSoft scheduling APIs from a batch interface, the data related to the `runcontrol ID` is not available for the submission of a job, even though it is available in the database. When unresolved data is present in the submitted job, the PeopleSoft system refuses submission and ends with an error.

The `ITWS_PROCESSREQUEST` component interface enables IBM Workload Scheduler to schedule PeopleSoft jobs that have in-line variables in their definitions. By invoking this component interface, the access method provides the ability to use data stored in the PeopleSoft database to resolve in-line variable values by taking data from the database and substituting it with variable definitions. It then allows job submission regardless of the use of in-line variable definitions in the jobs. The variable substitution mechanism does not support `work` records, so if the PeopleSoft process uses work records in its parameter list, you find a message similar to the following in the IBM Workload Scheduler joblog:

```
Error Position: 21
Return: 942 - ORA-00942: table or view does not exist
Statement:
select nvsdlist from PS_NVS_WRK WHERE BUSINESS_UNIT = :1 AND REPORT_ID = :2
Original Statement:
SELECT NVSDLIST FROM PS_NVS_WRK WHERE BUSINESS_UNIT = :1 AND REPORT_ID = :2.
```

To identify work records, use the following PeopleSoft naming conventions:

- A derived work record name ends with '`_WRK`'.
- A work record definition name for Structured Query Report reports starts with `R_`

When you use IBM Workload Scheduler to submit a process that has in-line bind variables, the name of the process type in the PeopleSoft GUI becomes `ITWS_process type`. For example, `SQR Process` becomes `ITWS_SQR Process`.

To schedule a job that contains in-line variables in its definition you must perform the following tasks:

- Leave the value of the `TWSXA_INLINE_CI` option set to `ITWS_PROCESSREQUEST`, that is the default value. See [Defining the configuration options on page 188](#) for a detailed explanation.
- Upload the PeopleSoft project as described in [Uploading the PeopleSoft project on page 193](#).

Uploading the PeopleSoft project

About this task

This section describes how to upload a new PeopleSoft project related to PeopleTools 8.44, or later, into the PeopleSoft database. The name of the PeopleSoft project is `ITWS`.

After installing the product, complete the following steps:

1. Mount the PT844 PeopleSoft project directory or copy it to the workstation from where you launch the Application Designer. IBM Workload Scheduler installs the PeopleSoft project directories, as shown in the following structure:

```
TWS_DIR/methods/
/---PeopleSoft
  /---PT844
    /---ITWS_PSPA
      ITWS_PSPA.ini
      ITWS_PSPA.XML
```

```
TWS_DIR\methods\
\---PeopleSoft
  \---PT844
    \---ITWS_PSPA
      ITWS_PSPA.ini
      ITWS_PSPA.XML
```

2. Start the Application Designer and from the sign-on window select to start the Application Designer in tier-two mode by entering the following information:
 - **Connection Type:** *database used*; for example, Oracle
 - **Database Name:** *database instance name*
 - **User ID:** *PeopleSoft operator name*; for example, PS
 - Password of user ID
3. Using the Application Designer, select **Tools -> Copy Project-> From file...**
4. Using the browser, edit the full path to specify the folder where the project that you want to load is located.

The project is contained in the `TWS_DIR/methods/PeopleSoft/PT844` subdirectories on UNIX® and in `TWA_home\methods\PeopleSoft\PT844` on Windows™.

After you specify the project folder, a list of projects appears in the **Project Name** field of the Copy Project From File window.

5. Choose **ITWS_PSPA** and click **Open**. If you already configured **ITWS_PSPA** (perhaps after installing a fix pack), a confirmation window enquires if you want to replace the existing one. Click **Yes**.

The Copy window is displayed showing a list of definition types.

6. Click **Options** to select the new settings.
 - a. Click **Report Filter**
 - b. Click **Select All**
 - c. Click **OK**
 - d. Click **Select All**
 - e. Click **Copy**. A progress bar is displayed.

After loading the project, the PeopleSoft Database contains the following objects:

- ITWS process type definitions
- ITWS permissions list
- ITWS component interfaces

7. Create the **ITWS_ROLE** security role. You can use either the PeopleSoft Web GUI or the Application Designer. Follow the steps below:

From the menu of the PeopleSoft Web GUI:

- a. Select **NavBar** in the upper right corner -> **Menu** -> **PeopleTools** -> **Security** -> **Permission and Roles** -> **Roles**.
- b. Select the **Add a new value** tab
- c. Type or select **ITWS_ROLE** in the **Role Name** field
- d. Select the **Permissions list** tab -> **ITWS** -> **Save**

From the Application Designer GUI:

- a. Using Maintain Security, edit the **ITWS_ROLE** window
- b. Select the **Permissions list** tab -> **ITWS** -> **Save**

8. Grant **ITWS_ROLE** authority to all users who want to schedule jobs from IBM Workload Scheduler. You can use either the PeopleSoft Web GUI or the Application Designer. Follow the steps below:

From the PeopleSoft Web GUI:

- a. Select **NavBar** in the upper right corner -> **Menu** -> **PeopleTools** -> **Security** -> **User Profiles**.
- b. Type the user name of the user who wants to schedule jobs from IBM Workload Scheduler.
- c. In **User Roles** window click on the + sign in the row where you want to add the role.
- d. Select the **Roles** tab.
- e. Add **ITWS_ROLE** and save.

From the Application Designer GUI:

- a. Using Maintain Security, edit the user name
- b. Select the **Roles** tab
- c. Add **ITWS_ROLE** and save

9. Add the ITWS process type definitions to the required PeopleTools process scheduler. You can use either the PeopleSoft Web GUI or the Application Designer. Follow the steps below:

From the PeopleSoft Web GUI:

- Select **NavBar** in the upper right corner -> **Menu** -> **PeopleTools** -> **Process Scheduler** -> **Process Scheduler Servers**.
- Click on the **Search** button and select the PeopleTools server you plan to use.
- In the **Server Definition** tab and in **Process Types run on this Server** window click on the + sign in the row of your choice.
- Click on the search icon. A new windows is displayed.
- Select all the **ITWS_*** entries (one for each row created).
- Click Save.

From the Application Designer GUI:

- Select **Process Scheduler Manager**
- Select your PeopleTools server
- Add the ITWS_* Type definitions and save



Note: From the SQL interactive command line, the same task can be performed by the following sample statement, customized for your database environment:

```
INSERT INTO PS_SERVERCLASS SELECT o.SERVERNAME,
o.OPSYS, 'ITWS_' || o.PRCSTYPE, o.PRCSPRIORITY,
o.MAXCONCURRENT FROM PS_SERVERCLASS
o WHERE ( SELECT count(*) FROM PS_SERVERCLASS i WHERE
i.SERVERNAME=o.SERVERNAME AND i.OPSYS=o.OPSYS AND
i.PRCSTYPE='ITWS_' || o.PRCSTYPE ) = 0
AND ( select count(*) from PS_PRCSTPEDEFN
a where a.PRCSTYPE='ITWS_' || o.PRCSTYPE AND a.OPSYS=o.OPSYS ) > 0
```

10. Restart the process servers.

You do not need to change the existing IBM Workload Scheduler job definitions, except for the scheduling nVision process, where the `runcontrol` ID must be specified using the `BUSINESS_UNIT.REPORT_ID` convention.

The following is an example of a job definition for the scheduling nVision process:

```
-process 'NVSRUN' -type nVision-Report -runcontrol AUS01.VARIABLE
```

where `NVSRUN` is the process name and `AUS01.VARIABLE` is the `BUSINESS_UNIT.REPORT_ID`.

Defining PeopleSoft jobs

This section provides job definition information for jobs using the extended agent for PeopleSoft.

Defining PeopleSoft jobs in IBM Workload Scheduler

An IBM Workload Scheduler job definition is required for every PeopleSoft job you want to manage. An IBM Workload Scheduler job is associated to an already defined PeopleSoft job and its definition includes:

- The name of the IBM Workload Scheduler job that runs the PeopleSoft job
- The unique identifier of the extended agent or dynamic workstation or workstation class where the IBM Workload Scheduler job runs. See [UNIQUE_ID on page 177](#) for more information about the unique identifier.
- The name of the user launching the job
- Recovery options
- The Script file specifications

For more information, refer to [Defining jobs for supported agents on page 183](#).

Task string parameters for PeopleSoft jobs

This section describes the task string parameters that control the operation of PeopleSoft jobs. You must specify them in the following places when you define their associated IBM Workload Scheduler jobs:

- In the **Task string** field of the Task page of the Properties - Job Definition panel, if you use the Dynamic Workload Console
- As arguments of the `scriptname` keyword in the job definition statement, if you use the IBM Workload Scheduler command line.
- As arguments of the `JOBCMD` keyword in the `JOBREC` statement in the `SCRIPTLIB` of IBM Z Workload Scheduler, if you are scheduling in an end-to-end environment. In this case the entire string following the `JOBCMD` keyword must be enclosed within quotation marks (").

The following is an example of a `JOBREC` statement:

```
JOBREC
  JOBCMD("/-process process_name -type 'process_type' -runcontrol runcontrol_ID")
  JOBUSR(TWS_user_name)
```

where:

process_name

The process name for the PeopleSoft job.

process_type

The process type for the PeopleSoft job. This entry must be enclosed within single quotes.

runcontrol_ID

The runcontrol ID for the PeopleSoft job.

TWS_user_name

The IBM Z Workload Scheduler user who runs the `psagent` access method from the end-to-end scheduling environment.

Table 13: Task string parameters for PeopleSoft jobs on page 197 describes the parameters to define PeopleSoft jobs.

Table 13. Task string parameters for PeopleSoft jobs

Parameter	Description
-process	The process name for the PeopleSoft job.
-type	The process type for the PeopleSoft job. This entry must be enclosed within single quotes.
-runcontrol	The runcontrol ID for the PeopleSoft job.
-outputdest	The destination of the PeopleSoft job output.
-outputtype	The output type of the PeopleSoft job. Possible values are: • Any • Email • File • None • Printer • Web • Window If you do not specify any value, IBM Workload Scheduler uses the value associated to the PeopleSoft job you are submitting.  Note: Depending on the PeopleSoft configuration, some combinations of the value of this option with the value of the outputformat option are not supported. In this case the PeopleSoft default value is used.
-outputformat	The output format of the PeopleSoft job. Valid values are: None PDF CSV PS DOC RPT Default RTF HTM TXT

Table 13. Task string parameters for PeopleSoft jobs (continued)

Parameter	Description
	LP
	WKS
	OTHER
	XLS
	 Note: Depending on the PeopleSoft configuration, some combinations of the value of this option with the value of the outputtype option are not supported. In this case the PeopleSoft default value is used.
-runlocationdescr	The PeopleSoft process scheduler responsible for processing the PeopleSoft job.
-foldername	The name of the report folder used for this job. The folder must have been already created using PeopleSoft Report Manager.
tracelvl	Specify the trace setting for the job. Possible values are: 1 Only error messages are written in the trace file. This is the default. 2 Informational messages and warnings are also written in the trace file. 3 A most verbose debug output is written in the trace file. Refer to Configuring the tracing utility on page 207 for detailed information.
	 Note: No syntax checking is performed on the output control values (<code>outputdest</code> , <code>outputtype</code> , <code>outputformat</code> , and <code>foldername</code>). If the values are not recognized, defaults are used.

The following is an example of a task string specification for a PeopleSoft 8.44 job:

```
-process XRFWIN -type 'SQR Report' -runcontrol 1 -runlocationdescr PSNT
```

Configuring the job status mapping policy

IBM Workload Scheduler calculates the status of an IBM Workload Scheduler job based on the PeopleSoft job Run Status and Distribution Status. In PeopleSoft, the run status monitors the running of the job until it reaches a final status; the distribution status monitors the status of the output of the job. If the final status of a PeopleSoft job is neither success nor warning, IBM Workload Scheduler ignores the distribution status and the IBM Workload Scheduler job status is ABEND.

If the final status of a PeopleSoft job is success or warning, you can decide whether to use the distribution status of the PeopleSoft job when determining the status of the IBM Workload Scheduler job by setting the PS_DISTSTATUS option in the options file:

0

The distribution status is ignored and the IBM Workload Scheduler job status is calculated as shown in [Table 15: Relationship between the run status and the IBM Workload Scheduler job status on page 200](#).

1

The distribution status is used and the IBM Workload Scheduler job status is calculated as shown in [Table 14: Relationship between the run status, the distribution status, and the IBM Workload Scheduler job status on page 199](#). This is the default value.

[Table 14: Relationship between the run status, the distribution status, and the IBM Workload Scheduler job status on page 199](#) shows the relationship between the run status, the distribution status, and the IBM Workload Scheduler job status. The return code associated with the status is shown in parentheses. IBM Workload Scheduler uses this return code to evaluate the return code condition you specified in the **Return Code Mapping Expression** field in the Properties panel of the job definition. For more details about this field, refer to the online help by clicking the "?" in the top-right corner of the panel.

Table 14. Relationship between the run status, the distribution status, and the IBM Workload Scheduler job status

PeopleSoft job run status	PeopleSoft job distribution status	IBM Workload Scheduler job status
• Success (9) • Warning (17)	• Posted (5) • None (0)	SUCC
• Success (9) • Warning (17)	• Not Posted (4) • Delete (6)	ABEND
• Success (9) • Warning (17)	• Not Available (1) • Processing (2) • Generated (3) • Posting (7)	EXEC
• Cancel (1) • Delete (2) • Error (3) • Canceled (8) • No Success (10) • Blocked (18) • Restart (19)	Any distribution status	ABEND

[Table 15: Relationship between the run status and the IBM Workload Scheduler job status on page 200](#) shows the relationship between the PeopleSoft run status and the IBM Workload Scheduler job status. The return code associated with the status is shown in parentheses. IBM Workload Scheduler uses this return code to evaluate the return code condition you

specified in the **Return Code Mapping Expression** field in the Properties panel of the job definition. For more details about this field, refer to the online help by clicking the "?" in the top-right corner of the panel.

Table 15. Relationship between the run status and the IBM Workload Scheduler job status

PeopleSoft final run status	IBM Workload Scheduler status
Cancel (1)	ABEND
Delete (2)	ABEND
Error (3)	ABEND
Hold (4)	WAIT
Queued (5)	WAIT
Initiated (6)	INIT
Processing (7)	EXEC
Canceled (8)	ABEND
Success (9)	SUCC
No Success (10)	ABEND
Pending (16)	EXEC
Warning (17)	SUCC
Blocked (18)	ABEND
Restart (19)	ABEND



Note: If IBM Workload Scheduler fails to retrieve the status of the PeopleSoft job, the IBM Workload Scheduler job status is **DONE**.

Common serviceability for the access methods

This section provides information common to all the access methods including return code mapping, configuring the tracing utility, and troubleshooting the access method.

The return code mapping feature

The return code mapping feature provides a standard way of mapping messages into return code values. You can also customize the return code mapping. This feature is available for the following access methods:

- PeopleSoft
- SAP

The return code mapping feature provides more granularity when defining the success or failure policies of jobs and improved flexibility in controlling job execution flows based on execution results. Job return code mapping provides the following capabilities:

- Users can define a job final status (successful or failed) based on a condition on the return code of the execution of the program or script of the job.
- The return code can be provided also to the recovery job that is associated with it in the job definition. This causes the recovery job to perform different processing based on the return code.

Parameters

#

Optional comment. All the lines starting with this symbol (#) are not used for mapping.

patternn

Pattern strings delimited by quotation marks (" and "). If you use only one pattern string, you can omit the quotation marks. If the pattern string contains a quotation marks character, then it must be escaped by backslash (\). The string can contain the following wildcards and special characters:

Asterisk (*)

Matches an arbitrary number of characters.

Question mark (?)

Matches a single character.

Backslash (\)

Escape character.

RC value

The return code value. This value is sent by the method to IBM Workload Scheduler by a `%RC nnnn` message.

Creating a return code mapping file

You can create a return code mapping file to customize your own return codes with respect to certain conditions that might affect a job when it runs. Use this file to set the success condition of the job, which IBM Workload Scheduler uses to assess if the job completes successfully or in error. The return code is sent to IBM Workload Scheduler in the form of a `%RC nnnn` message. If this message is received, the job state is updated accordingly.

Each method has its own set of files to map the messages into return code values. The mapping files can be either global or local for a workstation.

Return code mapping files that are specific to a workstation are named according to the following scheme:

```
TWA_DATA_DIR/methods/rcm/accessmethod-type-workstation.rcm
```

```
TWA_home\methods\rcm\accessmethod-type-workstation.rcm
```

Global mapping files have a file name according to the following scheme:

```
TWA_DATA_DIR/methods/rcm/accessmethod-type.rcm
```

```
TWA_home\methods\rcm\accessmethod-type.rcm
```

For the PeopleSoft access method, *type* is always equal to `rcmap`. For the SAP R/3 access method, *type* is as described in [Return code mapping file names for r3batch on page 205](#).

Syntax

About this task

Use the following syntax to create the return code mapping file:

```
[#] "pattern1? "pattern2"... "patternn" = RC value
```

Examples

The following is an example of a return code mapping file. The line numbers in bold do not belong to the file, but are shown for reference:

```
1. # This is an RC mapping file for joblog.
2.
3. "User * missing = 102
4. "\*\*\*? = 103
5. "User \
6. * \
7. missing = 102
```

In this example:

- Line **1** is a comment and is not used for mapping.
- Line **2** is blank and is ignored. All blanks preceding or following a pattern string are ignored, as well as those between the equals sign and the return code value.
- Line **3** matches every message starting with the string *User* and ending with the string *missing*.
- Line **4** matches every message starting with three asterisks (*) followed by a blank. When you use the asterisk in this way and not as a wildcard, you must escape it with a backslash.
- Lines **5** through **7** contain a pattern taking several lines. It matches the same messages as the pattern of line **3**.

Considerations

Note the following facts:

- The order of the pattern lines is important because the first matching pattern line is used to build the return code value.
- Empty pattern strings ("") are ignored by the pattern matching procedure.

For example, the following is a valid pattern sequence. The first line is more restrictive than the second line.

```
"625 "User * missing = 104
" "User * missing = 102
```

The following pattern sequence is formally valid, but the second pattern line is never used. Because the first line is more general, it is always matched first.

```
" "User * missing = 102
"625 "User * missing = 104
```

Return code mapping for psagent

For the PeopleSoft access method, you can write return code mapping files associating the internal states listed in [Table 16: Job states and return codes for the PeopleSoft access method on page 203](#).

When no return code mapping files are defined, or when a string returned by the access method does not satisfy any of the matching patterns of the mapping file, the access method uses the respective standard return codes listed in the tables.

Table 16. Job states and return codes for the PeopleSoft access method

psagent job state	psagent return code
"CANCEL"	1
"DELETE"	2
"ERROR"	3
"HOLD"	4
"QUEUED"	5
"INITIATED"	6
"PROCESSING"	7
"CANCELED"	8
"SUCCESS"	9
"NO SUCCESSPOSTED"	10
"POSTED"	11
"NOT POSTED"	12
"RESEND"	13
"POSTING"	14
"GENERATED"	15

Return code mapping for r3batch

About this task

Using return code mapping with `r3batch` can be useful in overcoming differences in the return code mechanisms of R/3, which returns a mixture of messages and numbers, and of IBM Workload Scheduler, which handles exclusively numeric return codes. By customizing the return code mapping files listed in [Return code mapping file names for r3batch on page 205](#), you can map messages from R/3 logs, spool lists, and exceptions from RFC function modules into return code values that IBM Workload Scheduler can handle.

Note that when you do not use this feature, `r3batch` does not send any return codes to IBM Workload Scheduler. In this case, IBM Workload Scheduler displays only the `r3batch` exit code, which cannot be used to set up `rccondsucc` conditions.

The return code mapping mechanism works as follows:

1. `r3batch` reads the output retrieved from the R/3 system (R/3 job log, process chain log, spool list, and so on appended to the `stdlist` of the related IBM Workload Scheduler job).
2. Following your specifications in the `rcm` files, the R/3 return messages or codes are mapped into your custom return codes and passed on to IBM Workload Scheduler.
3. These return codes are used together with the `rccondsucc` keyword set in the extended agent job definition and handled accordingly. Return code mapping is meaningful only if you use the return codes to write the expressions that determine job completion. Conversely, in the case of this extended agent, the use of `rccondsucc` is significant only if IBM Workload Scheduler gets return codes (not exit codes) from the access method.

To use the return code mapping feature:

- Leave the value of the `rcmap` option as `ON` (this is the default).
- Depending on which R/3 logs you want `r3batch` to read and map, leave the default settings of the `retrieve_joblog`, `retrieve_pchainlog`, and `retrieve_spoolist` options as `ON` and manually create the corresponding `rcm` files.
- If you want to map messages from the R/3 syslog, set the `log_r3syslog` option to `ON` and manually create the corresponding `rcm` file.

When setting up your return code mapping for `r3batch`, consider the following:

- You can define any return code numbers for your use because there are no reserved return codes for the access method or for IBM Workload Scheduler.
- Mapping files are scanned sequentially: the first match found performs the corresponding mapping. When you define a mapping file, write the most restrictive strings first.
- When you define a mapping file, remember that the R/3 log messages are read in their entirety. If you want to map only a part of the entry, you must use the wildcard characters.
- If two lines match two different patterns, then the return code is set to the higher value. In general the return code is set to the highest value among the ones yielded by the matched patterns. This is shown in the following example:

The job log returned after job PAYT410 has run is:

```

*** ERROR 778 *** EEW00778E Failed to modify the job PAYT410 with job id
***      05710310.

*** ERROR 176 *** EEW00176E Failed to add step 1.

*** ERROR 552 *** EEW00552E The R/3 job scheduling system has found an
***      error for user name * and job name PAYT410. Please check R/3
***      syslog.

*** ERROR 118 *** EEW00118E Execution terminated. Could not create and
***      start an instance of the R/3 batch job.
ERROR LEVEL=118

```

and the system log contains the following line:

```
|011:05:12|MAESTRO|SAPMSSY1|EFT|> Step 1 contains illegal values
```

The `r3batch-joblog.rcm` file contains the following matching line:

```
"118"*"*=100
```

while the `r3batch-syslog.rcm` file contains the following matching line:

```
"*MAESTRO*Step 1 contains illegal values "=9999
```

In this case, the return code sent back to IBM Workload Scheduler is 9999 because it is the higher of the two matching patterns.

- If no matching takes place, no return code is sent to IBM Workload Scheduler.

Return code mapping file names for r3batch

`r3batch` uses the global return code mapping files listed below. You must manually create the `rcm` directory under the following path:

On UNIX operating systems

```
TWA_DATA_DIR/methods
```

On Windows operating systems

```
TWA_home\methods
```

You can create the mapping files you want to implement in the `rcm` directory:

r3batch-joblog.rcm

Maps messages from the R/3 job log of a job into return code values. If this file is not present, the messages in the job log are ignored.

The format of the mapping file is:

```

message_text_pattern
[program_pattern[message_number_pattern[message_id_pattern]]]=RCvalue

```

where `program_pattern` is the external program that produced the output shown in the job log and `message_id_pattern` is the message class. For example, the following line appended in the job log:

```

04/26/2005 10:08:04 00
550Step 001 started (program BTCTEST, variant VAR1, user name TWSDEV)

```

will match the following pattern line in `r3batch-joblog.rcm`:

```
"*Step*" "*" "550" "*"=5
```

because:

message_text_pattern

```
"Step 001 started (program BTCTEST, variant VAR1, user name TWSDEV)"
```

program_pattern

```
"*"
```

message_number_pattern

```
"550"
```

message_id_pattern

```
"*"
```

`r3batch-pchainlog.rcm`

Maps messages from the protocol of a Process Chain into return code values. If this file is not present, the messages in the protocol are ignored.

The format of the mapping file is:

```
message_number_pattern
[message_id_pattern[message_variable1[message_variable2
[message_variable3[message_variable4[message_type]]]]]=RCvalue
```

`r3batch-spoolist.rcm`

Maps messages in the job spool list of an R/3 job into return code values. If this file is not present, the messages in the spool list are ignored.

The format of the mapping file is:

```
spool_list_row_pattern=RCvalue
```

`r3batch-syslog.rcm`

Maps messages in the syslog of an R/3 system into return code values. The R/3 system log should be checked only when R/3 returns the generic 552 error to `r3batch`.

If this file is not present, the messages in the system log are ignored.

The format of the mapping file is:

```
system_log_row_pattern=RCvalue
```

If you plan to map system log messages, be sure to set the `log_r3syslog` option of `r3batch` to `ON` (the default is `OFF`).

r3batch-msgrc.rcm

Maps ABAP exceptions and BAPI return codes of RFC function modules into return code values. If this file is not present, the mapping is done using a hardcoded table.

The format of the mapping file is:

```
message_number=RCvalue
```

message_number is the error message number. The last message number is always used. That is, if two error messages are generated, only the second one is checked against the mapping file.

Mapping return codes for intercepted jobs

About this task

To set up return code mapping for intercepted jobs, after defining the appropriate return code conditions in the `r3batch-joblog.rcm` file, do the following:

1. Create a customized template file named `rctemplate.jdf` in the following directory:

```
TWA_DATA_DIR/methods/r3batch_icp/
```

```
TWA_home\methods\r3batch_icp\
```

The file must contain the following:

```
alias;rccondsucc "Success Condition"
```

where, the "Success Condition" must match a condition saved in the `rcm` file.

2. Modify the `XANAME_r3batch.icp` file, located in the same path, to refer to the `jdf` file you created in the previous step as follows:

```
client job_mask user_mask rctemplate
```

IBM Workload Scheduler manages the intercepted R/3 job as a docommand job with all the options specified in the customized `jdf` file. You can check if your intercepted job is correctly submitted by reading the `job_interceptor.joblog`.

Configuring the tracing utility

Learn how to configure the trace utility for all the access methods.

IBM Workload Scheduler logs all the processing information in the following configuration file:

```
TWA_DATA_DIR/methods/accessmethod.properties
```

```
TWA_home\methods\accessmethod.properties
```



Note: If you delete this file accidentally, IBM Workload Scheduler creates a new file with all the default values and contains the following comment:

```
# This file was automatically created using the default values.
```

Customizing the .properties file

About this task

Depending on the access method you are working with, customize the trace parameters in the following properties files:

psagent.properties

For the PeopleSoft access method.

r3batch.properties, r3evmon.properties

For the SAP access method.

With this access method, you can also specify debug and trace parameters in the single job definitions. See [Creating SAP Standard R/3 jobs from the Dynamic Workload Console on page 254](#) and [Task string to define SAP jobs on page 262](#).

For each .properties file you can customize the following parameters:

accessmethod.trace.tracers.level

Specify the level of tracing you want to set. Possible values are:

DEBUG_MIN

Only error messages are written in the trace file. This is the default.

DEBUG_MID

Informational messages and warnings are also written in the trace file.

DEBUG_MAX

A most verbose debug output is written in the trace file.

The value you set in the .properties file applies to all the jobs of the corresponding access method. To specify a different trace setting for a particular job, specify the following option in the job definition:

```
-tracelvl=(1|2|3)
```

where:

- 1 = DEBUG_MIN
- 2 = DEBUG_MID
- 3 = DEBUG_MAX



Note: When making changes to the trace level setting, the changes are effective immediately after saving the .properties file. No restart is required.

`accessmethod.trace.handlers.traceFile.fileDir`

Specifies the path where the trace file is created. Depending on the access method, the default is:

SAP

On UNIX® operating systems

`TWA_DATA_DIR/methods/traces`

On Windows™ operating systems

`TWA_home\methods\traces`

All other access methods

On UNIX operating systems

`TWA_DATA_DIR/methods`

On Windows operating systems

`TWA_home\methods`

Ensure that the new path you specify has already been created as a fully qualified path with write permissions.

Traces are written in XML format. Always use slashes (or backslashes) when you specify a new path, even if you are working on Windows™ operating systems.

The trace files give information about the method execution to the desired level of detail. The minimum trace level is always on, to guarantee a First-Failure Data Capture (FFDC) ability. The trace file name is:

`trace-psagent.log`

For the PeopleSoft access method.

`trace-r3batch.log`, `trace-XAname-r3evmon.log`

For the SAP access method.

`accessmethod.trace.tracers.logging`

Specifies to enable or disable the trace utility. Possible values are:

true

To enable the trace utility. This is the default value.

false

To disable the trace utility. If you set this parameter to **false**, no traces are written in the `trace-accessmethod.log` file even if there are problems.

r3batch.trace.handlers.traceFile.maxFiles

The maximum number of trace files that are created before the oldest one is deleted. If this parameter is set to 1, the current trace file is never replaced and can grow without limit.

r3batch.trace.handlers.traceFile.maxFileSize

The maximum size (in bytes) that the trace file can reach before it is renamed and a new trace file is created. This parameter is valid only if the `r3batch.trace.handlers.traceFile.maxFiles` is set to a value greater than 1.

Configuration file example for the SAP access method

The following `r3batch.properties` file is an example of a configuration file for the SAP access method with the following characteristics:

- The level of tracing set is `DEBUG_MID`. This means that not only error messages but also informational messages and warnings are written in the trace file.
- The trace file is created in the `/home/maestro/methods` directory.
- The tracing process creates three trace files, whose maximum size can be 10 MB.

```
r3batch.organization=ABC
r3batch.product=IWS
r3batch.component=R3BATCH
r3batch.trace.tracers.level=DEBUG_MID
r3batch.trace.tracers.listenerNames=r3batch.trace.handlers.traceFile
r3batch.trace.tracers.logging=true
r3batch.trace.handlers.traceFile.fileDir=/home/maestro/methods
r3batch.trace.handlers.traceFile.formatterName=r3batch.trace.formatter
r3batch.trace.handlers.traceFile.maxFileSize=104805100
r3batch.trace.handlers.traceFile.maxFiles=3
```

Integration with SAP

The following sections give you information about IBM® Workload Scheduler for SAP, the SAP access method and job plug-ins, and how to schedule jobs by using the SAP Solution Manager.

IBM® Workload Scheduler is certified for the following SAP integrations:

- SAP Certified Integration with SAP NetWeaver
- SAP Certified Integration with SAP S/4HANA

Through the integrations, IBM® Workload Scheduler can access SAP BW on HANA Process Chains to handle and manage them externally and can invoke and track InfoPackages.

Introducing IBM Workload Scheduler for SAP

Improve SAP operations and enable business growth with IBM Workload Scheduler.

Use IBM Workload Scheduler for SAP, to create, schedule, and control SAP jobs using the job scheduling features of IBM Workload Scheduler. IBM Workload Scheduler supported agent workstations help extend the product scheduling capabilities

to SAP through the R/3 batch access method . In addition, you can define IBM Workload Scheduler job plug-ins for SAP BusinessObjects BI and SAP PI Channel. With the SAP Solution Manager integration, you can have the IBM Workload Scheduler engine run job scheduling tasks available from the Solution Manager user interface.

IBM Workload Scheduler provides a single and simplified point of planning, control and optimization of end-to-end production services across heterogeneous IT infrastructures. It enables you to control SAP operations from z/OS indifferently.

To understand if a SAP System is compatible with IBM Workload Scheduler, check if your system exposes the following interfaces:

- BC-XBP 6.10 (V2.0) - Background Processing
- BC-XBP 7.00 (V3.0) - Background Processing
- BW-SCH 3.0 – Business Information Warehouse for SAP BW/4HANA
- BW4-SCH - Business Information Warehouse for SAP BW/4HANA

For detailed information about supported SAP interfaces, prerequisite SAP notes, and supported SAP software versions, see the [Data Integration](#) report and select the **Supported Software** tab. In addition, see the dedicated SAP section in the [IBM Workload Scheduler Detailed System Requirements](#).

Features

[Table 17: IBM Workload Scheduler for SAP features on page 211](#) shows the tasks you can perform with IBM Workload Scheduler for SAP either in a distributed or an end-to-end environment, or both.

Table 17. IBM Workload Scheduler for SAP features

Feature	Distributed environment	End-to-end
Using IBM Workload Scheduler standard job dependencies and controls on SAP jobs	✓	✓
Listing jobs, defining jobs, variants, and extended variants using the IBM Workload Scheduler interface	✓	✓
Defining jobs and variants dynamically at run time	✓	✓
Scheduling SAP jobs to run on specified days and times, and in a prescribed order	✓	✓
Scheduling SAP BusinessObjects Business Intelligence (BI) jobs to gain greater control over your SAP BusinessObjects Business Intelligence (BI) reports through the IBM Workload Scheduler plug-in for SAP BusinessObjects Business Intelligence (BI).	✓	✓
Scheduling SAP Process Integration (PI) Channel jobs to control communication channels between the Process Integrator and a backend SAP R/3 system.	✓	✓

Table 17. IBM Workload Scheduler for SAP features (continued)

Feature	Distributed environment	End-to-end
Scheduling and monitoring job scheduling tasks available from the SAP Solution Manager user interface.	✓	
Defining the national language support options	✓	✓
Using the SAP Business Warehouse Support functions	✓	✓
Customizing job execution return codes	✓	✓
Using SAP logon groups for load balancing and fault-tolerance	✓	✓
Using Business Component-eXternal Interface Background Processing (XBP 2.0 and later) interface support to:	• Collect intercepted jobs • Track child jobs • Keep all job attributes when you rerun a job • Raise events	• Track child jobs • Keep all job attributes when you rerun a job • Raise events
Using Business Component-eXternal Interface Background Processing (XBP 3.0) interface support to:	• Create criteria profiles to log raised events, reorganize the event history, and intercept and relaunch jobs, according to the criteria you specify. • SAP application log and application return code • Spool list request and display for jobs that have run. • Temporary variants	• Create criteria profiles to log raised events, reorganize the event history, and intercept and relaunch jobs, according to the criteria you specify. • SAP application log and application return code • Spool list request and display for jobs that have run. • Temporary variants
Assigning an SAP job to a server group, for batch processing	✓	✓

Table 17. IBM Workload Scheduler for SAP features (continued)

Feature	Distributed environment	End-to-end
Exporting SAP factory calendars and adding their definitions to the IBM Workload Scheduler database	✓	
Defining internetwork dependencies and event rules for IBM Workload Scheduler based on SAP events	✓	
Defining event rules based on IDoc records	✓	
Defining event rules based on CCMS Monitoring Architecture alerts	✓	
Rerunning a job that submits a process chain from a specific process, from failed processes, or as a new instance	✓	✓
Displaying the details of a job that submits a process chain	✓	✓
Enabling job throttling	✓	✓

Access method for SAP

The SAP R/3 batch access method enables communication between an external SAP R/3 system and IBM Workload Scheduler and provides a single point of entry for automating the launching of jobs, monitoring the status of jobs and managing exceptions and recovery.

Using the SAP access method you can run and monitor SAP jobs from the IBM Workload Scheduler environment. These jobs can be run as part of a schedule or submitted for ad-hoc job processing. SAP extended agent or dynamic agent jobs can have all of the same dependencies and recovery options as other IBM Workload Scheduler jobs. SAP jobs must be defined in IBM Workload Scheduler to be run and managed in the IBM Workload Scheduler environment.

IBM Workload Scheduler provides a single and simplified point of planning, control and optimization of end-to-end production services across heterogeneous IT infrastructures. It enables you to control SAP operations from z/OS indifferently.

Scheduling process for the agent workstation hosting the r3batch access method

IBM Workload Scheduler launches jobs in SAP using IBM Workload Scheduler jobs defined to run on a supported agent workstation.

Supported agent workstations include:

- dynamic agents
- extended agents
- IBM Z Workload Scheduler Agents

The supported agent workstations use the access method, `r3batch`, to pass SAP job-specific information to predefined SAP instances. The access method uses information provided in an options file to connect and launch jobs on an SAP instance.

Multiple extended agent workstations can be defined to use the same host, by using multiple options files. Using the SAP extended agent workstation unique identifier as a key, `r3batch` uses the corresponding options file to determine which instance of SAP will run the job. See [UNIQUE_ID on page 177](#) for more information about identifying the extended agent workstation unique identifier. It makes a copy of a template job in SAP and marks the job as "scheduled". It then monitors the job through to completion, writing job progress and status information to a job standard list on the host workstation.

On dynamic agent workstations, more than one options file can be associated to the workstation.

For more information about job management, refer to the *IBM Workload Scheduler: User's Guide and Reference*.

For more detailed information about configuration files on extended agents and dynamic agents, see [Configuring the SAP access method on page 229](#).

Roles and responsibilities

In a typical enterprise, different users contribute to the implementation and operation of the product. [Table 18: Roles and responsibilities in IBM Workload Scheduler for SAP on page 214](#) describes the roles and responsibilities of all users in the process model, showing the tasks they perform.

Table 18. Roles and responsibilities in IBM Workload Scheduler for SAP

User role	User task
IBM Workload Scheduler administrator	• Creating the IBM Workload Scheduler RFC user on page 217 • Creating the authorization profile for the IBM Workload Scheduler user on page 217 • Copying the correction and transport files on page 220 • Importing ABAP/4 function modules into SAP on page 221
IBM Workload Scheduler configurator	• Changing the IBM Workload Scheduler RFC user ID password on page 225 • Migrating from previous versions on page 228 • Print parameter and job class issues on page 227 • Defining the configuration options on page 231 • Connecting to the SAP system on page 250 • Creating SAP Standard R/3 jobs from the Dynamic Workload Console on page 254 • Using the BDC Wait option on page 304 • Implementing job interception on page 306 • Defining user authorizations to manage SAP Business Warehouse InfoPackages and process chains on page 316 • Setting and using job throttling on page 334 • Exporting SAP factory calendars on page 339 • Setting National Language support options on page 374

Table 18. Roles and responsibilities in IBM Workload Scheduler for SAP (continued)

User role	User task
IBM Workload Scheduler developer	• Editing a standard SAP job on page 261 • Task string to define SAP jobs on page 262 • Displaying details about a standard SAP job on page 274 • Verifying the status of a standard SAP job on page 274 • Deleting a standard SAP job from the SAP database on page 275 • Balancing SAP workload using server groups on page 276 • Defining SAP jobs dynamically on page 282 • Managing SAP Business Warehouse InfoPackages and process chains on page 317
IBM Workload Scheduler developer	• Defining internetwork dependencies and event rules based on SAP background events on page 342 • Defining event rules based on IDoc records on page 351 • Defining event rules based on CCMS Monitoring Architecture alerts on page 361
IBM Workload Scheduler operator	• Rerunning a standard SAP job on page 280 • Mapping between IBM Workload Scheduler and SAP job states on page 276 • Raising an SAP event on page 278 • Killing an SAP job instance on page 277 • Displaying details about a process chain job on page 324

Configuring user authorization (Security file)

IBM Workload Scheduler manages security through the use of a configuration file, the security file. In the security file, you specify which scheduling objects a user can manage and how. You define these settings by writing user definitions. A user definition is an association between a name and a set of users, the objects they can access, and the actions they can perform on the specified objects.

For more detailed information about the security file, security file syntax, and how to configure the security file, see "Configuring user authorization (Security file)" in the *Administration Guide*.

The following table displays the access keywords required to grant authorization to access and work with SAP scheduling objects assigned to IBM Workload Scheduler users.

Table 19. Access keywords for activities with SAP scheduling objects

Activity		Access keywords required
Dynamic Workload Console	Define or search for SAP jobs on an extended agent workstation.	display on the workstation
	Retrieve the spool list on an extended agent workstation.	display on the workstation
	Rerun from a step on an extended agent.	rerun on the job
	Define or search for SAP jobs on a dynamic agent workstation, pool, or dynamic pool.	display and run on the workstation
	Retrieve the spool list on a dynamic agent workstation, pool, or dynamic pool.	display and run on the job
	Rerun from a step on a dynamic agent workstation, pool, or dynamic pool.	rerun on the job

Configuring the SAP environment

You must configure the SAP environment before using the SAP access method.

To communicate and manage the running of jobs on SAP systems using the access method for SAP, complete the following configuration steps in the SAP environment.

The steps require that you have knowledge of an SAP Basis Administrator.

Overview

About this task

Here is an overview of the customization procedure:

1. Create a new user ID for RFC communications in SAP for IBM Workload Scheduler.
2. Create the authorization profile as described in [Creating the authorization profile for the IBM Workload Scheduler user on page 217](#).
3. Copy the correction and transport files from the IBM Workload Scheduler server to the SAP server.
4. Import the correction and transport files into SAP and verify the installation.

Results



Note: The import procedure adds new ABAP/4 function modules and several new internal tables to the SAP system. It does not modify any of the existing objects.

Creating the IBM Workload Scheduler RFC user

About this task

For IBM Workload Scheduler to communicate with SAP, you must create a user ID in SAP for IBM Workload Scheduler batch processing. For security reasons, use a new user ID rather than an existing one.

1. Create a new RFC user ID.
2. Give this new RFC user ID the following attributes:
 - A user type of **CPIC**, **Communications**, or **DIALOG**, depending on the SAP release.
 - A password at least six characters in length. IBM Workload Scheduler requires this password to start or monitor SAP jobs. If this password changes in SAP, you must update the options file used by `r3batch` with the new password.
 - The appropriate security profiles, depending on your version of SAP.

Creating the authorization profile for the IBM Workload Scheduler user

The two ways to create the authorization profile for the IBM Workload Scheduler user.

There are two alternative ways to perform this task:

- Using transaction `SU02` and manually creating the profile.
- Using the Profile Generator (transaction `PFCG`).

Using transaction su02 and manually creating the profile

About this task

Perform the following steps:

1. Write a profile name, for example `Z_TWS`, and a description.
2. Manually add the authorizations according to the following table:

Object	Description	Authorization
S_ADMI_FCD	System authorizations	S_ADMI_ALL
S_APPL_LOG	Application logs	S_APPL_L_E2E
S_BTCH_ADM	Background processing: Background administrator	S_BTCH_ADM

Object	Description	Authorization
S_BTCH_JOB	Background processing: Operations on background jobs	S_BTCH_ALL
S_BTCH_NAM	Background processing: Background user name	S_BTCH_ALL
S_DEVELOP	ABAP Workbench: full authorization to modify objects of type PROG	E_ABAP_ALL
S_LOG_COM	Authorization to run external commands	S_LOGCOM_ALL
S_PROGRAM	ABAP: program run checks	S_ABAP_ALL
S_RFC	Authoritation. check for RFC access	S_RFC_ALL
S_RZL_ADM	CCMS: System Administration	S_RZL_ALL
S_SPO_ACT	Spool: Actions	S_SPO_ALL
S_SPO_DEV	Spool: Device authorizations	S_SPO_DEV_AL
S_XMI_LOG	Internal access authorizations for XMI log	S_XMILOG_ADM
S_XMI_PROD	Authorization for external management interfaces (XMI)	S_XMI_ADMIN

The authorizations are located in the "Basis: Administration" object class.

Depending on the version of SAP, the authorization S_RFC_ALL are located either in the "Cross-application Authorization Objects" or in the "Non-application-specific Authorization Objects" object class.

3. Save the profile.
4. Go to the user maintenance panel and assign the profile to the IBM Workload Scheduler SAP user.
5. Save the user data.

Using transaction PFCG (Profile Generator)

About this task

Perform the following steps:

1. Write a name, for example ZTWS, in **Role Name**.
2. Click **Create Role** and write a description for the role, such as "Role for the TWS user."
3. Save the role.
4. Select **Authorizations**.
5. Click **Change Authorization Data**.

6. In the pop-up, select **Templates**.

7. Manually add the following authorization objects:

Object	Description
S_ADMI_FCD	System authorizations
S_APPL_LOG	Application logs
S_BTCH_ADM	Background processing: Background administrator
S_BTCH_JOB	Background processing: Operations on background jobs
S_BTCH_NAM	Background processing: Background user name
S_PROGRAM	ABAP: Program run checks
S_DEVELOP	ABAP Workbench: full authorization to modify objects of type PROG
S_LOG_COM	Authorization to run external commands
S_RFC	Authorization check for RFC access
S_RZL_ADM	CCMS: System Administration
S_SPO_ACT	Spool: Actions
S_SPO_DEV	Spool: Device authorizations
S_XMI_LOG	Internal access authorizations for XMI log
S_XMI_PROD	Authorization for external management interfaces (XMI)

8. Fill in the values according to the following scheme:

Object	Description
S_ADMI_FCD	System authorizations ◦ System administration function: Full authorization
S_APPL_LOG	Activity: Display ◦ Application log Object name: Full authorization ◦ Application log subobject: Full authorization
S_BTCH_ADM	Background processing: Background administrator ◦ Background administrator ID: Full authorization
S_BTCH_JOB	Background processing: Operations on background jobs ◦ Job operations: Full authorization ◦ Summary of jobs for a group: Full authorization
S_BTCH_NAM	Background processing: Background user name

Object	Description
	◦ Background user name for authorization check: Full authorization
S_PROGRAM	ABAP: Program run checks ◦ User action ABAP/4 program: Full authorization ◦ Authorization group ABAP/4 program: Full authorization
S_RFC	Authorization check for RFC access ◦ Activity: Full authorization ◦ Name of RFC to be protected: Full authorization ◦ Type of RFC object to be protected: Full authorization
S_RZL_ADM	Activity: Full authorization
S_SPO_ACT	Spool: Actions ◦ Authorization field for spool actions: Full authorization ◦ Value for authorization check: Full authorization
S_SPO_DEV	Spool: Device authorizations ◦ Spool - Long device names: Full authorization
S_XMI_LOG	Internal access authorizations for XMI log ◦ Access method for XMI log: Full authorization
S_XMI_PROD	Authorization for external management interfaces (XMI) ◦ XMI logging - Company name: ABC* ◦ XMI logging - Program name: MAESTRO* ◦ Interface ID: Full authorization

9. Save the authorizations.
10. Generate a profile. Use the same name that you wrote in **Role Name**.
11. Exit the authorization management panel and select **User**.
12. Add the IBM Workload Scheduler user to the role.
13. Save the role.

Copying the correction and transport files

About this task

The setup file loads four correction and transport files into the IBM Workload Scheduler home directory. Copy these correction and transport files to the SAP server and import them into the SAP database, as follows:

1. On your SAP database server, log on to the SAP system as an administrator.
2. Copy the control file and data file from the methods directory and to the following directories on your SAP database server:

```
copy control_file /usr/sap/trans/cofiles/
copy data_file /usr/sap/trans/data/
```

The names of `control_file` and `data_file` vary from release to release. The files are located in the `methods` directory: UNIX®: `TWA_DATA_DIR\methods`, Windows™: `TWA_home\methods`, and have the following file names and format:

For SAP releases earlier than 6.10:

- `K000xxx.TV1` (control file) and `R000xxx.TV1` (data file)
- `K900xxx.TV2` (control file) and `R900xxx.TV2` (data file)

For SAP releases 6.10, or later:

- `K9000xx.TV1` (control file) and `R9000xx.TV1` (data file)
- `K9007xx.TV1` (control file) and `R9007xx.TV1` (data file)

where `x` is a digit generated by the SAP system.

Specifically, for IBM Workload Scheduler version 10.2.6 the following files are used:

For SAP releases earlier than 6.10:

- `K000538.TV1` (for standard jobs scheduling)
- `R000538.TV1` (for standard jobs scheduling)
- `K900294.TV2` (for IDoc monitoring and job throttling)
- `R900294.TV2` (for IDoc monitoring and job throttling)

For SAP releases 6.10, or later:

- `K900044.TV1` (for standard jobs scheduling)
- `R900044.TV1` (for standard jobs scheduling)
- `K900751.TV1` (for IDoc monitoring and job throttling)
- `R900751.TV1` (for IDoc monitoring and job throttling)

Importing ABAP/4 function modules into SAP

How to generate, activate and commit new ABAP/4 modules to a SAP system.

About this task

This section describes the procedure to generate, activate, and commit new ABAP/4 function modules to your SAP system and several new internal tables. You do not modify any existing SAP system objects. For information about the supported SAP R/3 releases, see the System Requirements Document at Download Documents, System Requirements, Release Notes.

The number of ABAP/4 modules that you install with the import process varies from release to release. The modules are installed in the `methods` directory and have the following file names and format:

- `K9000xx.TV1` (function modules for standard jobs scheduling extensions)
- `K9007xx.TV1` (function modules for IDoc monitoring and job throttling)

where *x* is a digit generated by the SAP system. The methods directory is located in:

On UNIX operating systems

```
TWA_DATA_DIR/methods
```

On Windows operating systems

```
TWA_home\methods
```

Before importing the ABAP/4 function modules, review the considerations documented in [Migrating from previous versions on page 228](#).

To import ABAP/4 function modules into SAP:

1. Change to the following directory:

```
cd /usr/sap/trans/bin
```

2. Add the transport file to the buffer:

```
tp addtobuffer transport sid
```

where:

transport

The transport request file.

sid

The SAP system ID.

For example, if the transport file in the `TWA_home\methods` directory is named `K9000xxx.TV1`, the transport request is `tv1K9000xxx`.

3. Run the **tp tst** command to test the import:

```
tp tst transport sid
```

After running this command, examine the log files in the `/usr/sap/trans/log` directory for error messages. Warnings of severity level 4 are normal.

If there are errors, check with a person experienced in correction and transport, or try using unconditional modes to do the import.

4. Run the following command to import all the files in the buffer:

```
tp import transport sid
```

This command generates the new ABAP/4 modules and commits them to the SAP database. They automatically become active.

After running this command, examine the log files located in the `/user/sap/trans/log` directory for error messages. Warnings of severity level 4 are normal.

If a problem is encountered, use unconditional mode when running this step:

```
tp import transport sid U126
```

5. When the import is complete, check the log files located in the `/usr/sap/trans/log` directory to verify that the ABAP/4 modules were imported successfully.

If you apply the standard transport and the IDOC transport, 26 ABAP/4 modules are installed by the import process. For a list of the transport files to be used, refer to [Importing ABAP/4 function modules into SAP on page 221](#). [Table 20: ABAP/4 modules installed on page 223](#) lists the ABAP modules installed.

Table 20. ABAP/4 modules installed

ABAP/4 module	Installed?
ENQUEUE_/IBMTWS/EQ_XAPPL	✓
DEQUEUE_/IBMTWS/EQ_XAPPL	✓
/IBMTWS/UNREGISTER_XAPPL	✓
/IBMTWS/GET_XAPPL_REGISTRATION	✓
/IBMTWS/MODIFY_JOB_CLASS	✓
/IBMTWS/REGISTER_XAPPL	✓
J_101_BDC_STATUS	✓
J_101_DATE_TIME	✓
J_101_IDOC_SELECT	✓
J_101_JOB_ADJUST_CLIENT	✓
J_101_JOB_FIND	✓
J_101_JOB_FINDALL	✓
J_101_JOB_HAS_EXTENDED_VARIANT	✓
J_101_JOB_LOG	✓
J_101_RAISE_EVENT	✓
J_101_REPORT_ALL_SELECTIONS	✓
J_101_REPORT_GET_TEXTPOOL	✓
J_101_VARIANT_COPY	✓
J_101_VARIANT_CREATE	✓
J_101_VARIANT_DELETE	✓

ABAP/4 module	Installed?
J_101_VARIANT_EXISTS	✓
J_101_VARIANT_GET_DEFINITION	✓
J_101_VARIANT_GET_HELP_VALUES	✓
J_101_VARIANT_MAINTAIN_CNT_TBL	✓
J_101_VARIANT_MAINTAIN_SEL_TBL	✓
J_101_VARIANT_MODIFY	✓

Table 21: ABAP/4 modules contents on page 224 shows the contents of the ABAP modules for the IDoc records and job throttling feature.

Table 21. ABAP/4 modules contents

Object	Description	Used by...
/IBMTWS/	Type = Development Namespace. For IBM Workload Scheduler.	Internal use only
/IBMTWS/EQ_XAPPL	Type = Lock Object. Synchronizes the job throttler instances and job interception collector jobs that are running against the same SAP system.	Job throttling Job interception
/IBMTWS/GET_XAPPL_REGISTRATION	Type = Function Module. It is used to query for existing external application registration data in table IBMTWS/XAPPL, for example the registration data of a job throttler instance or job interception collector.	Job throttling Job interception
/IBMTWS/MODIFY_JOB_CLASS	Type = Function Module. Modifies the job class of an intercepted job that is controlled by the job throttler. For details, see Step 3. Enabling job class inheritance on page 335 .	Job throttling Job interception
/IBMTWS/REGISTER_XAPPL	Type = Function Module. Registers an external application, for example the job throttler.	Job throttling Job interception
/IBMTWS/TWS4APPS	Type = Function group. For IBM Workload Scheduler.	Internal use only
/IBMTWS/UNREGISTER_XAPPL	Type = Function Module. Unregisters an external application, for example the job throttler.	Job throttling Job interception
/IBMTWS/XAPPL	Type = Table. Stores the registration data of external applications. An external application can be a job throttler instance or a job interception collector.	Job throttling Job interception

Object	Description	Used by...
J_101_IDOC_SELECT	Type = Function Module. Selects IDoc records from SAP internal tables. For details, see Defining event rules based on IDoc records on page 351 .	IDoc event rules
J_101_TWS_EDIDC	Type = Data structure in FM interface	Function module J_101_IDOC_SELECT
J_101_TWS_IDOC_SELECTION	Type = Data structure in FM interface	Function module J_101_IDOC_SELECT
J_101_TWS_STATE_SELECTION	Type = Data structure in FM interface	Function module J_101_IDOC_SELECT

To uninstall the transport you can use the STMS transaction see: [Deleting Imported Requests from the Import Queue](#).

Changing the IBM Workload Scheduler RFC user ID password

About this task

If the password of the IBM Workload Scheduler Remote Function Call (RFC) user ID is modified after the initial installation, the options file used by `r3batch` must be updated to reflect this change.

On UNIX® operating systems, log on as root to the system where IBM Workload Scheduler is installed.

On Windows™ operating systems, log on as an administrator and start a DOS shell on the system where IBM Workload Scheduler is installed.

1. Generate an encrypted version of the new password using the `enigma` command, located in `TWA_home/methods`.

To do this in a command shell, type:

```
enigma new_password
```

where

new_password

is the new password for the IBM Workload Scheduler RFC user ID.

The `enigma` command prints an encrypted version of the password.

2. Copy the encrypted password into the options file, which is located in the following directory:

On UNIX operating systems

```
TWA_DATA_DIR/methods
```

On Windows operating systems

```
TWA_home\methods
```

The file can be edited with any text editor.

Results

Ensure that you copy the password exactly, preserving uppercase, lowercase, and punctuation. The encrypted password looks similar to:

```
{aes}hyb/LQNyVzIf9oA8/xgY+CSqAuAh7+CvTT7HuDpdïu5YUAH0KJEHJtA=
```

If the encrypted password is not entered correctly, IBM Workload Scheduler is unable to start or monitor SAP batch jobs.

Securing data communication

You can increase the security of your SAP system through the use of an external security product. Secure Network Communications (SNC) can integrate the external security product with the SAP system.

Data communication paths between the client and server components of the SAP system that use the SAP protocols RFC or DIAG are more secure with SNC. The security is strengthened through the use of additional security functions provided by an external product that are otherwise not available with SAP systems.

SNC provides security at application level and also end-to-end security. IBM® Workload Scheduler is extended to read SNC configuration parameters and forward them to the SAP RFC communication layer used when logging in to the SAP system. IBM® Workload Scheduler does not provide or ship SNC software but instead enables the use of third-party SNC products to secure the RFC communication.

Levels of protection

You can apply one of the following levels of protection:

Authentication only

This is the minimum level of security protection available with SNC. The system verifies the identity of the communication partners.

Integrity protection

The system detects any changes to the data which might have occurred between the two components communicating.

Privacy protection

This is the maximum level of security protection available with SNC. The system encrypts the messages being transferred so that any attempt to eavesdrop is useless. Privacy protection also includes integrity protection of the data.

Example

The following options in the local options file are used to configure SNC for IBM Workload Scheduler:

- **r3snclib**: the path and file name of the SNC library.
- **r3sncmode**: enables or disables SNC between r3batch and the SAP R3 system.
- **r3sncmyname**: the name of the user sending the RFC for SNC.

- **r3sncpartnername**: the SNC name of the SAP R3 communication partner (application server).
- **r3sncqop**: the SNC protection level.

See [Defining the local options on page 234](#) for a description of these options in the local options file.

Print parameter and job class issues

The workstation running the r3batch access method for SAP uses the official RFC interfaces of SAP for job scheduling. When you migrate from previous versions of SAP, there can be problems with print parameters in jobs launched by IBM Workload Scheduler. This is because of limitations in the RFC interfaces.

These limitations are no longer true with XBP 2.0 and later.

The following is a list of print parameters supported by BAPI XBP 1.0 for SAP release 4.6x and later:

- archiving mode
- authorization
- columns
- delete after output
- lines
- number of copies
- output device
- print immediately
- recipient
- sap cover page
- selection cover page
- spool retention period

To resolve the loss of print parameters when copying a job, install the appropriate SAP Support Package as stated in the SAP notes 399449 and 430087.

The same applies to the job class. Official SAP interfaces only allow class C jobs. Installing the SAP Support Package also resolves this issue.

Unicode support

Access method for SAP supports the Unicode standard.

What is Unicode

Unicode was devised to address the problem caused by the profusion of code sets. Since the early days of computer programming hundreds of encodings have been developed, each for small groups of languages and special purposes. As a result, the interpretation of text, input, sorting, display, and storage depends on the knowledge of all the different types of character sets and their encodings. Programs are written to either handle one single encoding at a time and switch between them, or to convert between external and internal encodings.

The problem is that there is no single, authoritative source of precise definitions of many of the encodings and their names. Transferring text from one computer to another often causes some loss of information. Also, if a program has the code and the data to perform conversion between many subsets of traditional encodings, then it needs to hold several Megabytes of data.

Unicode provides a single character set that covers the languages of the world, and a small number of machine-friendly encoding forms and schemes to fit the needs of existing applications and protocols. It is designed for best interoperability with both ASCII and ISO-8859-1, the most widely used character sets, to make it easier for Unicode to be used in applications and protocols.

Unicode makes it possible to access and manipulate characters by unique numbers, their Unicode code points, and use older encodings only for input and output, if at all. The most widely used forms of Unicode are:

- UTF-32, with 32-bit code units, each storing a single code point. It is the most appropriate for encoding single characters.
- UTF-16, with one or two 16-bit code units for each code point. It is the default encoding for Unicode.
- UTF-8, with one to four 8-bit code units (bytes) for each code point. It is used mainly as a direct replacement for older MBCS (multiple byte character set) encodings.

Unicode support on SAP

Starting with SAP version 4.7 (R/3 Enterprise), Unicode is used on all layers of the SAP system:

- UTF-8, UTF-16, and UTF-32 on the database
- UTF-16 on the application server and graphical user interface

`r3batch` uses the UTF-8 code page internally. Because it communicates with SAP at the application server layer, it uses UTF-16 when communicating with Unicode-enabled SAP systems.

To use Unicode support, the following conditions must be met:

- Access method for SAP must run on a supported operating system. See the SAP support section in the [IBM Workload Scheduler Detailed System Requirements](#). The product does not support Unicode on the other operating systems where it can be installed.
- The SAP systems that communicate with `r3batch` must be running Unicode-enabled SAP versions.

If these conditions are not met, you cannot use Unicode support and must make sure that `r3batch`, the Dynamic Workload Console, and the target SAP system code page settings are aligned. Use the options related to national language support described in [SAP supported code pages on page 375](#).

Migrating from previous versions

This version of IBM Workload Scheduler for SAP supports all the SAP versions listed in [IBM Workload Scheduler Detailed System Requirements](#). The IBM Workload Scheduler access method for SAP uses the official SAP RFC interfaces for job scheduling. These are:

- The BC-XBP 6.10 (V2.0) Interface function modules for SAP versions 6.10 and later.
- The BC-XBP 7.00 (V3.0) Interface function modules for SAP versions 7.00, Support Package 16, in addition to the BC-XBP 6.10 (V2.0) function modules.

To avoid conflicts with other vendors, the IBM Workload Scheduler ABAP modules now belong to the IBM Workload Scheduler partner namespace J_101_xxx and /IBMTWS. After you have completed the imports as described in [Importing ABAP/4 function modules into SAP on page 221](#), the RFC J_101_xxx function modules and the /IBMTWS function modules are installed on your system.

If you had a previous installation of IBM Workload Scheduler extended agent for SAP on your system, you can delete the following function modules from your SAP system:

```
Z_MAE2_BDC_STATUS
Z_MAE2_DATE_TIME
Z_MAE2_JOB_COPY
Z_MAE2_JOB_DELETE
Z_MAE2_JOB_FIND
Z_MAE2_JOB_FINDALL
Z_MAE2_JOB_LOG
Z_MAE2_JOB_OPEN
Z_MAE2_JOB_START
Z_MAE2_JOB_STATUS
Z_MAE2_JOB_STOP
```

These are old versions of the ABAP functions, which belong to the customer name space. You can also delete the function group YMA3. It is not necessary to delete the function modules and the function group, but delete them if you want to clean up your system.

To upgrade SAP, perform the following steps:

1. Uninstall the IBM® Workload Scheduler ABAP module.
2. Upgrade SAP.
3. Install the IBM® Workload Scheduler ABAP module.

Configuring the SAP access method

This section provides detailed information about the SAP options file creation.

The files for the SAP access method are located in the following path:

On UNIX operating systems

```
TWA_DATA_DIR/methods
```

On Windows operating systems

```
TWA_home\methods
```

If `r3batch` finds the local configuration file for an extended agent or dynamic agent, it ignores the duplicate information contained in `r3batch.opts`. If instead it does not find a local configuration file then it will use `r3batch.opts` global options file.

To successfully use the SAP access method, you must first install the SAP RFC libraries, as described in the System Requirements Document in the SAP Access Method Requirements section.

Dynamic agents

r3batch.opts

A common configuration file for the `r3batch` access method, whose settings affect all the `r3batch` instances. It functions as a “global” configuration file.

DYNAMIC_AGENT_FILE_r3batch.opts

One or more configuration files that are specific to each dynamic agent workstation within a particular installation of a `r3batch` access method. The `DYNAMIC_AGENT_FILE_r3batch.opts` is the name of the options file, where `DYNAMIC_AGENT` is not necessarily the name of the dynamic agent workstation, because the dynamic agent can have more than one `.opts` file associated. If you do not create a local options file, the global options file is used. Every dynamic agent workstation must have one or more local options file with its own configuration options.



Note: The value for `DYNAMIC_AGENT` must be written in uppercase alphanumeric characters. Double-byte character set (DBCS), Single-byte character set (SBCS), and Bidirectional text are not supported.



Note: If you have a **pool** or **dynamic pool** containing n agents, you must create an options file for the dynamic pool and copy it in the `TWA_home/methods` of each agent of the pool so that all members of the pool have a local options file with the same name. Then you must create another options file for the specific agent in the same directory.

For example, if the SAP access method is installed for `AGENT1` and `AGENT2` that belong to the dynamic pool `DYN_POOL`, you need to create the following options files in the path `TWA_home/methods` on each agent:

AGENT 1

- `FILE_AGENT1_r3batch.opts`
- `FILE_DYN_POOL_r3batch.opts`

AGENT2

- `FILE_AGENT2_r3batch.opts`
- `FILE_DYN_POOL_r3batch.opts`

On dynamic workstations, you can create a new options file or edit an existing options files from a graphical user interface panel available on the General tab of the SAP job definition panels in the Dynamic Workload Console.

Extended agents

r3batch.opts

A common configuration file for the `r3batch` access method, whose settings affect all the `r3batch` instances. It functions as a “global” configuration file.

XA_Unique_ID_r3batch.opts

A configuration file that is specific to each IBM Workload Scheduler extended agent workstation that uses the `r3batch` access method. Its options affect only the `r3batch` instance that is used by that particular workstation. It functions as a “local” configuration file.



Note: `XA_Unique_ID` is the unique identifier for the extended agent workstation. See [UNIQUE_ID on page 177](#) for more details about identifying the unique identifier for an extended agent workstation.

For example, to define two extended agent workstations with unique identifiers, `07756YBX76Z6AFX2` and `S4HANAR3BW`, that access two SAP systems, `SAP1` and `SAP2`, with the `r3batch` access method, you must define the following three configuration files:

- Global `r3batch.opts`
- Local file `07756YBX76Z6AFX2_r3batch.opts`
- Local file `S4HANAR3BW_r3batch.opts`

Defining the configuration options

This section describes the options you can configure in `r3batch.opts` and in `XANAME_r3batch.opts`.

Defining the global options

[Table 22: r3batch global configuration options on page 231](#) lists the options that can be specified only in the *global* configuration file `r3batch.opts`.

Table 22. r3batch global configuration options

Option	Description	Default
dep_sem_proj	(Optional) The project ID for the external dependency semaphore used for handling SAP background processing events as external follows dependencies.	d
icp_sem_proj	(Optional) The project ID for the job interception semaphore.	c
job_sem_proj	(Optional) The project ID for the job semaphore.	a
max_jobs_to_release_for_user	Defines the maximum number of jobs released for each user each time the release job is submitted. If this option is less than or equal to 0, the option is ignored and all jobs are released when the release job is submitted. This	ON

Table 22. r3batch global configuration options (continued)

Option	Description	Default
	option is present both in global and local options. If you define it in both configuration files, the local value overrides the global one.	
primm_enable	(Optional) Enables (ON) the SAP print parameter PRIMM (Print Immediately) for all jobs.	OFF
prnew_enable	(Optional) Enables (ON) the SAP print parameter PRNEW (New Spool Request) for all jobs.	OFF
prrel_enable	(Optional) Enables (ON) the SAP print parameter PRREL (Immediately delete the spool output after printing) for all jobs.	OFF
prsap_enable	(Optional) Enables (ON) the SAP print parameter PRSAP (Print SAP Cover Page) for all jobs. The default value is OFF.	OFF
prunx_enable	(Optional) Enables (ON) the SAP print parameter PRUNX (Print Operating System Cover Page) for all jobs.	OFF
release_all_intercepted_jobs_for_request	Releases jobs for each user on a cyclic basis, based on the number of jobs specified in the max_jobs_to_release_for_user option. The default value is ON, which means that all jobs are submitted for each user: • If the max_jobs_to_release_for_user option is less than or equal to 0, all jobs are released for each user. • If the max_jobs_to_release_for_user option is higher than 0, the specified number of jobs is submitted for each user on a cyclic basis. For example, if max_jobs_to_release_for_user=5, the first 5 jobs are submitted for each user, then the following 5 jobs for each user, and so on, until all jobs for all users are submitted. If this option is set to OFF, it releases for each user only the number of jobs specified in the max_jobs_to_release_for_user option. The remaining jobs are submitted only when a new release job is submitted: • If the max_jobs_to_release_for_user option is less than or equal to 0, all jobs are released for each user. • If the max_jobs_to_release_for_user option is higher than 0, only the specified number of jobs is submitted, then no other job is submitted until the new release job. If max_jobs_to_release_for_user=5, the first 5 jobs are submitted for each user, then no other job is submitted until the new release job.	00

Table 22. r3batch global configuration options (continued)

Option	Description	Default
	This option is present both in global and local options. If you define it in both configuration files, the local value overrides the global one.	
var_sem_proj	(Optional) The project ID for the variant semaphore.	b



Note: If the **max_jobs_to_release_for_user** option is higher than 0, only the specified number of jobs is submitted, then no other job is submitted until the new release job. If **max_jobs_to_release_for_user**=5, the first 5 jobs are submitted for each user, then no other job is submitted until the new release job.

Modifying the default values of the semaphore options is particularly useful when the IDs that are generated would be the same as the IDs already used by other applications.

On UNIX and Linux, to resolve the problem of duplicated IDs, IBM Workload Scheduler for SAP uses system-5 semaphores to synchronize critical ABAP function module calls. It uses one semaphore for job-related tasks and another one for tasks related to variant maintenance.

To synchronize on the same semaphore, the communication partners must use the same identifier. There are several ways to choose this identifier. IBM Workload Scheduler for SAP uses two parameters: a path name and a project ID (which is a character value). The path name parameter is the fully qualified path to the options file. The project ID is taken from the options described in [Table 22: r3batch global configuration options on page 231](#). If these options are omitted, IBM Workload Scheduler for SAP uses default values, which work for most installations.



Note:

1. The semaphore options must be edited directly in the global options file using a text editor; you cannot use the options editor to modify these values.
2. If two semaphore options are assigned the same value, all the semaphore values are reset according to the following rule:

job_sem_proj

It keeps the value assigned, or its default value.

var_sem_proj

It is reset to the first character that, in the ASCII table, follows the value assigned to `var_sem_proj`.

**icp_sem_proj**

It is reset to the second character that, in the ASCII table, follows the value assigned to `var_sem_proj`.

dep_sem_proj

It is reset to the third character that, in the ASCII table, follows the value assigned to `var_sem_proj`.

Defining the local options

Table 23: [r3batch local configuration options on page 234](#) lists the options that you can specify only in the *local* configuration files.

Table 23. r3batch local configuration options

Option	Description
bapi_sync_level	(Optional) Specifies the synchronization level between the SAP function modules BAPI_XBP_JOB_COPY and BAPI_XBP_JOB_START_ASAP. Allowed values are: high All RFC calls between BAPI_XBP_JOB_START_ASAP and BAPI_XBP_JOB_COPY are synchronized. This is the default. medium The RFC calls to BAPI_XBP_JOB_START_ASAP are synchronized. low The RFC calls are not synchronized.
blank_libpath	(Optional) Clears (ON) the operating system variables LD_LIBRARY_PATH and LIBPATH. The default value is OFF.
fn_cache_enabled	(Optional) Enables or disables the file cache on the agent. Can be ON (default value).
fn_cache_purge_interval	(Optional) Specifies the time of validity (in days) of the cached files. If it is left unspecified or set equal to or less than 0, the files are valid indefinitely.
get_job_status_retry	(Optional) Sets the number of times a Remote Function Call must be attempted to retrieve the actual status of an SAP Job. Allowed values are in the range from 1 to 9999. The default value is 5.
get_job_status_retry_delay	(Optional) Sets the number of seconds between two consecutive calls of a Remote Function Call. Allowed values are in the range from 1 to 9999.

Table 23. r3batch local configuration options (continued)

Option	Description
job_duration	(Optional) Enables (ON) that the CPU time value in the production plan report that is run from the Dynamic Workload Console is set to the actual duration of the SAP job. Default value is OFF. To retrieve the job duration from the SAP system, ensure that the authorization profile contains the following authorization objects: • S_DEVELOP • S_TCODE with parameter SE38 (only for SAP 6.40 and 7.00) For details about the authorization profile, see Creating the authorization profile for the IBM Workload Scheduler user on page 217.
max_jobs_to_release_for_user	Defines the maximum number of jobs released for each user each time the release job is submitted. If this option is less than or equal to 0, the option is ignored and all jobs are released when the release job is submitted. This option is present both in global and local options. If you define it in both configuration files, the local value overrides the global one.
primm_enable	(Optional) Enables (ON) the SAP print parameter PRIMM (Print Immediately) for all jobs. The default value is OFF.
prnew_enable	(Optional) Enables (ON) the SAP print parameter PRNEW (New Spool Request) for all jobs. The default value is OFF.
prrel_enable	(Optional) Enables (ON) the SAP print parameter PRREL (Print Release) for all jobs. The default value is OFF.
prsap_enable	(Optional) Enables (ON) the SAP print parameter PRSAP (Print SAP Cover Page) for all jobs. The default value is OFF.
prunx_enable	(Optional) Enables (ON) the SAP print parameter PRUNX (Print Operating System Cover Page) for all jobs. The default value is OFF.
release_all_intercepted_jobs_for_request	Releases jobs for each user on a cyclic basis, based on the number of jobs specified in the max_jobs_to_release_for_user option. The default value is ON, which means that all jobs are submitted for each user: • If the max_jobs_to_release_for_user option is less than or equal to 0, all jobs are released for each user. • If the max_jobs_to_release_for_user option is higher than 0, the specified number of jobs is submitted for each user on a cyclic basis. For example,

Table 23. r3batch local configuration options (continued)

Option	Description
	if max_jobs_to_release_for_user=5, the first 5 jobs are submitted for each user, then the following 5 jobs for each user, and so on, until all jobs for all users are submitted. If this option is set to OFF, it releases for each user only the number of jobs specified in the max_jobs_to_release_for_user option. The remaining jobs are submitted only when a new release job is submitted: • If the max_jobs_to_release_for_user option is less than or equal to 0, all jobs are released for each user. • If the max_jobs_to_release_for_user option is higher than 0, only the specified number of jobs is submitted, then no other job is submitted until the new release job. If max_jobs_to_release_for_user=5, the first 5 jobs are submitted for each user, then no other job is submitted until the new release job. This option is present both in global and local options. If you define it in both configuration files, the local value overrides the global one.
r3client	(Mandatory) The SAP client number.
r3gateway	(Optional) The host name of the SAP gateway.
r3group	(Optional) The name of the SAP logon group.
r3gwservice	(Optional) The service number of the SAP gateway.
r3host	(Mandatory) The host name of the SAP message server when using logon groups, or the host name of the application server in all other cases. If this server can be reached through one or more SAP gateways, use a string in the format <code>/H/gateway/H/</code> for each of them.
r3instance	(Mandatory) The SAP instance number. If r3group is set, this option is ignored.
r3password	(Mandatory) The password for the r3user. Ensure that you enter the same password when creating this user in the SAP system. It can be a maximum of eight characters and is stored in encrypted format. The value is case sensitive. For information about how to encrypt the password see Encrypting SAP user passwords on page 249.
r3sid	(Mandatory) The SAP system ID.

Table 23. r3batch local configuration options (continued)

Option	Description
r3snclib	(Optional) Specifies the path and file name of the SNC library. This option becomes mandatory if r3sncmode is activated (1).
r3sncmode	(Optional) Enables (1), or disables (0), secure network communication (SNC) between r3batch and the SAP R3 system. The default setting is (0). Refer to the SAP documentation for more information about using the SAP cryptographic Library for SNC.
r3sncmyname	(Optional) Specifies the name of the user sending the RFC for secure network communication (SNC).
r3sncpartnername	(Optional) Specifies the SNC name of the SAP R3 communication partner (application server). This option becomes mandatory if r3sncmode is activated (1).
r3sncqop	(Optional) Specifies the secure network communication (SNC) protection level.
r3user	(Mandatory) The name of the SAP user with which the access method connects to the SAP system. It must have the appropriate privileges for running background jobs. It is sometimes also called the Maestro™ User ID.
report_list_max_limit	(Optional) Sets the maximum number of ABAP reports which can be loaded. The default value is -1, which means no limit.

Defining the common options

Table 24: r3batch common configuration options on page 237 lists additional options that you can specify in either configuration file.

Table 24. r3batch common configuration options

Option	Description	Default
bdc_job_status_failed	(Optional) How IBM Workload Scheduler sets the completion status of a job running BDC sessions, according to a possible BDC processing failure. The allowed values are: <i>n</i> If at least <i>n</i> BDC sessions failed (where <i>n</i> is an integer greater than 0), IBM Workload Scheduler sets the job completion status as failed. all If all the BDC sessions failed, IBM Workload Scheduler sets the job completion status as failed.	ignore

Table 24. r3batch common configuration options (continued)

Option	Description	Default
	ignore When all the BDC sessions complete, regardless of their status, IBM Workload Scheduler sets the job completion status as successful. This is the default.  Note: This option is ignored if you defined the job by setting the nobdc or nobdcwait option. For details about these options, see Task string to define SAP jobs on page 262.	
ccms_alert_history	(Optional) Enables (ON) or disables (OFF) the product to retrieve all the matching CCMS alerts, included those that were generated before the monitoring process started. The default value is OFF, meaning that the product retrieves only the CCMS alerts that are generated after the monitoring process started.  Note: This option takes effect the first time you start the CCMS alert monitoring. If you initially set it to OFF and later you want to retrieve the alerts generated before the monitoring process started, stop the monitoring and delete the <code>XAname_r3xalmon.cfg</code> file located in <code>TWA_DATA_DIR/methods/r3evmon_cfg</code> on UNIX® and <code>TWA_home\methods\r3evmon_cfg</code> on Windows™. In the options file, set <code>ccms_alert_history=on</code> and start the monitoring process again.	OFF
commit_dependency	(Optional) Enables (ON) or disables (OFF) the product to commit internetwork dependencies after processing. If you enable this option, internetwork dependencies are committed immediately by default. If you disable or delete this option, the -commit parameter set in the internetwork dependency definition is applied. For details about the -commit parameter, see Table 34: Parameters to define an SAP internetwork dependency on page 343.	OFF
enable_appl_rc	(Optional) Enables (ON) or disables (OFF) the mapping of the application return code to the IBM Workload Scheduler return code.  Note: This feature does not modify the exit code of the access method. For more details, refer to the rccondsucc keyword	OFF

Table 24. r3batch common configuration options (continued)

Option	Description	Default
	 in the job definition documented in <i>IBM Workload Scheduler: User's Guide and Reference</i> .	
evmon_interval	(Optional) The polling rate (in seconds) that the <code>r3evmon</code> process applies to monitor the list of events.	60
ifuser	(Optional) The ID of the user who runs the access method to retrieve job information.	None
idoc_no_history	(Optional) Enables (ON) or disables (OFF) the product to retrieve only IDoc data that is generated after the monitoring process started. If you specify OFF, all matching IDocs are retrieved, including those that were generated before the monitoring process started. When processing this option, <code>r3evmon</code> uses the <code>XAname_r3idocmon.cfg</code> file to retrieve the date and time for the next monitoring loop.	ON
idoc_shallow_result	(Optional) Enables (ON) or disables (OFF) the product to retrieve only the most recent matching IDocs. For example, suppose you set <code>idoc_shallow_result=ON</code> . If the status of an IDoc changes several times during the monitoring interval and the same status, matching an event rule condition, occurs more than once in the sequence of statuses, only the most recent matching IDoc is retrieved. If you specify OFF, all matching IDocs are retrieved.	ON
jobdef	(Optional) If enabled, you can use the Dynamic Workload Console to define jobs, in addition to the command line. Specify <code>r3batch</code> to enable the option, and any other value to disable it.	<code>r3batch</code>
job_interceptable	(Optional) Enables (ON) or disables (OFF) the job launched by <code>r3batch</code> to be intercepted by SAP. If enabled, when <code>r3batch</code> launches a job and the SAP job interception feature is enabled, the job can be intercepted if it matches previously defined criteria. If disabled, the job launched by <code>r3batch</code> cannot be intercepted by SAP.	OFF
ljuser	(Optional) The ID of the user who runs the access method to launch jobs (LJ tasks) and manage jobs (MJ tasks).	None
log_r3syslog	(Optional) Enables (ON) or disables (OFF) the access method to write the latest entries from the SAP syslog to its trace file when an RFC returns with a general error.	OFF

Table 24. r3batch common configuration options (continued)

Option	Description	Default
long_interval	(Optional) The maximum interval, in seconds, between status checks. It cannot be greater than 3600 seconds. See also <i>short_interval</i> .	3600
max_n0_counter	(Optional) The maximum value of the N0 counter. If the N0 counter reaches the specified value, it starts again from 0.	$2^{15} - 1$
max_name_counter	(Optional) The maximum value of the variant name counter. If the name counter reaches the specified value, it starts again from 0.	40
n0_counter_policy	(Optional) The N0 counter policy: step The N0 counter is increased once for every step. job The N0 counter is increased once for every job.	job
name_counter_policy	(Optional) The name counter policy: step The name counter is increased once for every step. job The name counter is increased once for every job.	job
nojobdefs	(Optional) Disables (1) or enables (0) the definition of new SAP jobs using the Dynamic Workload Console. If this option is set to 1, you must create the job definitions in the SAP job before creating the IBM Workload Scheduler job that is going to schedule them.	0
oldcopy	(Optional) Enables (1) or disables (0) the access method to use the old way of copying jobs, even though the function module BAPI_XBP_JOB_COPY is present on the SAP system.	0
pchain_recover	(Optional) The action taken by IBM Workload Scheduler when you rerun a job that submits a process chain. The allowed values are: rerun IBM Workload Scheduler creates another process chain instance and submits it to be run again. restart IBM Workload Scheduler restarts the original process chain from the failing processes to the end.	rerun

Table 24. r3batch common configuration options (continued)

Option	Description	Default
	For details about rerunning a process chain job, refer to Rerunning a process chain job on page 326 .	
pchain_details	(Optional) Enables (ON) or disables (OFF) the display of details about an SAP process chain that you scheduled as an IBM Workload Scheduler job.	OFF
pchainlog_bapi_msg	(Optional) Enables (ON) or disables (OFF) the product to retrieve additional messages from the BAPI calls from the SAP Business Warehouse process chains and appends them to the stdlist of IBM Workload Scheduler.	ON
pchainlog_level	(Optional) Supplements the option retrieve_pchainlog. Specifies which level of process chain logs you want to retrieve. Allowed values are: 1 Only the first level of process chain is logged. level_number Process chains are logged down to the level of chain you indicate here. For example, if you indicate 2 only the first two levels are logged. all All process chains are logged.	If you omit this option, and leave retrieve_pchainlog set to ON, the default is level 1.
pchainlog_verbosity	(Optional) Supplements the option retrieve_pchainlog. Specifies which type of process chain logs you want to retrieve. Allowed values are: chains_only Logs only the process chains. chains_and_failed_proc In addition to the process chains, logs all failed processes. complete Logs all process chains and processes.	If you omit this option, and leave retrieve_pchainlog set to ON, the default is complete.

Table 24. r3batch common configuration options (continued)

Option	Description	Default
	 Note: This option affects the entire process chain; verbosity cannot be reduced for individual processes.	
pc_launch_child	(Optional) Enables (ON) or disables (OFF) the product to launch child jobs that are in scheduled state.  Note: You can use this option only if you activated the parent-child feature on the SAP system. On the XBP 2.0 or later SAP system, you can activate this feature by using the INITXBP2 ABAP report.	OFF
placeholder_abap_step	(Optional) If XBP version 2.0 is used, the name of the ABAP report used as the dummy step in the SAP placeholder job that is created to monitor an SAP event defined as external dependency.	If this option is not specified, either as global or local option, the default BTCTEST is used.
qos_disable	(Optional) Enables (ON) or disables (OFF) the creation of the environment variable QOS_DISABLE on Microsoft™ Windows™ systems that use the Quality of Service (QoS) feature, before r3batch opens an RFC connection. Without this option, because of problems in the implementation of the QoS service, the connection between r3batch and the SAP RFC library does not work.	OFF
r3auditlevel	(Optional) The audit level for the XBP. A number from 0 (low) to 3 (high).	3
rcmap	(Optional) Enables (ON) or disables (OFF) the return code mapping capabilities of Access method for SAP.	ON
retrieve_applinfo	(Optional) Enables (ON) or disables (OFF) the retrieval and appending of the SAP application log to the stdlist of IBM Workload Scheduler.	OFF
retrieve_ipaklog	(Optional) Enables (ON) or disables (OFF) the retrieval and appending of the SAP BW InfoPackage logs to the stdlist of IBM Workload Scheduler.	ON

Table 24. r3batch common configuration options (continued)

Option	Description	Default
	 Note: The retrieval and appending of SAP BW InfoPackage job logs to the <code>stdlist</code> might be time-consuming for jobs that produce large logs.	
retrieve_joblog	(Optional) Enables (ON) or disables (OFF) the retrieval and appending of the SAP job logs to the <code>stdlist</code> of IBM Workload Scheduler.  Note: 1. The retrieval and appending of job logs to the <code>stdlist</code> might be time-consuming for jobs that produce large logs. 2. If you disable the retrieval of the job logs, you also disable the return code mapping function for the log entries. 3. This option does not affect the <code>BDC wait</code> feature.	ON
retrieve_pchainlog	(Optional) Enables (ON) or disables (OFF) the retrieval and appending of the SAP BW process chain logs to the <code>stdlist</code> of IBM Workload Scheduler.  Note: 1. The retrieval and appending of SAP BW process chain logs to the <code>stdlist</code> might be time-consuming for jobs that produce large logs. 2. If you disable the retrieval of the SAP BW process chain logs, you also disable the return code mapping function for the log entries. 3. This option on its own retrieves the log of only the first level of a process chain. To retrieve more complete logs, use this option with the pchainlog_level and pchainlog_verbosity options.	ON
retrieve_spoollist	(Optional) Enables (ON) or disables (OFF) the retrieval and appending of the SAP job spool lists to the <code>stdlist</code> of IBM Workload Scheduler.  Note:	ON

Table 24. r3batch common configuration options (continued)

Option	Description	Default
	 1. The retrieval and appending of SAP job spool lists to the <code>stdlist</code> might be time-consuming for jobs that produce large spool lists. 2. If you disable the retrieval of the SAP job spool lists, you also disable the return code mapping function for the spool list entries.	
retry	(Optional) The retry count for SAP function module calls. Specify an integer greater than 0.	5
rfc_interval	(Optional) The polling rate (in milliseconds) with which r3batch listens for results of RFC requests. The rate cannot exceed 1,000 milliseconds. Consider that the lower the value of the <code>rfc_interval</code> option, the higher the frequency with which RFC request results are collected and, as a consequence, CPU consumption on the r3batch system is high.	10
rfc_open_delay	(Optional) The maximum number of seconds to wait between two consecutive calls before opening an RFC connection.	1800
rfc_open_retry	(Optional) The retry count for opening an RFC connection to the SAP system. Specify an integer greater than 0 to limit the number of retries, or -1 for an unlimited number of retries.	5
rfc_timeout	(Optional) The time (in seconds) that r3batch waits before canceling a non-responding RFC communication. Allowed values are in the range from 0 to 9999; 0 means no timeout.	600
short_interval	(Optional) The minimum interval, in seconds, between status checks. It cannot be less than 2 seconds. Setting this option to low values makes the notification of status changes faster, but increases the load on the hosting machine. See also <i>long_interval</i> .	10
throttling_enable_ job_class_inheritance	(Optional) Enables (ON) or disables (OFF) the inheritance of priority class. ON means that the intercepted job inherits the priority class of its progenitor job, if it is higher than its own class; otherwise it keeps its own class. OFF means that the intercepted job keeps its own class, regardless of its progenitor's class.	ON

Table 24. r3batch common configuration options (continued)

Option	Description	Default
	 Note: By setting this option, the parent-child feature is automatically enabled on the SAP system.	
throttling_enable_job_interception	(Optional) Enables (ON) the job interception feature at job throttler startup, or keeps the current setting (OFF). ON means that when the job throttler starts, it enables the job interception feature on the SAP system. When the job throttler is stopped, the job interception feature is also automatically restored to the setting that was previously configured on the SAP system. OFF means that the job interception feature is kept as it is currently set in the SAP system.	ON
throttling_job_interception_version	Specifies the BC-XBP interface version to be used when the job throttler starts. Valid values are: • 2 • 3 The default BC-XBP interface version that is used is 2 (version 2.0).	2
throttling_interval	(Optional) The interval (in seconds) between each job throttling run.	5
throttling_max_connections	(Optional) The maximum number of connections (connection pool size) that the job throttler can open to communicate with the SAP system. The minimum value is 3.	5
throttling_release_all_on_exit	(Optional) Enables (ON) or disables (OFF) the release of all intercepted jobs. ON means that when the job throttler is stopped, it releases all the intercepted jobs. OFF means that when the job throttler is stopped, it does not release the intercepted jobs therefore the jobs remain intercepted, in scheduled state.	ON
throttling_send_ccms_data	(Optional) Enables (ON) or disables (OFF) the sending of data from job throttling to the SAP CCMS Monitoring Architecture. ON means that the job throttler sends its status data to CCMS continuously. OFF means that the job throttler does not send its status to CCMS.	OFF

Table 24. r3batch common configuration options (continued)

Option	Description	Default
throttling_send_ccms_rate	(Optional) Rate (in number of runs) at which the job throttler sends its status data to the SAP CCMS monitoring architecture. The minimum value is 1, meaning that the job throttler sends the data at every run.	1
twsmeth_cp	(Optional) The code page that r3batch uses to write its output. This option must be consistent with twsmeth_lang . It can be any of the existing TIS codepages.	The code page used by the IBM Workload Scheduler workstation that hosts the r3batch access method.
twsmeth_lang	(Optional) The language used to report messages. This option must be consistent with twsmeth_cp .	The language of the locale of the workstation that hosts the r3batch access method.
twsga_cp	(Optional) The encoding used by r3batch to establish RFC connections with SAP systems. Use this option if r3batch is not Unicode-enabled. Possible values are: • 1100 • 1103 • 8000 • 8300 • 8400	1100
twsga_lang	(Optional) The language used to log in to SAP systems. Specify one of the following (DE, EN, and JA can be set from the Option Editor. The other languages can be set using any text editor): DE German EN English ES Spanish FR French	EN

Table 24. r3batch common configuration options (continued)

Option	Description	Default
	IT Italian JA Japanese KO Korean pt_BR Brazilian Portuguese zh_CN Simplified Chinese zh_TW Traditional Chinese  Note: If you are working with InfoPackages and process chains on operating systems that do not support Unicode, this option must be set.	
use_fips	(Optional) Enables (ON) or disables (OFF) the FIPS mode of operation for IBM Workload Scheduler.	OFF
utf8cmdline	(Optional) Enables (1) or disables (0) the encoding of extended parameters in UTF-8 format. The default value is 0.	 Note: If you have both global and local options files and you want to change the default value for utf8cmdline, modify the local options file because this overrides

Table 24. r3batch common configuration options (continued)

Option	Description	Default
		 the global options.
variant_delay	(Optional) The time, in seconds, that r3batch allows the SAP system to clean up the structures used for communication between r3batch and the SAP system. This option is valid when you launch a job that uses extended variants and requires a copy of a job template. Use this option only when you want to reduce r3batch response time, because it increases the load on the hosting machine. Higher values of variant_delay increase the response time and decrease the load. Allowed values are in the range from 0 to 3600.	10
variant_selection_screen	(Optional) Specifies the functional interface used to read report selection screens. Specify one of the following: Custom To communicate with the SAP system using the IBM® Workload Scheduler custom function module. SAP To communicate with the SAP system using the XBP 3.0 function module.	Custom
xbpversion	(Optional) The XBP version used on the target SAP system. Specify an integer value. This value overwrites the XBP version automatically determined during RFC logon.  Note: For details about XBP 3.0 and SAP NetWeaver 2004s with SP9, refer to the SAP Note 977462.	The XBP version determined by r3batch during RFC logon from the SAP system.

SAP option file example

Below is an example of an options file for SAP. It can help you determine your specific site requirements, although your options file might be different.

```
r3client=100
r3host=/H/tiraix64.lab.rome.abc.com
r3instance=00
r3password={aes}hyb/LQNYVzIf9oA8/xgY+CSqAuAh7+CvTT7HuDpdiu5YUAH0KJEHJtA=
r3sid=GS7
r3user=twstest
long_interval=120
r3auditlevel=3
```



```
short_interval=10
twswa_lang=EN
```

Encrypting SAP user passwords

When you add your entries in the options file from the Dynamic Workload Console, the password value is automatically encrypted before it is written in the file. If you modify the file with a text editor, run the **enigma** program to encrypt the password before writing it in the file, as follows:

```
enigma your_password
```

You can include the password on the command line or enter it in response to a prompt. The program returns an encrypted version that you can then enter in the options file.

Configuration options usage

The format of the configuration file is the following:

```
option1=value1
option2=value2
option3=value3
...
```

with no blanks before the option, after the value, or before or after the equals (=) character.

You can put all the common information, such as the `LJuser`, `IFuser`, `JobDef`, and `LogFileName` options in `r3batch.opts`, while you can put tailored data for the target SAP system of the extended agent or dynamic agent (for example, `SAP1`) in a local configuration file (for example, `XA1_r3batch.opts`).

You can put a local option in the global configuration file if you want to give the same option to all the `r3batch` instances. For example, if the SAP user name is the same in all your SAP systems, you can place the `r3user` option in the global file without duplicating that information in all the local configuration files.

A global option, such as `job_sem_proj`, only has effect in the global configuration file. If you put global options in a local file they have no effect.

`r3batch` reads the global configuration file first and then the local file. Every option (except the global options) contained in the local configuration file will override those in the global file. For example, if both the global and the local configuration files contain the `r3user` option, `r3batch` uses the one in the local file.

There are six mandatory options that `r3batch` requires:

- `r3client`
- `r3host`
- `r3instance`
- `r3password`
- `r3sid`
- `r3user`

You can put them all in the local configuration file or you can spread them between the global and the local files. For example, you could put `r3user` and `r3password` in the global configuration file and `r3sid`, `r3instance`, `r3client`, and `r3host` in the local one.

The `r3user` option is both local and mandatory. It must be put either in the global configuration file or the local configuration file.



Note: These configuration files are not created during the installation process.

Connecting to the SAP system

The Access method for SAP uses the SAP remote connection call (RFC) library to connect to the SAP system. The connection address for an SAP system is denoted as a `connection string`.

To successfully use the SAP R/3 access method, you must first install the SAP RFC libraries, as described in the System Requirements Document in the SAP R/3 Access Method Requirements section.

Connecting to a specific application server

To connect to a specific application server, you enter strings which, according to the complexity of the networks, might be more or less complex and contain passwords to secure the routers.

In its basic form, a connection string consists of the host name (or IP name) of an SAP application server; for example:

```
/H/hemlock.romlab.rome.abc.com
```

This type of connection string works only in very simple network environments, where all application servers can be reached directly through TCP/IP. Usually, modern companies use more complex network topologies, with a number of small subnetworks, which cannot communicate directly through TCP/IP. To support this type of network, the SAP RFC library supports SAP routers, which are placed at the boundaries of the subnetworks and act as proxies. For this type of network, the connection string is a composite of basic connection strings for each SAP router, followed by the basic connection string for the target SAP system; for example:

```
/H/litespeed/H/amsaix33/H/hemlock.romlab.rome.abc.com
```

Moreover, you can secure the SAP routers with passwords, to prevent unauthorized access. In this case, the basic connection string for the SAP router is followed by `/P/` and the password of the router.



Note: The SAP RFC library limits the length of the connection string to a maximum of 128 characters. This is a real limitation in complex network environments. As a workaround, it is recommended to use simple host names, without the domain name whenever possible. Alternatively, you can use the IP address, but this is not recommended, because it is difficult to maintain.

IBM Workload Scheduler for SAP supports both types of connection strings, basic and composite, where:

r3host

The connection string.

r3instance

The SAP instance number.

r3sid

The SAP system ID.

For example:

```
r3host=/H/litespeed/H/amsaix33/H/hemlock.romlab.rome.abc.com
r3instance=00
r3sid=TV1
```

Connecting to a logon group

About this task

In large SAP installations, the application servers are usually configured in logon groups for load balancing and fault-tolerance purposes. Load balancing is done by a dedicated server, called the `message server`. The message server automatically assigns users to the application server with the least workload of the logon group it controls.

Ensure that the file services (on UNIX: `/etc/services` on Windows: `C:\Windows\system32\drivers\etc\services`) contain an entry for the message server port of the SAP system to which `r3batch` connects. The entry has the following format:

```
sapmsSID 36system_number/tcp
```

where `SID` is the SAP system ID, and `system_number` is the SAP system number.

Set the following options to configure `r3batch` to connect to a logon group:

r3host

The hostname of the message server.

r3group

The name of the logon group.

r3sid

The SAP system ID.

For example:

```
r3host=pwdf0647.wdf.sap-ag.de
r3group=PUBLIC
r3sid=QB6
```

Configuring SAP event monitoring

This section provides detailed information about how to configure your system to monitor SAP events:

- [Prerequisite to defining event rules based on SAP events on page 252](#)
- [Monitoring SAP events on page 252](#)

Prerequisite to defining event rules based on SAP events

About this task

To be able to define event rules based on one or more SAP events, stop the IBM Workload Scheduler WebSphere Application Server Liberty Base and copy the following file (located on the system where you installed IBM Workload Scheduler:

On UNIX operating systems

```
TWA_DATA_DIR/methods/SAPPlugin/SapMonitorPlugIn.jar
```

On Windows operating systems

```
TWA_home\methods\SAPPlugin\SapMonitorPlugIn.jar
```

to the following directory of the master domain manager and of its backup nodes:

On UNIX operating systems

```
TWA_DATA_DIR/eventPlugIn
```

On Windows operating systems

```
TWA_home\eventPlugIn
```

For the changes to take effect, stop and restart the IBM Workload Scheduler WebSphere Application Server Liberty Base. If the master domain manager is connected to the Dynamic Workload Console, stop and restart also the Dynamic Workload Console Application Server.

Monitoring SAP events

Whenever you define an event rule based on an SAP event in your IBM Workload Scheduler plan, that event is monitored by IBM Workload Scheduler. Monitoring SAP events is allowed only if you use XPB version 3.0, or later.

IBM Workload Scheduler monitors two types of SAP event:

Events defined by the SAP system

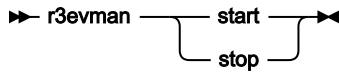
The events that are triggered automatically by system changes, for example when a new operation mode is activated. This type of event cannot be modified by the user.

Events defined by the user

The events that are triggered by ABAP or external processes, for example when a process triggers an SAP event to signal that external data has arrived and must be read by the SAP system. For details about how to trigger events by external processes, refer to [Raising an SAP event on page 278](#).

If you modify the `r3batch` option files, to make the changes effective you must stop and restart the extended agent monitoring processes with the following command. For UNIX only, this command must be entered by the owner of the IBM Workload Scheduler installation:

Figure 15. Command syntax



Where:

start | stop

The action to perform:

start

Starts monitoring SAP events.

stop

Stops monitoring SAP events.

Defining SAP jobs

You must define some jobs to be able to run jobs on an SAP workstation from IBM Workload Scheduler.

To define and manage jobs on an SAP workstation from IBM Workload Scheduler, you must define the following:

Jobs in SAP that you want to run under IBM Workload Scheduler control

You can define these jobs using standard SAP tools or using the Dynamic Workload Console.

Jobs in IBM Workload Scheduler that correspond to the jobs in SAP

The IBM Workload Scheduler job definitions are used in scheduling and defining dependencies, but the SAP jobs are actually run.

You can define SAP job definitions from the Dynamic Workload Console and then have IBM Workload Scheduler launch the jobs in SAP R/3 using jobs defined on the following workstations that support the r3batch access method:

- An IBM Workload Scheduler extended agent workstation. A workstation that is hosted by a fault-tolerant agent or master workstation.
- A dynamic agent workstation.
- A dynamic pool.
- A z-centric workstation.

You can manage your SAP environment from both:

- An IBM Workload Scheduler distributed environment
- An IBM Z Workload Scheduler environment.

The SAP job definitions can reference the following types of SAP jobs:

- Standard R/3
- Business Warehouse Process Chains
- Business Warehouse InfoPackages

For information about Business Warehouse Process Chains and Business Warehouse InfoPackages, see [Using Business Information Warehouse on page 316](#).

Creating SAP Standard R/3 jobs from the Dynamic Workload Console

How to create and manage an SAP job that is associated to an IBM Workload Scheduler job that manages it.

About this task

You can easily create and manage Standard R/3 jobs on a remote SAP system entirely from the Dynamic Workload Console, and then continue to manage the remote SAP job from IBM Workload Scheduler.

The IBM Workload Scheduler job definition, available for both distributed and z/OS environments, maps to the newly created job on the SAP system. The SAP job can run on extended agent workstations, dynamic agent workstations, pools, dynamic pools, and workstations depending on the type of job definition you choose to create.



Note: Using this procedure to create a new IBM Z Workload Scheduler Agent SAP Standard R/3 job, you cannot manage variants. To manage variants, use the SAP graphical user interface or use the **List Jobs on SAP** entry from the navigation tree of the Dynamic Workload Console.

To create a new SAP Standard R/3 job on a remote SAP system that maps to an IBM Workload Scheduler job definition, you have to associate your SAP Standard R/3 jobs to IBM Workload Scheduler jobs and you can do as follows:

- Creating an SAP job on a remote SAP system: [Creating an SAP job from the Dynamic Workload Console on page 254](#)
- Creating an IBM Workload Scheduler job and associating it to an SAP job: [Create an IBM Workload Scheduler job and associate it to an SAP job on page 255](#)

When performing operations that require a connection to a remote SAP system, you must configure the SAP connection data. The connection is made through an IBM Workload Scheduler workstation with the r3batch access method installed. Each workstation can have one or more options files that can be used to customize the behavior of the r3batch access method, except for extended agent workstations, where only one options file can be defined and therefore a selection is not required. For information about setting the SAP connection data, see [Setting the SAP data connection on page 256](#).

Creating an SAP job from the Dynamic Workload Console

How to create an SAP job definition on a remote SAP system from the Dynamic Workload Console.

About this task

You can also create and save SAP Standard R/3 jobs directly on the remote SAP system from IBM Workload Scheduler, as you would from the SAP graphical user interface. To create Standard R/3 jobs on the SAP system from the Dynamic Workload Console, perform the following steps:

1. From the **Design** menu, select **Manage Jobs on SAP**.
2. In the Job Filter Criteria, select **Standard Job** and specify the workstation name. This parameter is mandatory because it identifies the remote SAP system.
3. Specify the workstation where the SAP job runs. This is the workstation with the r3batch access method that communicates with the remote SAP system.
4. If the workstation is not an extended agent workstation, you must also specify the options file to be used.
5. Click **Display** to view a list of the Standard R/3 jobs for the specified workstation.
6. Click **New** to create a new Standard R/3 job and enter the required information.
7. Click **Save** to create the job on the SAP system.

What to do next

After creating the new SAP job on the SAP from the Dynamic Workload Console, you must reference it in an IBM Workload Scheduler SAP Standard R/3 job if you want to manage the job from within IBM Workload Scheduler as explained in [Create an IBM Workload Scheduler job and associate it to an SAP job on page 255](#).

Create an IBM Workload Scheduler job and associate it to an SAP job

Create an IBM Workload Scheduler job definition and map it to a new or existing SAP job to manage it.

About this task

To create a new IBM Workload Scheduler job and then associate it to a new SAP job, follow these steps:

1. From the **Design** menu, click Workload Designer page.
2. Select an engine. The Workload Designer window is displayed.
3. Click **Create New**, select **Job definition** and in the **ERP** category choose:

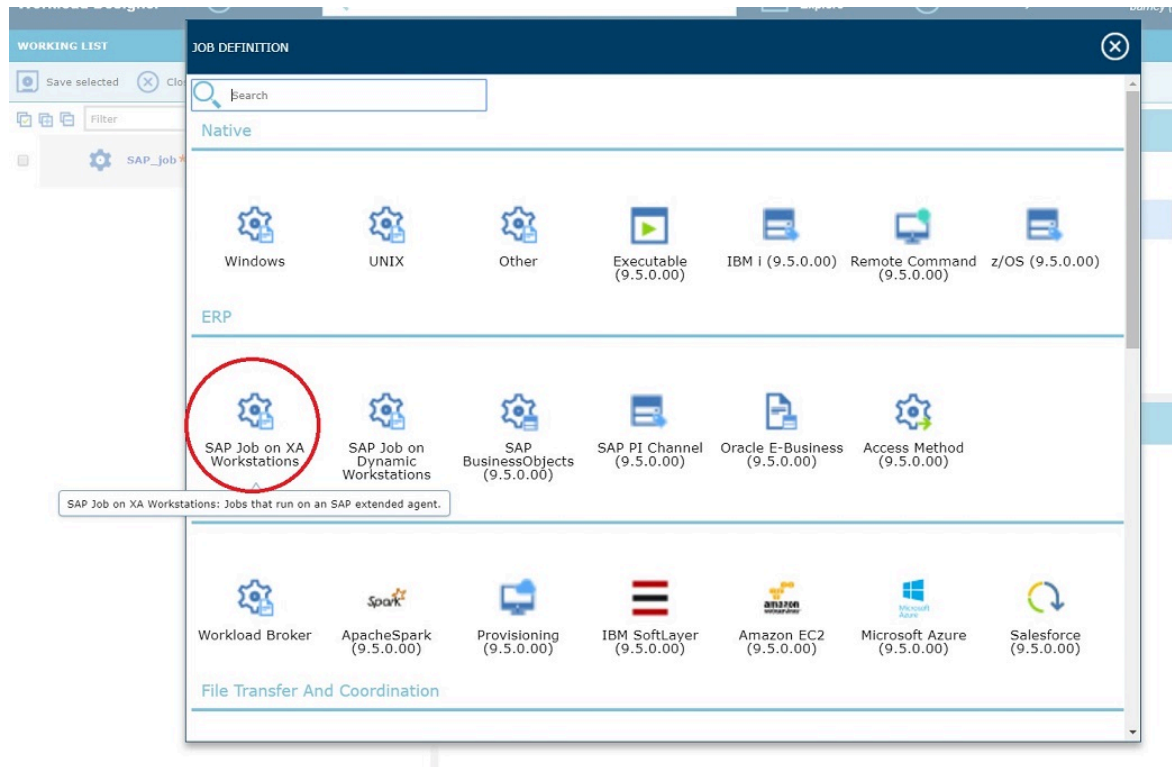
◦

SAP Job on Dynamic Workstations

For distributed systems only. This job definition can run on dynamic agent workstations, dynamic pools, and IBM Z Workload Scheduler Agent workstations.

SAP Job on XA Workstations

This job definition can run on extended agent workstations, which are workstations hosted by fault-tolerant agents or master workstations.



- **SAP**

- SAP**

For z/OS systems only. This job definition references an existing job on the SAP system and can run on dynamic agent workstations, dynamic pools, and IBM Z Workload Scheduler Agent.

4. In the Properties pane, specify the properties for the SAP job definition you are creating using the tabs available. The tabs for each type of SAP job definition are similar, but there are some differences depending on the type of engine you selected and the type of workstation on which the job runs. For more detailed information about the UI elements on each tab, see the Dynamic Workload Console online help.
5. In the **Task** section, specify the IBM Workload Scheduler job that you want to associate to the SAP job.
6. Click **Save** to create the SAP job definition in the IBM Workload Scheduler database.

Setting the SAP data connection

You can configure a default connection to be used when performing actions that access the remote SAP system.

About this task

There are several operations you can perform which require connection details to establish a link to a remote SAP system. The connection is made through an IBM Workload Scheduler workstation with the r3batch access method installed used to communicate with the SAP system. Each workstation can have one or more options files that can be used to customize the behavior of the r3batch access method, except for extended agent workstations, where only one options file can be defined and therefore a selection is not required.

For example, you can use **Workload Designer** to create IBM Workload Scheduler job definitions that reference remote SAP jobs, or you can create a SAP job on a remote SAP system. You can also search for SAP jobs on the remote system from the **Working List** and **Quick Open** panes.

To configure a default SAP data connection to be used when creating items in the Workload Designer that require a SAP connection, perform the following steps:

1. From the **Design** menu, select **Manage SAP Criteria Profile**.
2. Select an engine.
3. In **Workstation**, enter the name of the workstation that communicates with the SAP system. If you do not know the name of the workstation, click the lookup icon and specify a search criteria to find the workstation.
4. In **Options file**, enter the file name of the options file or click the lookup icon to search for options files that reside on the specified workstation and select one.
5. Click **Go** to establish the connection.

Results

A default SAP connection is now configured. It will be used each time an item that requires access to a SAP system is defined.

Managing SAP variants using the Dynamic Workload Console

Managing variants using the Dynamic Workload Console.

About this task

This section describes how to manage variants using the Dynamic Workload Console:

1. Click **Design > SAP > Manage Jobs on SAP** from the portfolio.
2. Specify an engine connection.
3. In **Workstation name**, type the name of the workstation where the SAP job runs. This is the workstation with the r3batch access method that communicates with the remote SAP system. If you do not know the name of the workstation, click (...) browse to enter your filter criteria and click **Search**. If you enter a string representing part of the workstation name, it must be followed by the asterisk (*) wildcard character. Both the question mark (?) and asterisk (*) are supported as wildcards. You can also simply use the asterisk wildcard character (*) to display all workstations. Optionally, specify any of the other search criteria available and click **Search**.

Result

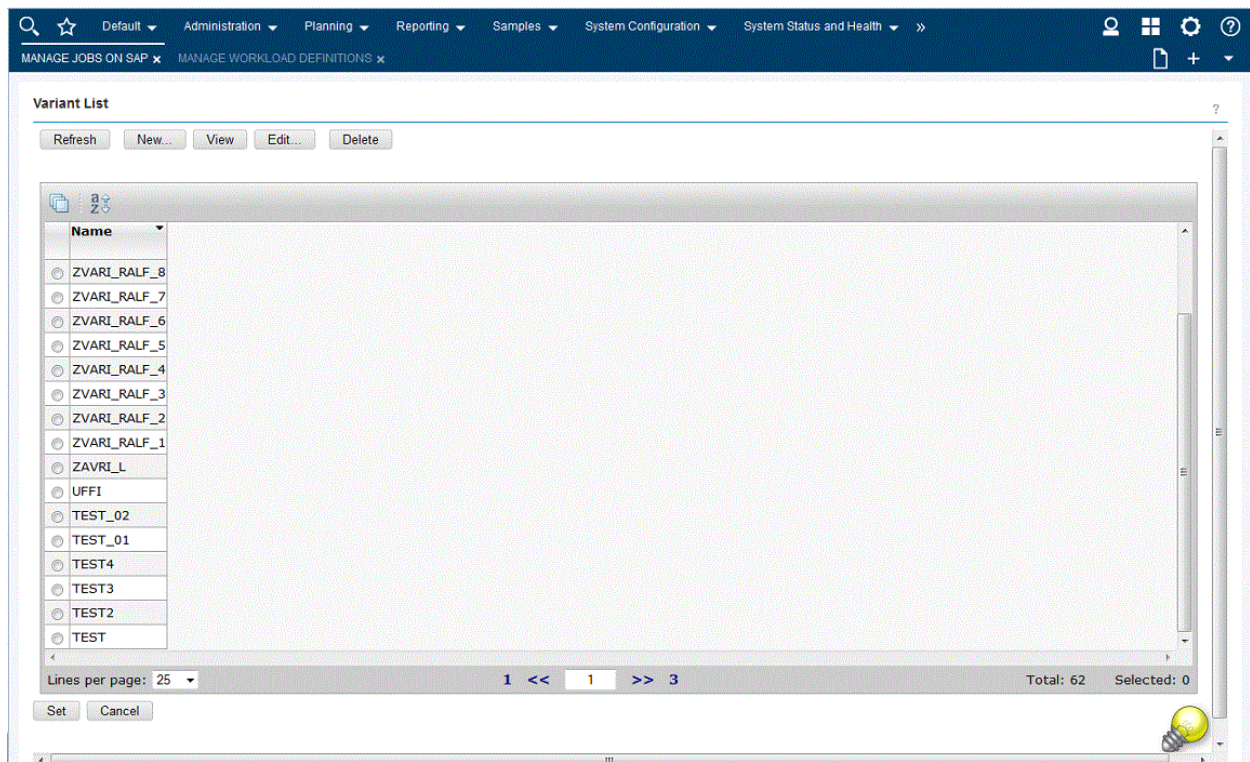
From the results displayed, select the workstation and click **OK**.

4. In **Options file**, specify an options file that resides on the specified workstation. Each workstation can have one or more options files that can be used to customize the behavior of the r3batch access method, except for extended agent workstations, where only one options file can exist and therefore does not need to be specified. For the workstation specified, enter the file name of the options file or click the browse (...) button to search for options files that reside on the specified workstation and select one.
5. Click **Display**. The list of available jobs on the remote SAP system for the specified engine is displayed.
6. A list of SAP jobs on the remote SAP system are displayed.

7. Select a SAP job from the list and click **Edit**.
8. On the **SAP Steps** page, select a program of type ABAP from the list and click **Edit**. The properties for the ABAP program are displayed.
9. In the **Variant** field, click the ellipsis (...) icon to display the Variant List panel. This panel lists all the variants associated with the ABAP specified in the **Name** field.

Result

Figure 16. The Variant List panel



10. From this panel, you can take the following actions:

Refresh

To refresh the content of the variant list with the information contained in the SAP database.

New

To create a new variant as described in [Creating or editing a variant on page 259](#).

View

To display information on an existing variant.

Edit

To modify information on an existing variant as described in [Creating or editing a variant on page 259](#).

Delete

To delete a variant.

Set

To associate the value chosen from the list to the ABAP.

Creating or editing a variant

About this task

You can create or edit a variant from the **Variant List** panel. To display the **Variant List** panel, see [Managing SAP variants using the Dynamic Workload Console on page 257](#).

1. In the Variant List panel, click **New** or **Edit**. The Variant Information page is displayed by default. If you are editing an existing variant, the fields and selections are not empty.

Figure 17. The Variant Information page of the Variant List panel

The screenshot displays the 'Manage Jobs on SAP' interface. On the left, a sidebar contains a 'Variant Values' section. The main area is titled 'Variant Information' and is divided into three sections: 'Variant', 'Properties', and 'Counter'. The 'Variant' section contains input fields for '*Variant Name' (ZVARI_RALF_8), '*Description' (ZVARI_RALF_8), 'ABAP Name' (ZVARI), 'User Name' (TWSTEST4), 'Client' (001), and 'Day' (02.04.2012). The 'Properties' section has four checkboxes: 'Background' (checked), 'Protected' (checked), 'Invisible' (unchecked), and 'Extended' (checked). The 'Counter' section has a 'Counter' button. At the bottom, there are 'OK' and 'Cancel' buttons.

2. The panel consists of the following pages:
 - Variant Information:
 - a. Enter or modify the variant name and description.
 - b. Optionally, check a **Properties** box:

Background

The variant can only be used in background processing.

Protected

The variant is protected against being changed by other users.

Invisible

The variant will not be displayed in the F4 value list on the SAP GUI. Not available for the BC-XBP 3.0 interface.

Extended

Allows for the use of placeholders and counters as variant values. If you check this box, **Counter** becomes available.

For extended variants, you can use placeholders and counters that eliminate the error-prone task of adjusting values and therefore minimize the effort for variant maintenance. Placeholders and counters are preprocessed by IBM Workload Scheduler and the values are automatically adjusted when the job is launched. Supported placeholders and counters are:

Table 25. Placeholders and counters for extended variants

Symbol	Meaning	Syntax
\$S	Timestamp	YYYYMMDDHHMM
\$D	Day of the month	DD
\$_D	Date	YYYYMMDD
\$M	Month	MM
\$Y	Year	YY
\$_Y	Year	YYYY
\$H	Hour	HH
\$T	Minute	MM
\$_T	Time	HHMMSS
\$Nx	Counters	10 counters: \$N0 - \$N9 (\$N = \$N0)
\$(<i>date expression</i>)	Date expression	Like the <code>datecalc</code> command. Enclosed within \$(and).
\$(<i>arithmetic expression</i>)	Arithmetic expression	Arithmetic expressions allowing for +, -, *, and / operations between integers and counters.

◦ Variant Values:

In the Variant Values page, the fields and values are dynamically built through `r3batch` depending on the characteristics of the variant or step and are identical to the ones in the equivalent SAP panel.

Editing a standard SAP job

Before you begin

You can edit SAP Standard R/3 jobs in two different ways in IBM Workload Scheduler.

- The Dynamic Workload Console contains the **Manage Jobs on SAP** entry in the portfolio for creating and editing SAP Standard R/3 jobs on remote SAP systems.
- From the Workload Designer you can create and edit remote SAP jobs. See [Creating SAP Standard R/3 jobs from the Dynamic Workload Console on page 254](#).

About this task

To edit a SAP standard R/3 job, follow these steps:

1. Click **Design > SAP > Manage Jobs on SAP**.
2. Select the name of the engine connection from which you want to work with SAP jobs.
3. Leave the default setting in the **SAP Job Type** section to **Standard R/3 Job**.
4. In **Workstation name**, type the name of the workstation where the SAP job runs. This is the workstation with the r3batch access method that communicates with the remote SAP system. If you do not know the name of the workstation, click (...) browse to enter your filter criteria and click **Search**. If you enter a string representing part of the workstation name, it must be followed by the asterisk (*) wildcard character. Both the question mark (?) and asterisk (*) are supported as wildcards. You can also simply use the asterisk wildcard character (*) to display all workstations. Optionally, specify any of the other search criteria available and click **Search**.

Result

From the results displayed, select the workstation and click **OK**.

5. In **Options file**, specify an options file that resides on the specified workstation. Each workstation can have one or more options files that can be used to customize the behavior of the r3batch access method, except for extended agent workstations, where only one options file can exist and therefore does not need to be specified. For the workstation specified, enter the file name of the options file or click the browse (...) button to search for options files that reside on the specified workstation and select one.
6. Click **Display**. The list of available jobs on the remote SAP system for the specified engine is displayed.
7. Select the job you want to modify in the list and click **Edit**. The **List Jobs on SAP** panel is displayed.
8. Edit the properties on the **R/3 Job Definition** and **R/3 Steps** pages as appropriate. Refer to the contextual online help available for more detailed information about the UI elements available on each page.



Note:



- On the **R/3 Job Definition** page, when you modify the **Job Class**, **Target Host**, or **Server Group** and click **OK**, the **Job ID** is maintained and remains synchronized with the one associated to the current job. Instead, when you modify the **Job Name** and click **OK**, the **Job ID** is automatically replaced with the one associated to the new job name.
- On the **R/3 Steps** page, for each step you modify, the new step information is saved in the SAP database. For each step you add or delete, the **Job ID** is maintained and remains synchronized with the one associated to the modified step.

9. Click **OK** to save your changes.

Task string to define SAP jobs

This section describes the task string parameters that define and control the running of SAP jobs. You can specify them in the following places when you define their associated IBM Workload Scheduler jobs:

- In the **SAP Command Line** section of the Task page of the Submit Ad Hoc Jobs action from the Dynamic Workload Console.
- In the **SAP Command Line** field of the More Options page of the SAP job definition, if you use the Dynamic Workload Console and selected a **SAP** job definition.
- As arguments of the `scriptname` keyword in the job definition statement, if you use the IBM Workload Scheduler command line.
- As arguments of the `JOBCMD` keyword in the `JOBREC` statement in the `SCRIPTLIB` of IBM Z Workload Scheduler, if you are scheduling in an end-to-end environment. The following is an example of a `JOBREC` statement:

```
JOBREC
  JOBCMD('/-job job_name -user user_name -i job_ID -c class_value')
  JOBUSR(TWS_user_name)
```

where:

class_value

The priority with which the job runs in the SAP system. For details, see [Table 26: Task string parameters for SAP jobs on page 265](#).

job_ID

The unique SAP job ID. For details, see [Table 26: Task string parameters for SAP jobs on page 265](#).

job_name

The name of the SAP job to run. For details, see [Table 26: Task string parameters for SAP jobs on page 265](#).

user_name

The SAP user who owns the target job. For details, see [Table 26: Task string parameters for SAP jobs on page 265](#).

TWS_user_name

The IBM Z Workload Scheduler user who runs the `r3batch` access method from the end-to-end scheduling environment.

The string syntax is the following:

Figure 18. Job definition syntax

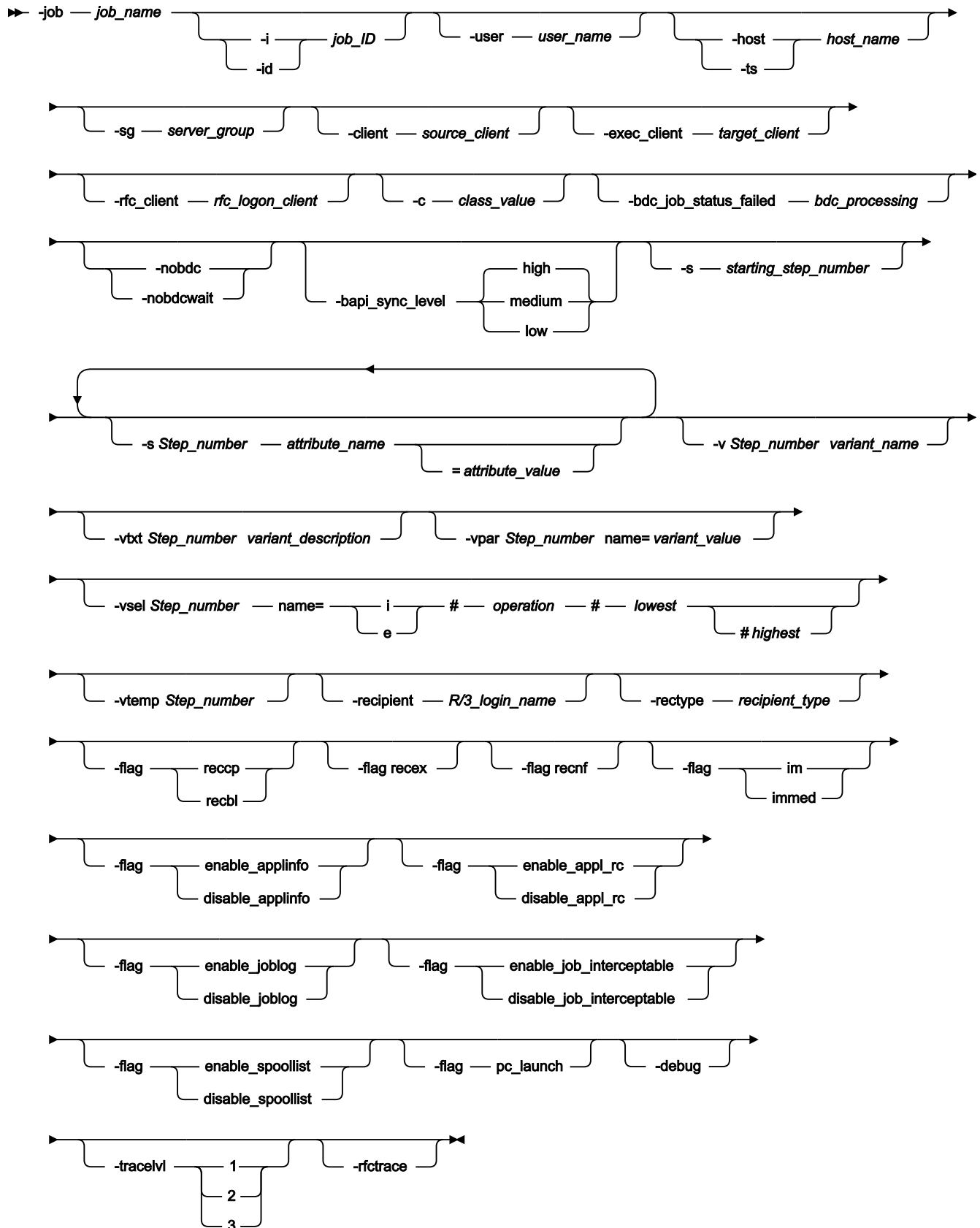


Table 26: Task string parameters for SAP jobs on page 265 describes the parameters for the task string to define SAP jobs.

**Note:**

1. You can specify both `-i` or `-id` and `-user` in the same job definition, but the user name is ignored.
2. When you specify the job ID, both `-client` and `-exec_client` are ignored because the ID is unique for the entire SAP system.
3. Typically, the `-debug` and `-trace` options are for debugging the extended agent and should not be used in standard production.

Table 26. Task string parameters for SAP jobs

Section	Parameters	Description	GUI Support
JOB	<code>-job job_name</code>	The name of the job to run. This parameter is mandatory.	✓
	<code>-i job_ID</code>	The unique SAP job ID. Specify this parameter if you are submitting a job that refers to a predefined job template stored in the SAP database for which you want to change a parameter.	✓
	<code>-id job_ID</code>		
	<code>-user user_name</code>	The SAP user who owns the target job. Use this parameter when the target SAP system has only one job with the specified name for the specified user. This parameter has no effect if a job ID is specified in the job definition.	✓
	<code>-host host_name</code>	The name of the SAP workstation where the job is to be run. <i>host_name</i> has the format <i>hostname_SAPsystemname_SAPsystemnumber</i> . For example, the name of a host might be <i>amss80a0_gs7_90</i>	✓
	<code>-ts host_name</code>		
	<code>-sg server_group</code>	These parameters are mutually exclusive with <code>-sg</code> . The name of the SAP server group where the job is to be run. Use this parameter to run the job on an application server that belongs to the group. The server group must exist on the SAP system, otherwise an error code is returned and the job is not launched.	✓

Table 26. Task string parameters for SAP jobs

(continued)

Section	Parameters	Description	GUI Support
		This parameter is case-sensitive and can be up to 20 characters. It is mutually exclusive with -host and -ts.	
JOB	-client <i>source_client</i>	The number that identifies the SAP client where the job definition is to be found, regardless of the client number defined by the <code>r3client</code> keyword in the options file. This parameter has no effect if a job ID is specified in the job definition.	
	-exec_client <i>target_client</i>	The number that identifies the SAP client where the job is to be run, regardless of the client number defined by the <code>r3client</code> keyword in the options file. This parameter has no effect if a job ID is specified in the job definition.	
	-rfc_client <i>rfc_logon_client</i>	The number that identifies the SAP client to be used for RFC logon. This value overwrites the value specified by the <code>r3client</code> keyword in the corresponding <code>r3batch</code> options file.	
	-c <i>class_value</i>	The priority with which the job runs in the SAP system. Possible values are: A High priority B Medium priority C Low priority. This is the default value.	✓
	-bdc_job_status_failed <i>bdc_processing</i>	How IBM Workload Scheduler sets the completion status of a job running BDC sessions, according to a possible BDC processing failure. The allowed values are: n If at least <i>n</i> BDC sessions failed (where <i>n</i> is an integer greater than 0), IBM Workload Scheduler sets the job completion status as failed.	✓

Table 26. Task string parameters for SAP jobs

(continued)

Section	Parameters	Description	GUI Support
		all	
		If all the BDC sessions failed, IBM Workload Scheduler sets the job completion status as failed.	
		ignore	
		When all the BDC sessions complete, regardless of their status, IBM Workload Scheduler sets the job completion status as successful. This is the default.	
		If -nobdc or -nobdcwait is set, this option is ignored.	
	-nobdc	Disables the BDC Wait option (enabled by default) to have the job considered as completed even if not all its BDC sessions have ended.	✓
	-nobdcwait		
	-bapi_sync_level	Specifies the synchronization level between the SAP function modules BAPI_XBP_JOB_COPY and BAPI_XBP_JOB_START_ASAP. Allowed values are:	
		high	
		All RFC calls between BAPI_XBP_JOB_START_ASAP and BAPI_XBP_JOB_COPY are synchronized. This is the default.	
		medium	
		The RFC calls to BAPI_XBP_JOB_START_ASAP are synchronized.	
		low	
		The RFC calls are not synchronized.	
STEP	-s <i>starting_step_number</i>	The number of the starting step.	✓
	-s <i>Step_number</i> <i>attribute_name=attribute_value</i>	The step number and its attributes, where:	✓

Table 26. Task string parameters for SAP jobs

(continued)

Section	Parameters	Description	GUI Support
		<i>step_number</i> The number of the step being defined. Each step is identified by a sequential number (1, 2, 3, ... <i>n</i>) using the <i>step number</i>	
		<i>attribute_name</i> The name of the attribute.	
		<i>attribute_value</i> The value of the attribute. It is optional for some attributes.	
		Attributes can be defined in any order, but cannot be repeated for the same step. Attribute validation is performed before the job is created in the SAP system. If the validation fails, the IBM Workload Scheduler job goes into the ABEND state. For a detailed description of each attribute and its value, see Defining attributes for ABAP steps on page 293 and Defining attributes for external programs and external commands steps on page 297 .	
		For example, the following step (step 8) is an ABAP module running the report MYPGM and has two attributes, only one of which has a value.	
		<pre>-s8 type=A -s8 program=MYPGM -s8 pr_cover="My title" -s8 pr_immed</pre>	
VARIANT	<code>-vstep_number name</code>	The variant name for the specified step number.	✓
	<code>-vtxtstep_number variant_description</code>	The textual description of the variant, in the IBM Workload Scheduler logon language (customizable with the <code>TWSXA_LANG</code> option of <code>r3batch</code>). The maximum length is 30 characters.	✓
	<code>-vparstep_number name=value</code>	For ABAP modules only. The value for a variant parameter for the specified step number. This parameter is mandatory when creating a new variant. See Defining attributes for ABAP steps on page 293 for a complete list of the supported attributes for ABAP steps.	✓

Table 26. Task string parameters for SAP jobs

(continued)

Section	Parameters	Description	GUI Support
	-vselstep_number	For ABAP modules only. The value for a variant selection option for the specified step number.	✓
	name=sign#operation#lowest[#highest]	sign Sign of the operation. Possible values are: I Include E Exclude operation Possible values are: EQ Equals NE Not equal to BT Between NB Not between LT Less than LE Less than or equal to GT Greater than GE Greater than or equal to CP Contains pattern	

Table 26. Task string parameters for SAP jobs

(continued)

Section	Parameters	Description	GUI Support
NP			
		Does not contain pattern	
		lowest	
		Low value of the selection. You can use up to 45 characters.	
		highest	
		High value of the selection. You can use up to 45 characters. This attribute is optional.	
		For a complete list of the supported attributes for ABAP steps, see Defining attributes for ABAP steps on page 293 .	
	-vtempstep_number	For ABAP modules only. Specifies to assign a temporary variant to the specified step number. Temporary variants are created ad-hoc by the SAP system and assigned to the job instance when it is run. The lifecycle of the temporary variant is determined by the SAP system. If the job is deleted by SAP, then the temporary variant is deleted. See Examples: Dynamically defining and updating SAP jobs on page 298 to refer to examples that demonstrate the behavior of temporary variants.	
SPOOL	-recipient name	The login name of an SAP user.	
	-flag {reccp recbl}	Specifies how the spool list is sent to the recipient. Possible values are: reccp The spool list is sent as a copy. recbl The spool list is sent as a blind copy.	
	-flag recex	Specifies that the spool list is sent as an express message to the recipient.	
	-flag recnf	Specifies that the recipient is not allowed to forward the spool list.	

Table 26. Task string parameters for SAP jobs

(continued)

Section	Parameters	Description	GUI Support
	-rectype type	Specifies the recipient type. Possible values are:	
		''	
		SAP user (default value)	
	'B'	SAP user	
	'C'	Shared distribution list	
	'D'	X.500 address	
	'G'	Organization object/ID	
	'H'	Organization unit	
	'I'	SAP object	
	'L'	Telex number	
	'O'	SAPoffice user	
	'P'	Private distribution list	
	'R'	SAP user in another SAP system	
	'U'	Internet address	
	'1'	Other recipient type	

Table 26. Task string parameters for SAP jobs

(continued)

Section	Parameters	Description	GUI Support
FLAGS	-flag im	Specifies to launch job immediately, meaning that if there are no spare work processes, the job fails.	✓
	-flag immed		
	-flag enable_applinfo	Enables or disables the retrieval and appending of the SAP application log to the stdlist of IBM Workload Scheduler.	✓
	-flag disable_applinfo		
	-flag enable_appl_rc	Enables or disables the mapping of the SAP application return code to the IBM Workload Scheduler return code.	
	-flag disable_appl_rc		
		The SAP application return code is mapped only if -flag enable_applinfo is set and the application log contains the application return code.	
	-flag enable_joblog	Enables or disables retrieval of the joblog.	✓
	-flag disable_joblog		
	-flag enable_job_interceptable	Enables or disables the job launched by r3batch to be intercepted by SAP. If enabled, when r3batch launches a job and the SAP job interception feature is enabled, the job can be intercepted if it matches previously defined criteria. If disabled, the job launched by r3batch cannot be intercepted by SAP. This setting overwrites the setting in the common options file.	✓
	-flag disable_job_interceptable		
	-flag enable_spoollist	Enables or disables retrieval of the spool lists of the job.	✓
	-flag disable_spoollist		
-flag pc_launch	Specifies to launch child jobs that are in scheduled state.		
	ON		
	The product launches child jobs that are in scheduled state.		
	OFF		
	The product does not launch child jobs that are in scheduled state. This is the default value.		

Table 26. Task string parameters for SAP jobs

(continued)

Section	Parameters	Description	GUI Support
	 Note: You can use this option only if you activated the parent-child feature on the SAP system. On the XBP 2.0 (or later)SAP system you can activate this feature using the INITXBP2 ABAP report		
TRACING	-debug	Enables maximum trace level.	✓
	-tracelvl 1 2 3	Specifies the trace setting for the job. Possible values are: 1 Only error messages are written in the trace file. This is the default. 2 Informational messages and warnings are also written in the trace file. 3 A most verbose debug output is written in the trace file. For detailed information, refer to Configuring the tracing utility on page 207 . Enables RFC trace.	✓
	-rfctrace		
	-trace		

The following is an example for an SAP job named `BVTTEST` with ID `03102401` and user `myuser`:

```
-job BVTTEST -i 03102401 -user myuser -debug
```

Managing SAP jobs

This section describes how to manage SAP jobs.

Displaying details about a standard SAP job

About this task

Perform the following steps to display details for standard jobs on specific workstations.

For information about how to display details about a job that submits an SAP process chain, refer to [Displaying details about a process chain job on page 324](#).

1. Click **Design > SAP > Manage Jobs on SAP**.
2. In **Engine name**, select the name of the IBM Workload Scheduler engine connection from which you want to view SAP job details.
3. In **Workstation name**, type the name of the workstation where the SAP job runs. This is the workstation with the r3batch access method that communicates with the remote SAP system. If you do not know the name of the workstation, click (...) browse to enter your filter criteria and click **Search**. If you enter a string representing part of the workstation name, it must be followed by the asterisk (*) wildcard character. Both the question mark (?) and asterisk (*) are supported as wildcards. You can also simply use the asterisk wildcard character (*) to display all workstations. Optionally, specify any of the other search criteria available and click **Search**.

Result

From the results displayed, select the workstation and click **OK**.

4. In **Options file**, specify an options file that resides on the specified workstation. Each workstation can have one or more options files that can be used to customize the behavior of the r3batch access method, except for extended agent workstations, where only one options file can exist and therefore does not need to be specified. For the workstation specified, enter the file name of the options file or click the browse (...) button to search for options files that reside on the specified workstation and select one.
5. Click **Display**. The list of available jobs for the specified engine is displayed.
6. Select the job for which you want to display the details and click **Details**. The List Jobs on SAP panel is displayed containing job and time information.
7. When you have finished viewing the details for the job, click **OK** to return to the list of SAP jobs on the workstation specified.

Verifying the status of a standard SAP job

About this task

To verify the status of a standard SAP job, perform the following steps:

1. Click **Design > SAP > Manage Jobs on SAP**.
2. In **Engine name**, select the name of the IBM Workload Scheduler engine connection from which you want to verify the status of an SAP job.
3. In **Workstation name**, type the name of the workstation where the SAP job runs. This is the workstation with the r3batch access method that communicates with the remote SAP system. If you do not know the name of the workstation, click (...) browse to enter your filter criteria and click **Search**. If you enter a string representing part of the workstation name, it must be followed by the asterisk (*) wildcard character. Both the question mark (?) and asterisk (*) are supported as wildcards. You can also simply use the asterisk wildcard character (*) to display all workstations. Optionally, specify any of the other search criteria available and click **Search**.

(*) are supported as wildcards. You can also simply use the asterisk wildcard character (*) to display all workstations. Optionally, specify any of the other search criteria available and click **Search**.

Result

From the results displayed, select the workstation and click **OK**.

4. In **Options file**, specify an options file that resides on the specified workstation. Each workstation can have one or more options files that can be used to customize the behavior of the r3batch access method, except for extended agent workstations, where only one options file can exist and therefore does not need to be specified. For the workstation specified, enter the file name of the options file or click the browse (...) button to search for options files that reside on the specified workstation and select one.
5. Click **Display**. The list of available jobs for the specified engine is displayed.
6. Select the job for which you want to verify the status and click **Status**. The current status for the SAP job is displayed, as well as the database name where the job is installed.
7. When you have finished verifying the status for the job, click **OK** to return to the list of SAP jobs on the workstation specified.

Deleting a standard SAP job from the SAP database

About this task

To delete a standard SAP job from the SAP database, perform the following steps:

1. Click **Design > SAP > Manage Jobs on SAP**.
2. In **Engine name**, select the name of the IBM Workload Scheduler engine connection from which you want to delete the SAP job.
3. In **Workstation name**, type the name of the workstation where the SAP job runs. This is the workstation with the r3batch access method that communicates with the remote SAP system. If you do not know the name of the workstation, click (...) browse to enter your filter criteria and click **Search**. If you enter a string representing part of the workstation name, it must be followed by the asterisk (*) wildcard character. Both the question mark (?) and asterisk (*) are supported as wildcards. You can also simply use the asterisk wildcard character (*) to display all workstations. Optionally, specify any of the other search criteria available and click **Search**.

Result

From the results displayed, select the workstation and click **OK**.

4. In **Options file**, specify an options file that resides on the specified workstation. Each workstation can have one or more options files that can be used to customize the behavior of the r3batch access method, except for extended agent workstations, where only one options file can exist and therefore does not need to be specified. For the workstation specified, enter the file name of the options file or click the browse (...) button to search for options files that reside on the specified workstation and select one.
5. Click **Display**. The list of available jobs for the specified engine is displayed.
6. Select the job or jobs you want to delete and click **Delete**. A confirmation message prompts you to confirm the delete action.
7. When the delete action is complete, click **OK** to return to the list of SAP jobs on the workstation specified.

Balancing SAP workload using server groups

SAP jobs run on application servers that host work processes of type *batch*. Critical batch jobs are run in specific time frames, on specific application servers. With SAP Basis version 6.10 and later, application servers can be assigned to server groups. With IBM Workload Scheduler you can assign a server group to a job. In this way, when a job is launched, the SAP system runs it on an application server that belongs to that group, balancing the workload among the various application servers in the group.

If the application servers defined in a group are modified in the SAP system, the job defined as belonging to that server group is not affected and does not need to be modified. The batch execution targets are reorganized in the SAP system without having to change job definitions in IBM Workload Scheduler.

This function is supported with the following versions of SAP:

- SAP Basis 6.10, with Service Pack 40
- SAP Basis 6.20, with Service Pack 41
- SAP Basis 6.40, with Service Pack 04
- SAP Basis 7.00, and later

Mapping between IBM Workload Scheduler and SAP job states

When an SAP job is launched by IBM Workload Scheduler, you can monitor its progress. The status transitions in IBM Workload Scheduler (internal status) and the corresponding SAP status are listed in [Table 27: Status transitions in IBM Workload Scheduler \(internal status\) and the corresponding SAP R/3 status on page 276](#).

Table 27. Status transitions in IBM Workload Scheduler (internal status) and the corresponding SAP R/3 status

IBM Workload Scheduler Job State	SAP Job State
INTRO	Not Available
WAIT	Ready, Release
EXEC	Active
SUCC	Finished
ABEND	Canceled

The INTRO state indicates that IBM Workload Scheduler is in the process of introducing the job, but in SAP, the job has not yet entered the ready state. Because it takes some time to get a job queued and into the ready column, the INTRO state might last a few minutes if the SAP system is particularly busy.

Even if a job is finished in SAP, IBM Workload Scheduler keeps it in the EXEC state if its BDC sessions are not complete and you have not selected the **Disable BDC Wait** option. For details about this option, see [Using the BDC Wait option on page 304](#).

Managing spools

Browse spool lists on request without having to download the entire spool which can occupy significant space on the file system.

Spool lists can be very large so rather than download them as part of a job run, you can request to browse the spool list, chunks at a time, even if you have disabled the option, `retrieve_spoollist`, to append the spool list to the IBM Workload Scheduler joblog.

From the Dynamic Workload Console, you can list the spool data available for SAP Standard R/3 jobs that have run. Each spool is identified by the following information:

- The spool number.
- The related step number.
- The name of the spool request.
- The title of the spool request.
- The total number of pages for the spool information.
- The user who executed the SAP job related to the spool.
- The date the spool was created based on the Coordinated Universal Time (UTC) time standard.
- The client for which the spool was created.

Browsing spool data

You can list the spool data available for SAP Standard R/3 jobs that have run and browse the contents of the spool.

About this task

To browse spool data for a specific job that has run:

1. From the **Monitoring and Reporting** menu, click **Orchestration Monitor**.
2. In the **Monitor Workload** input fields enter the engine name, the plan, and any filtering data that helps you filter the selection of jobs (you can also select **Edit** for a guided selection of filtering criteria) and select **Run**.
3. In the output table select an SAP Standard R/3 job and click **More Actions > Show Spool List**. The list of spool data available for the selected job is displayed.
4. Select a spool and click **Spool**.

Results

By default, the first ten pages of the spool are made available. You can change this default by editing the number of pages specified in **Pages for screen**. Use the page functions to jump to a specific page number, jump to the last page of the spool, jump to the first page of the spool, or move forward or back through the number of pages indicated by **Pages for screen**.

Killing an SAP job instance

About this task

This section describes how to kill an IBM Workload Scheduler job that submits either a standard SAP job or an SAP process chain.

To kill an SAP job instance, do the following:

The IBM Workload Scheduler job status is set to ABEND. The SAP job or process chain is set to canceled in the SAP system.



Note: If you kill a process chain job, the SAP system stops as soon as the process that is currently running completes.

1. Use the **Monitor Workload** query of the Dynamic Workload Console to display a list of defined job instances containing the job you want to kill. From the **Monitoring and Reporting** menu, click **Orchestration Monitor**.
2. In the **Monitor Workload** input fields enter the engine name, the plan, and any filtering data that helps you filter the selection of jobs (you can also select **Edit** for a guided selection of filtering criteria) and select **Run**.
3. The **Monitor Jobs** panel is displayed. Select the job instance you want to kill and click **More Actions > Kill**.

Raising an SAP event

About this task

You can raise events on XBP 2.0 (or later) SAP jobs in the IBM Workload Scheduler database in one of the following ways:

Using the Monitor Workload in the Dynamic Workload Console

Perform the following steps:

1. On the SAP system, create a job that has as start condition a SAP event. When you create this job, its status is released.
2. Check that this job was not intercepted by the interception function.
3. Log in to the Dynamic Workload Console.
4. From the **Monitoring and Reporting** menu, click **Orchestration Monitor**.
5. In the Monitor Workload window select the engine, enter **Workstation** in the Object Type field, and select the plan to display the list of workstations you want to monitor. Click **Run**.

A list of workstations is displayed.

6. Select a workstation that has been defined to connect to a remote SAP system.
7. From the toolbar, select **More Actions > Raise Event**. The Raise Event panel opens.

Figure 19. The Raise Event panel

The screenshot shows the 'XA700AIX Raise Event' dialog box. The title bar includes a search icon, a star icon, and dropdown menus for 'Default', 'Administration', and 'Planning'. Below the title bar, there are three tabs: 'MONITOR JOBS', 'MONITOR JOBS(2)', and 'MONITOR JOBS(3)'. The main content area is titled 'XA700AIX Raise Event'. Underneath, there is a section titled 'Raise Event' with two input fields: '*Event Id' (containing 'SAP_EVENT_ID') and 'Event Parameter' (empty). At the bottom of the dialog are 'OK' and 'Cancel' buttons.

8. The panel consists of the following:

Event ID

The identifier of the event that is to be raised.

Event Parameter

The parameter of the event that is to be raised.

9. Click **OK**. The event is raised.

Creating a job that launches a Windows™ or UNIX™ command that raises an event

Perform the following steps:

1. From the Design menu, click Workload Designer page.
2. Specify an engine name, either distributed or z/OS. The Workload Designer window opens. Job types and characteristics vary depending on whether you select a distributed or a z/OS engine.
3. Click **Create new** and select **Job definition**.
4. Select the **Native** category and then either **Windows** or **UNIX** job.
5. Use the **General** section to provide general information about the new job definition.
6. Use the **Task** section to provide task information for the job.
7. In the **Task** section, select **Command** and in the command string type the following command that raises the event:

```
<data_dir>/methods/r3event -c workstation_name -u user_name
-e SAP_event_ID -p parameter
```

where:

workstation_name

The name of the workstation where the SAP job is defined.

user_name

The name of the SAP user with which the access method connects to the SAP system.

This is the name specified in the r3user option.

SAP_event_ID

The identifier of the event.

parameter

The parameter defined for the event.

8. Save the job definition.

What to do next

See [Defining conditions and criteria on page 300](#) for information about how to define criteria that manages which raised events to log.

Rerunning a standard SAP job

You can rerun a standard SAP job from the start, or from a specific numeric step of the SAP instruction.

About this task

To rerun a standard SAP job, you can use one of the following user interfaces:

conman

For details, refer to the *IBM Workload Scheduler: User's Guide and Reference*.

Dynamic Workload Console

Dynamic Workload Console

For details about how to rerun a job that submits an SAP process chain, refer to [Rerunning a process chain job on page 326](#).

For an SAP extended agent, a **step** is the numeric step of the SAP instruction from which a job can be restarted. Before you rerun an SAP job with IBM Workload Scheduler, you have the option of providing a step name for the job. This affects `r3batch` in the following ways:

- If you use a step name that is up to 9 digits (or 8 digits preceded by a character) in length, this name is used as the starting step number for the rerunning job.
- If you use any different format, the name is ignored and the job is rerun starting from the first step.

For example, to rerun a job from the third step, you can use: `A03`, `3`, `00003`, or `H3`.

In z/OS environments, you need to set the status of the job to **Ready** before you can rerun the job.



Note: By default, if you specify a job step to rerun, the new job is assigned the name of the step you indicated. To keep the original job name, set the IBM Workload Scheduler global option **enRetainNameOnRerunFrom** to `yes`. This option works only when used with the following arguments: `rr jobselect;from=[wkstat#]job`. For details about these arguments, see *IBM Workload Scheduler: User's Guide and Reference, Managing objects in the plan - conman, Conman commands, rerun*. For details about this option, see *IBM Workload Scheduler: Planning and Installation*.

When `r3batch` reruns a job from its first step, either because you specified it as the starting step or because no starting step was specified, it uses the `new copy` feature, if applicable. If the starting step is greater than one, `r3batch` uses the `old copy` to rerun the job. For a description about the difference between the new and old copy of a rerunning job, refer to [Old copy and new copy of a rerunning job on page 281](#).

To rerun a SAP Standard R/3 job from the Dynamic Workload Console, perform the following steps:

1. In the Dynamic Workload Console select **Monitoring & Reporting > Workload Monitoring > Monitor Workload**.
2. In the Monitor Workload input fields select **Job** as the Object Type, the engine name, the plan, and any filtering data that helps you filter the selection of jobs (you can also select **Edit** for a guided selection of filtering criteria) and select **Run**.
3. A list of jobs is displayed. Select an SAP Standard R/3 job.

4. Rerun the job.

Distributed environment

- a. Click **Rerun....** The General properties for the rerun operation are displayed.
- b. Optionally, you can choose to not rerun the same job but instead, substitute the selected SAP job with a different job definition and run it. Type the job definition name in the **From Job Definition** field, or use the browse button to search for it and select it.
- c. Optionally, type the workstation name of the workstation on which you want to rerun the job in the **Workstation Name** field.
- d. Optionally, in **Step**, enter a specific numeric step of the SAP instruction from which you want to rerun the job rather than rerunning the whole job.
- e. Optionally, specify the start and finish time for the job.
- f. Click **Rerun**.

The job reruns immediately or at the specified start time.

z/OS environment

In a z/OS environment, an alias for the job name is not required so the job reruns with the same name. The list of jobs always reports the latest action performed on the job.

- a. Before you can rerun a job, you must change the status of the job to **Ready**. Select a job and click **Set Status**.
- b. In **Change Status**, select **Ready**.
- c. Click **OK** to return to the list of jobs.

The job reruns immediately and the internal status reports **Started**.

Old copy and new copy of a rerunning job

When the access method for SAP launches a job, it makes a copy of a template job and runs it.

The **new copy** feature is available for SAP versions 3.1i, and later. It copies an entire job, preserving steps, job class, and all print and archive parameters. It is performed by using a new SAP function module that is part of the SAP Support Package as stated in the SAP Notes 399449 and 430087.

The **old copy** feature, instead, is based on standard SAP function modules, and creates a new SAP job and adds the steps with a loop that starts from the step name or number you specified. Be aware that, unless you have XBP 2.0 or later:

- The **old copy** does not preserve all the print and archive parameters.
- The job class of the copy is always set to class **c**.

Refer to [Print parameter and job class issues on page 227](#) to learn how to resolve the problem of lost job class and print and archive parameters.

SAP Note 758829 is required to ensure correct operation of the **new copy** and **old copy** features. See also [Table 51: Miscellaneous troubleshooting items on page 376](#).

Defining SAP jobs dynamically

This section describes how to create and submit SAP jobs dynamically without creating or referencing predefined job templates.

When you launch a job created as described in [Creating SAP Standard R/3 jobs from the Dynamic Workload Console on page 254](#) and [Task string to define SAP jobs on page 262](#), IBM Workload Scheduler makes a copy of the predefined job (also known as a template job) and runs the copy. If you want to run the job on several SAP systems, you must manually create the template job on each system.

To create and submit SAP jobs dynamically, without creating or referencing predefined job templates, submit:

- In the SAP system, a job that does not reference an existing template in the SAP R/3 database.
- A job that references a predefined job template stored in the SAP R/3 database for which you want to change a parameter.

To take full advantage of this feature, make sure that you have XBP version 2.0 or later installed, because earlier versions of XBP do not support the full set of print and archive parameters, or provide a way to set the job class or the spool list recipient.

Task string to define SAP jobs dynamically

This section describes the task string that controls the running of SAP jobs. You can build an entire job definition by using the six main sections concerning SAP job parameters. These sections are grouped in [Table 28: Task string parameters for SAP jobs \(dynamic definition\) on page 285](#) and are related to the:

- Job
- Job steps
- Variants associated with the steps (for ABAP modules only)
- Spool list recipients associated with the job
- Flags associated with the job
- Tracing specifications for the job

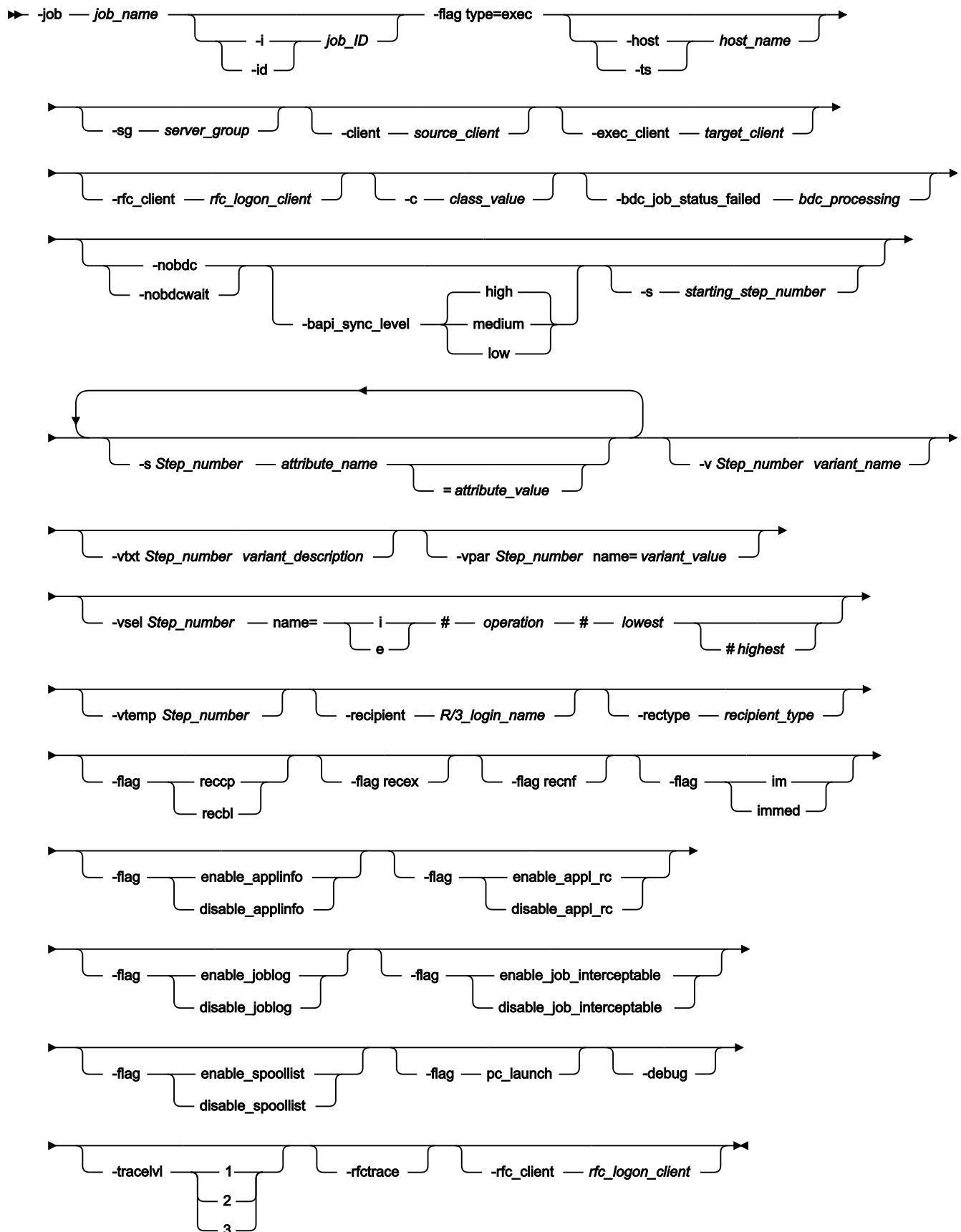
You can specify them in the following places when you define their associated IBM Workload Scheduler jobs:

- In the **SAP Command Line** section of the Task page of the Submit Ad Hoc Jobs action from the Dynamic Workload Console.
- In the **SAP Command Line** field of the More Options page of the SAP job definition, if you use the Dynamic Workload Console and selected a **SAP** job definition.
- As arguments of the `scriptname` keyword in the job definition statement, if you use the IBM Workload Scheduler command line.
- As arguments of the `JOBCMD` keyword in the `JOBREC` statement in the `SCRIPTLIB` of IBM Z Workload Scheduler, if you are scheduling in an end-to-end environment. The following is an example of a `JOBREC` statement:

```
JOBREC  
  JOBCMD('/-job job_name -user user_name -i job_ID -c class_value')  
  JOBUSR(TWS_user_name)
```

To define and submit an SAP job dynamically, use the following syntax:

Figure 20. Job definition syntax



The following is an example of a definition for the SAPTEST job:

```
-job SAPTEST -C A -s1 program=BTCTEST -s1 type=A -s1 pr_release
-s2 report=BTCTEST -s2 variant=BVT -s2 type=A -flag type=exec
-vpar2 TESTNAME=test -vtxt2 Test
```

Table 28: Task string parameters for SAP jobs (dynamic definition) on page 285 describes the parameters for the task string to define SAP jobs dynamically.



Note: The parameter values are case sensitive.

Table 28. Task string parameters for SAP jobs (dynamic definition)

Section	Parameters	Description
JOB	-job <i>job_name</i>	The name of the job to be run. This parameter is mandatory.
	-i <i>job_ID</i>	The unique SAP job ID. Specify this parameter if you are submitting a job that refers to a predefined job template stored in the SAP database for which you want to change a parameter.
	-id <i>job_ID</i>	
	-host <i>host_name</i>	The name of the SAP workstation where the job is to be run. <i>host_name</i> has the format <i>hostname_SAPsystemname_SAPsystemnumber</i> . For example, the name of a host might be amss80a0_gs7_90
	-ts <i>host_name</i>	
		These parameters are mutually exclusive with -sg.
	-sg <i>server_group</i>	The name of the SAP server group where the job is to be run. Use this parameter to run the job on an application server that belongs to the group. The server group must exist on the SAP system, otherwise an error code is returned and the job is not launched.
		This parameter is case-sensitive and can be up to 20 characters. It is mutually exclusive with -host and -ts.
	-client <i>source_client</i>	The number that identifies the SAP client where the job definition is to be found, regardless of the client number defined by the r3client key in the options file. This parameter has no effect if a job ID is specified in the job definition.
	-exec_client <i>target_client</i>	The number that identifies the SAP client where the job is to be run, regardless of the client number defined by the r3client key in the options file. This parameter requires that

Table 28. Task string parameters for SAP jobs (dynamic definition)

(continued)

Section	Parameters	Description
		the client-dependent data (such as the user name and report variants) exists on both the source and target clients.
		This parameter has no effect if a job ID is specified in the job definition.
	-rfc_client <i>rfc_logon_client</i>	The number that identifies the SAP client to be used for RFC logon. This value overwrites the value specified by the r3client keyword in the corresponding r3batch options file.
	-c <i>class_value</i>	The priority with which the job runs in the SAP system. Possible values are: A High priority B Medium priority C Low priority. This is the default value.
	-flag type=exec	Specify this parameter to enable the dynamic definition of the SAP job. This parameter is mandatory.
JOB	-bdc_job_status_failed <i>bdc_processing</i>	How IBM Workload Scheduler sets the completion status of a job running BDC sessions, according to a possible BDC processing failure. The allowed values are: n If at least <i>n</i> BDC sessions failed (where <i>n</i> is an integer greater than 0), IBM Workload Scheduler sets the job completion status as failed. all If all the BDC sessions failed, IBM Workload Scheduler sets the job completion status as failed. ignore When all the BDC sessions complete, regardless of their status, IBM Workload

Table 28. Task string parameters for SAP jobs (dynamic definition)

(continued)

Section	Parameters	Description
		Scheduler sets the job completion status as successful. This is the default value.
		If -nobdc or -nobdcwait is set, this option is ignored.
	-nobdc -nobdcwait	Disables the BDC Wait option (enabled by default) to have the job considered as completed even if not all its BDC sessions have ended.
	-bapi_sync_level	Specifies the synchronization level between the SAP function modules BAPI_XBP_JOB_COPY and BAPI_XBP_JOB_START_ASAP. Allowed values are: high All RFC calls between BAPI_XBP_JOB_START_ASAP and BAPI_XBP_JOB_COPY are synchronized. This is the default. medium The RFC calls to BAPI_XBP_JOB_START_ASAP are synchronized. low The RFC calls are not synchronized.
STEP	-s <i>starting_step_number</i> -sstep_number <i>attribute_name=attribute_value</i>	The number of the starting step. The step number and its attributes, where: <i>step_number</i> The number of the step being defined. Each step is identified by a sequential number (1, 2, 3, ... <i>n</i>) using the <i>step number</i> . <i>attribute_name</i> The name of the attribute. <i>attribute_value</i> The value of the attribute. It is optional for some attributes.

Table 28. Task string parameters for SAP jobs (dynamic definition)

(continued)

Section	Parameters	Description
		Attributes can be defined in any order, but cannot be repeated for the same step. Attribute validation is performed before the job is created in the SAP system. If the validation fails, the IBM Workload Scheduler job goes into the ABEND state. For a detailed description of each attribute and its values, see Defining attributes for ABAP steps on page 293 and Defining attributes for external programs and external commands steps on page 297 .
		For example, the following step (step 8) is an ABAP module running the report "MYPGM" and has two attributes, only one of which has a value.
		<pre>-s8 type=A -s8 program=MYPGM -s8 pr_cover="My title" -s8 pr_immed</pre>
VARIANT ¹	<i>-vstep_number name</i> <i>-vtxtstep_number variant_description</i> <i>-vparstep_number name=value</i> <i>-vselstep_number</i> <i>name=sign#operation#lowest[#highest]</i>	The variant name for the specified step number. The textual description of the variant, in the IBM Workload Scheduler logon language (customizable with the <code>TWSXA_LANG</code> option of <code>r3batch</code>). The maximum length is 30 characters. Not valid for temporary variants. For ABAP modules only. The value for a variant parameter for the specified step number. This parameter is mandatory when creating a new variant. For a complete list of the supported attributes for ABAP steps, see Defining attributes for ABAP steps on page 293. For ABAP modules only. The value for a variant selection option for the specified step number. sign Sign of the operation. Possible values are: I Include E Exclude operation Possible values are:

Table 28. Task string parameters for SAP jobs (dynamic definition)

(continued)

Section	Parameters	Description
	EQ	Equals
	NE	Not equal to
	BT	Between
	NB	Not between
	LT	Less than
	LE	Less than or equal to
	GT	Greater than
	GE	Greater than or equal to
	CP	Contains pattern
	NP	Does not contain pattern
	lowest	Low value of the selection. You can use up to 45 characters.
	highest	High value of the selection. You can use up to 45 characters. This attribute is optional.

For a complete list of the supported attributes for ABAP steps, see [Defining attributes for ABAP steps on page 293](#).

Table 28. Task string parameters for SAP jobs (dynamic definition)

(continued)

Section	Parameters	Description
	<i>-vtempstep_number</i>	For ABAP modules only. Specifies to assign a temporary variant to the specified step number. Temporary variants are created ad-hoc by the SAP system and assigned to the job instance when it is run. The lifecycle of the temporary variant is determined by the SAP system. If the job is deleted by SAP, then the temporary variant is deleted. See Examples: Dynamically defining and updating SAP jobs on page 298 to refer to examples that demonstrate the behavior of temporary variants.
SPOOL	<i>-recipient name</i>	The login name of an SAP user.
	<i>-flag {reccp recbl}</i>	Specifies how the spool list is sent to the recipient. Possible values are: reccp The spool list is sent as a copy. recbl The spool list is sent as a blind copy.
	<i>-flag recex</i>	Specifies that the spool list is sent as an express message to the recipient.
	<i>-flag recnf</i>	Specifies that the recipient is not allowed to forward the spool list.
	<i>-rectype type</i>	Specifies the recipient type. Possible values are: blank SAP user (default value) B SAP user C Shared distribution list D X.500 address G Organization object/ID

Table 28. Task string parameters for SAP jobs (dynamic definition)

(continued)

Section	Parameters	Description
		H
		Organization unit
		I
		SAP object
		L
		Telex number
		O
		SAPoffice user
		P
		Private distribution list
		R
		SAP user in another SAP system
		U
		Internet address
		1
		Other recipient type
FLAGS	-flag im	Specifies to launch the job immediately, meaning that if there are no spare work processes, the job fails.
	-flag immed	
	-flag enable_applinfo	Enables or disables the retrieval and appending of the SAP application log to the stdlist of IBM Workload Scheduler.
	-flag disable_applinfo	
	-flag enable_appl_rc	Enables or disables the mapping of the SAP application return code to the IBM Workload Scheduler return code.
	-flag disable_appl_rc	
		The SAP application return code is mapped only if -flag enable_applinfo is set and the application log contains the application return code.
	-flag enable_joblog	Enables or disables retrieval of the joblog.
	-flag disable_joblog	
	-flag enable_joblog	Enables or disables retrieval of the joblog.
	-flag disable_joblog	

Table 28. Task string parameters for SAP jobs (dynamic definition)

(continued)

Section	Parameters	Description
	-flag enable_job_interceptable -flag disable_job_interceptable	Enables or disables the job launched by r3batch to be intercepted by SAP. If enabled, when r3batch launches a job and the SAP job interception feature is enabled, the job can be intercepted if it matches previously defined criteria. If disabled, the job launched by r3batch cannot be intercepted by SAP. This setting overwrites the setting in the common options file.
-flag pc_launch	Specifies to launch child jobs that are in scheduled state. ON The product launches child jobs that are in scheduled state. OFF The product does not launch child jobs that are in scheduled state. This is the default value.  Note: You can use this option only if you activated the parent-child feature on the SAP system. On the XBP 2.0 (or later) SAP system, you activate this feature by using the INITXBP2 ABAP report.	
TRACING	-debug -tracelvl 1 2 3	Enables maximum trace level. Specifies the trace setting for the job. Possible values are: 1 Only error messages are written in the trace file. This is the default. 2 Informational messages and warnings are also written in the trace file.

Table 28. Task string parameters for SAP jobs (dynamic definition)

(continued)

Section	Parameters	Description
		3
		A most verbose debug output is written in the trace file.
		For more details, refer to Configuring the tracing utility on page 207 .
		Enables RFC trace.
	-rfctrace	
	-trace	



Note: See [Examples: Dynamically defining and updating SAP jobs on page 298](#) to refer to examples that demonstrate the behavior of variants and temporary variants.

1. The following rules apply when you create or update SAP jobs dynamically:
 - To create or reference a variant within an ABAP step, you can use one of the following equivalent syntaxes:
 - -s1 Variant=Var1
 - -s1 Parameter=Var1
 - -v1 Var1
 - If a variant does not exist, it is created with the parameters specified in the job definition statement. In this case, all the required attributes of the variant must be given a value. You cannot create empty variants. For example, if you specify **-vtemp1**, with no value assigned, an empty temporary variant is erroneously created.
 - If a variant is already present in the SAP system, its values are modified according to the command line parameters. If the existing variant is an extended one, a new instance of it is created with resolved placeholders and updated counters. This new variant instance is then updated using the values from the command line. Finally, the job step is run using this variant instance.
 - All changes to the variant values are permanent. That is, IBM Workload Scheduler neither restores the old values of the variants, nor deletes the variants created after the job is run. IBM Workload Scheduler does not change the case of the variant values.

Defining attributes for ABAP steps

To create and submit SAP jobs dynamically, look at the table and define the attributes for ABAP steps.

[Table 29: Supported attributes for ABAP step definition on page 294](#) shows a complete list of the supported attributes for ABAP step module definition:

Table 29. Supported attributes for ABAP step definition

Attribute name	Synonym	Description	Required
type	typ	Specify the step type. Possible values are: • A • ABAP The product performs a check for correct attribute values prior to launching the job.	✓
program		Specify the ABAP program name.	✓
parameter		Specify the ABAP variant name.	✓
user	authcknam	Specify the user of the step.	✓
language	lang	Specify the step language. This attribute accepts language names in either the ISO format (two characters, for example DE, EN) or the R/3 format (one character, for example D, E). If this attribute is not specified, the login language of the access method is used (customize using the option twsga_lang in the r3batch options files). The product performs a check for a valid language prior to launching the job.	✓
pr_dest	printer pdest	Print Parameter: Specify the printer for the output.	
pr_copies	prcop	Print Parameter: Specify the number of copies. The value of this attribute must be numeric. A corresponding check is performed prior to launching the job.	
pr_lines	linct	Print Parameter: Specify the page length. The value of this attribute must be numeric. A corresponding check is performed prior to launching the job.	
pr_columns	linsz	Print Parameter: Specify the page width. The value of this attribute must be numeric. A corresponding check is performed prior to launching the job.	
pr_auth	prber	Print Parameter: Authorization	
pr_arcmode	armod	Print Parameter: Archiving mode	
pr_sapbanner	prsap	Print Parameter: SAP cover page	

Table 29. Supported attributes for ABAP step definition (continued)

Attribute name	Synonym	Description	Required
pr_exp	pexpi	Print Parameter: Spool retention period The value of this attribute must be a single digit. A corresponding check is performed prior to launching the job.	
pr_recip	prrec	Print Parameter: Recipient	
pr_spoolname	plist	Print Parameter: Name of spool request ¹	
pr_format	paart	Print Parameter: Print formatting ¹	
pr_dep	prabt	Print Parameter: Department on cover page ¹	
pr_spools	prdsn	Print Parameter: Name of spool data set ¹	
pr_spoolprio	priot	Print Parameter: Spool request priority ¹	
pr_immed	primm	Print Parameter: Print immediately ²	
pr_release	prrel	Print Parameter: Delete after printing ²	
pr_banner	prbig	Print Parameter: Selection cover page ²	
pr_newspool	prnew	Print Parameter: New spool request ^{1 2}	
pr_cover	prtxt	Print Parameter: Text for cover page ¹ . If the string contains spaces it must be enclosed between single quotes.	
pr_hostcover	prunx	Print Parameter: Host spool cover page ¹ . Possible values are: ' ' Blank. Does not use any cover page. 'X' Prints the host cover page. 'D' Prints the default host cover page.	
al_sapobject	sap_object	SAP ArchiveLink: Object type of business object	
al_object	object	SAP ArchiveLink: Document type	
al_info	info	SAP ArchiveLink: Info field	
al_id	archiv_id	SAP ArchiveLink: Target storage system ¹	
al_doctype	doc_type	SAP ArchiveLink: Document class ¹	
al_rpchost	rpc_host	SAP ArchiveLink: PRC host ¹	
al_rpcserv	rpc_servic	SAP ArchiveLink: RPC service / RFC destination ¹	

Table 29. Supported attributes for ABAP step definition (continued)

Attribute name	Synonym	Description	Required
al_iface	interface	SAP ArchiveLink: Name of communication connection component ¹	
al_client	mandant	SAP ArchiveLink: Client ¹	
al_report		SAP ArchiveLink: Report name ¹	
al_text	arctext	SAP ArchiveLink: Text information field ¹	
al_date	datum	SAP ArchiveLink: Archiving date ¹	
al_user	arcuser	SAP ArchiveLink: Data element for user ¹	
al_printer		SAP ArchiveLink: Target printer ¹	
al_format	formular	SAP ArchiveLink: Output format ¹	
al_path	archivpath	SAP ArchiveLink: Standard archive path ¹	
al_protocol	protokoll	SAP ArchiveLink: Storage connection protocol ¹	
al_version		SAP ArchiveLink: Version number ¹	

**Note:**

1. This attribute is available for BC-XBP 2.0 and later.
2. This attribute is a flag, that is, it does not have a value, for example: `-s2 pr_release`.

IBM Workload Scheduler performs the following syntax validation on job attributes:

- Only valid attributes are allowed.
- Checks if a particular attribute requires a value.
- The values of the following attributes are checked:
 - type
 - language
 - pr_copies
 - pr_lines
 - pr_columns

Validation is performed before the job is created in the SAP system. If the validation fails, the IBM Workload Scheduler job goes into the ABEND state.

Defining attributes for external programs and external commands steps

Table 30: Supported attributes for external programs and external commands step definition on page 297 shows a complete list of the supported attributes for external programs and external commands step definition.

Table 30. Supported attributes for external programs and external commands step definition

Attribute name	Synonym	Description	Required
type	typ	The step type can assume one of the following values: For external programs • X • EXTPRG For external commands • C • EXTCMD Before launching the job, the product performs a check for correct attribute values.	✓
report		ABAP program name or name of the external program or command.	✓
parameter		Parameters for the external program or command.	
user	authcknam	User of the step.	
language	lang	Step language. This attribute accepts language names in either the ISO format (two characters, for example DE, EN) or the R/3 format (one character, for example D, E). If this attribute is not specified, the login language of the access method is used (customize using the twsga_lang option in the <code>r3batch</code> option files). The product performs a check for a valid language prior to launching the job.	
targethost	xpgtgtsys	Target host for the external program or command. This name must be exactly the same as the name shown in the External Operating System Commands table in the SAP system (transaction <code>sm69</code>).	
os	opsystem	Operating system for the external command. This name must be exactly the same as the name shown in the External Operating System Commands table in the SAP system (transaction <code>sm69</code>).	

Table 30. Supported attributes for external programs and external commands step definition (continued)

Attribute name	Synonym	Description	Required
termcntl	waitforterm	Control flag: if an external command or program is to be run synchronously. ²	
tracecntl		Control flag: if SAP tracing level 3 is activated for tracing SAPXPG, the program that starts an external command or program. ^{1 2}	
stdoutcntl		Control flag: indicates if standard output from an external command or program is to be written to the job log. ^{1 2}	
stderrcntl		Control flag: indicates if standard error from an external command or program is to be written to the job log. ^{1 2}	

**Note:**

1. This attribute is available for BC-XBP 2.0 and later.
2. This attribute is a flag, that is, it does not have a value, for example: `-s2 pr_release`.

IBM Workload Scheduler performs the following syntax validation on job attributes:

- Only valid attributes are allowed.
- Checks if a particular attribute requires a value.
- The values of the following attributes are checked:
 - type
 - language
 - pr_copies
 - pr_lines
 - pr_columns

Validation is performed before the job is created in the SAP system. If the validation fails, the IBM Workload Scheduler job goes into the ABEND state.

Specifying job parameters using variable substitution

Parameters can be provided at run time using the variable substitution feature. For example, the value appears as:

```
-s1 report=&VARNAME
```

The variable substitution process occurs while IBM Workload Scheduler is creating the symphony file.

Examples: Dynamically defining and updating SAP jobs

This section describes some usage examples of this feature:

Job definition and run scenario using the -flag type=exec parameter

The following example creates and runs a 3-step job. The first step runs the ABAP MYPROG1 using variant VAR01 and associated variant parameter. Step 2 has a step user defined. Step 3 uses the same ABAP as step 1 with no associated variant.

The only requirement is that the elements referred to are known in the SAP system (user, program). If the variant does not exist, there should be a set of values to define the content of the variant for its creation (pairs of `-vparN` `-vselN` parameters for the parameters and selections of the ABAP program).

```
-job TESTJOB01 -c A
-s1 type=A -s1 program=MYPROG1
-v1 VAR01 -vpar1 TESTNAME=TST
-s2 report=SP00LX1 -s2 user=PRTUSER
-s3 type=A -s3 program=MYPROG1 -flag type=exec
```

The job returns job ID 12345678

Job copy and overwrite the job created in the previous step

The following job statement references the job created in the previous example. A new copy of the job is made and the parameters specified in the invocation are used to update the definition. In this case the variant for step 1 is modified and a new external program step (Step 4) is added.

```
-job TESTJOB01 -i 12345678
-s1 variant=VAR01A
-vpar1 TESTNAME=TST2
-s4 type=X -s4 report=niping -s4 parameter=-t
-flag type=exec
```

Copy and overwrite a job referencing an existing job template

The following example shows a job creation referencing a job template (previously created without using this feature). A template job called `TEMPLAJOB` already exists on the SAP system with an ID of 56780123. It is a single ABAP step job to which we now add some print parameters.

```
-job TEMPLAJOB
-I 56780123 -s1 pr_immed
-flag type=exec
```

A temporary variant is created using the information indicated in the expression

The following is the syntax to be used:

```
-vpar1 <parameter_name>=<parameter_value> ...
-vsel1 <selection_option_name>
... -vtempl
```

The following example shows how you can submit a job that creates a temporary variant that is assigned to step number 1, and assigns a value to a variant parameter for step number 1:

```
-job TESTJOB01 -C A -flag type=exec -user R3USER
-s1 type=A -s1 program=MYPROG1
-vtempl -vpar1 TESTNAME=TST
```

The following example shows how you can submit a job that creates a temporary variant that is assigned to step number 1, assigns a value to a variant parameter for step number 1, and assigns a value to a variant selection option (date) for step number 1:

```
-job TESTJOB01 -C A -flag type=exec -user R3USER
-s1 type=A -s1 program=MYPROG1
-vtempl -vpar1 FILENAME=FLN
-vsel1 date=E#BT#20110101#20110412
```

Assign a temporary variant to the specified step number

The following is the syntax to be used:

```
-v1 <temporary_variant_name> -vtempl
```

The following is an example of how you can submit a job to assign a temporary variant, which has already been created (as in the previous example), and assign a value to step number 1:

```
-job TESTJOB01 -C A -flag type=exec -user R3USER
-s1 type=A -s1 program=MYPROG1
-vtempl -v1 &0000000000001
```

The value for a temporary variant that already exists is substituted with the value indicated in the expression

The following is the syntax to be used:

```
-v1 <temporary_variant_name> -vpar1 <parameter_name>=<parameter_value> ...
-vsel1 <selection_option_name> ... -vtempl
```

The following is an example of how you can submit a job that substitutes the value of a temporary variant, which must already exist, with a new value. The temporary variant must exist, otherwise, the expression returns an error.

```
-job TESTJOB01 -C A -flag type=exec -user R3USER
-s1 type=A -s1 program=MYPROG1
-vtempl -v1 &0000000000001 -vpar1 TESTNAME=TST2
```

Defining conditions and criteria

IBM Workload Scheduler accesses the Computer Center Management System (CCMS) Background Processing components of SAP systems through the BC-XBP interface to provide additional capabilities from the Dynamic Workload Console, one of those being the Criteria Manager.

IBM Workload Scheduler supports the BC-XBP 3.0 interface which provides functions to control R/3 batch jobs.

The Criteria Manager is a tool that enables you to define conditions and criteria that, when combined, form complex dependencies that you can use in the following contexts:

- Managing raised events in the SAP event history.
- Managing reorganization tasks against the SAP event history.
- Intercepting jobs.

If you have other types of criteria defined on your SAP system, then you can perform other actions in addition to those listed in this section.

The criteria profile

The Criteria Manager enables you to define a criteria profile which is a container for a combination of criteria. The criteria profile can be of various types and each criteria type has a standard set of selection criterion. For each criteria, you can specify a single value, a range of values by indicating a lower and upper limit, and multiple values. The following is the standard set of selection criterion for each criteria profile type. In addition to these, you can also see any other types of criteria profiles you have defined on your SAP system:

Event History

EVENTID

The identifier of the event defined in the SAP system.

EVENTPARM

The parameter of the event defined in the SAP system.

PARMID

The identifier of the parameter of the event defined in the SAP system.

Event History Reorg

Event State

The state of the event.

Event Timestamp

The timestamp for the event.

Interception

Job Name

A name identifying the job.

Job Class

The class assigned to the job that represents the priority with which the job runs in the SAP system.

The criteria hierarchy

You create and combine criteria in a criteria hierarchy. The criteria hierarchy is a set of all the criteria that must be fulfilled for a specific action to take place in the specific context. For example, you can define a criteria hierarchy to log all raised events in the SAP event history with an event name that begins with "CRITICAL_EVENT" and with an event argument equal to 150.

The criteria in the hierarchy is grouped in nodes and relationships between the nodes are determined by the logical operators AND or OR. You can nest nodes in other nodes.

To have the criteria profile begin processing, the criteria profile must be activated. Only one criteria profile of the same type can be active at one time.

Example

An example

See [Example: Defining which raised events to log on page 302](#) for an example that demonstrates how to build a criteria hierarchy to manage the logging of raised events in the SAP event history.

Example: Defining which raised events to log

The event history stores all events that are raised by the system. You can define specific criteria so that only raised events that match certain criteria are logged.

The event history enables IBM Workload Scheduler to consume events that are raised by the SAP system.

Checking the log of raised events gives you access to the following information:

- Verify that an event was raised in the system.
- Verify if the event was processed.

In the example that follows, an event history criteria profile is created that contains the definition of the criteria, the criteria hierarchy, that events must fulfill to be logged in the event history. The criteria profile must then be activated so that it can begin processing events according to the criteria.

The criteria profile, `Event profile 1`, contains a criteria hierarchy that logs only those events in the event history with event name that begins with `CRITICAL_EVENT` and event argument equal to "789".

Create the criteria profile

A criteria profile contains the definition of the criteria you want to set for logging raised events.

About this task

Create a criteria profile, `Event profile 1`, of type, Event History, to contain the criteria hierarchy.

1. In the Navigation bar at the top, click **Design > SAP > Manage SAP Criteria Profiles**.
2. In **Engine name**, select the name of the IBM Workload Scheduler engine connection from which you want to work with SAP jobs.
3. In **Workstation name**, type the name of the workstation where the SAP job runs. This is the workstation with the r3batch access method that communicates with the remote SAP system. If you do not know the name of the workstation, click the **Lookup Workstations** icon to enter your filter criteria and click **Search**. If you enter a string representing part of the workstation name, it must be followed by the asterisk (*) wildcard character. Both the question mark (?) and asterisk (*) are supported as wildcards. You can also simply use the asterisk wildcard character (*) to display all workstations. Optionally, specify any of the other search criteria available and click **Search**.

Result

From the results displayed, select the workstation and click **OK**.

4. In **Options file**, specify an options file that resides on the specified workstation. Each workstation can have one or more options files that can be used to customize the behavior of the r3batch access method, except for extended agent workstations, where only one options file can exist and therefore does not need to be specified. For the

workstation specified, enter the file name of the options file or click pick icon to search for options files that reside on the specified workstation and select one.

5. Click **Go**.
6. From the Criteria Manager main view, click **New** to create a criteria profile.
7. Select **Event History** as the type of criteria profile you want to create.
8. Enter descriptive text that enables you to easily identify the criteria profile in the table of criteria profiles. Type `Event profile 1`. Avoid using special characters such as, < (less than), > (greater than), or the ' (apostrophe) in this field.
9. Click **Save**.

Results

The criteria profile is displayed in the list of criteria profiles and it is not yet active.

What to do next

Next, begin building the criteria hierarchy. The criteria profile is the container for the criteria hierarchy.

Build the criteria hierarchy

The criteria hierarchy is stored in the criteria profile and is made up of criteria. A group of criteria is contained in a node.

Before you begin

The criteria hierarchy is made up of a combination of nodes and criteria. A node contains a group of criteria where the relationship between the criteria is determined by an AND or an OR relation. You can nest nodes in other nodes. By default, a top level AND node is created in the criteria hierarchy. You can create other AND or OR nodes nested in this parent node. You can also add one or more criterion to the nodes. Add an AND node when all of the criteria defined in the node must be fulfilled. Add an OR node when at least one of the criteria defined in the node must be fulfilled.

About this task

In this example, define a criterion that logs all events whose name begins with `CRITICAL_EVENT` and with event argument equal to 789.

1. Click  to create a new criterion in the default AND node.
2. In **Description**, type `Criterion 1`.
3. In **EVENTID**, click  to specify the value for the EVENTID field.
4. Leave the default value **Select** to indicate to use the selection criterion specified when processing events.
5. In **Options**, select **Pattern** and in **Single Value or Lower Limit**, type `CRITICAL_EVENT*`.

Result

This sets the condition for the event name.

6. In **EVENTPARM**, click  to specify the value for the EVENTPARM field.
7. Leave the default value **Select** to indicate to use the selection criterion specified when processing events.
8. In **Options**, select **Equal to** and in **Single Value or Lower Limit**, type `789`.

Result

This sets the condition for the event argument.

9. Click **Save** to save the criterion definition.

Results

The criteria profile now contains a criterion that specifies which raised events must be logged. You can continue to create another criteria in the same parent node or you can nest either an AND or an OR node in the parent node to determine the logical relation between the criteria that the nested node will contain. Add an AND node within which you can create one or more criteria where all the criteria specified in the node must be fulfilled, or add an OR node within which you can create one or more criteria where at least one of the criteria specified must be fulfilled.

What to do next

To apply this criteria profile so that it begins processing events according to the criteria defined, you must activate the criteria profile.

Activate the criteria profile

To apply the `Event profile 1` criteria profile so that it begins processing raised events according to the criteria specified in the criteria hierarchy, you must activate the criteria profile.

About this task

A criteria profile can either be active or not active. For a criteria profile to take effect, the profile must be activated. Only one criteria profile of the same type can be active at one time. Criteria profiles cannot be edited if they are in the active state. Follow the procedure to activate the `Event profile 1` criteria profile.

1. Select the `Event profile 1` criteria profile from the table of criteria profiles.
2. Select **Activate** from the toolbar. 

Results

The status of the criteria profile is updated to show that it is now active. The criteria profile can now begin to process raised events according to the specifications of the criteria hierarchy and log them to the event history. If another criteria profile of the same criteria type was active, its status changes to inactive.

Using the BDC Wait option

By using the Batch Data Collector (BDC) Wait option, you can specify that a SAP job launched by IBM Workload Scheduler is not to be considered complete until all of its BDC sessions have completed.

About this task

The Batch Data Collector (BDC) Wait option prevents other IBM Workload Scheduler jobs that are dependent on the SAP job from being launched until all of the related BDC sessions for the SAP job have ended.

To use the option, a SAP job must write informational messages in its job log. This can be done by modifying the SAP function module `BDC_OPEN_GROUP` as follows:


```

FUNCTION BDC_OPEN_GROUP.
...
CALL 'BDC_OPEN_GROUP' ID 'CLIENT'      FIELD CLIENT
                        ID 'GROUP'       FIELD GROUP
                        ID 'USER'        FIELD USER
                        ID 'KEEP'        FIELD KEEP
                        ID 'HOLDDATE'    FIELD HOLDDATE
                        ID 'DESTINATION' FIELD DEST
                        ID 'QID'         FIELD QID
                        ID 'RECORD'      FIELD RECORD
                        ID 'PROG'        FIELD PROG.

*
IF SY-SUBRC EQ 0.
  BQID  = QID.
  BUSER = SY-MSGV1.
  BGROUP = GROUP.
* CALL FUNCTION 'DB_COMMIT'.
  CALL FUNCTION 'ENQUEUE_BDC_QID'
    EXPORTING DATATYP = 'BDC '
    GROUPID   = BGROUP
    QID       = BQID
    EXCEPTIONS FOREIGN_LOCK = 98
    SYSTEM_FAILURE = 99.

  IF SY-SUBRC EQ 0.
    message i368(00) with 'BDCWAIT: ' qid.
  ENDIF.
ENDIF.
*
PERFORM FEHLER_BEHANDLUNG USING SY-SUBRC.
*
*
ENDFUNCTION.

```



Note: The actual parameters of the call of the C function `(CALL 'BDC_OPEN_GROUP' ID ...)` might vary depending on the SAP release. With this approach, you obtain a global change in your SAP system.

The completion status of a SAP job launched by IBM Workload Scheduler is based on the value you set for the **bdc_job_status_failed** option. By default, this option is set to `ignore`, meaning that the job is considered successfully completed when the BDC sessions are finished, regardless of their success or failure. For details about the **bdc_job_status_failed** option, refer to [Table 26: Task string parameters for SAP jobs on page 265](#).

Job interception and parent-child features

This section describes how the job interception and parent-child features of BC-XBP 2.0 and 3.0 are supported by IBM Workload Scheduler.



Note: The process of defining relaunch criteria and collecting and relaunching intercepted jobs is supported only in distributed environments and not in z/OS environments.

Implementing job interception

The high-level steps required to implement job interception.

About this task

Job interception is a feature of both the BC-XBP 2.0 and BC-XBP 3.0 interfaces. It enables IBM Workload Scheduler to have a very sophisticated control over the jobs launched by SAP users from the SAP graphical interface.

The job interception mechanism becomes active when the SAP job scheduler is about to start an SAP job (that is, when the start conditions of an SAP job are fulfilled). It checks the job parameters (`job_name`, `creator`, `client`) against the entries in the SAP table `TBCICPT1`, and when the job parameters match the criteria, the SAP job is set back to the `scheduled` status and is marked with a special flag, denoting that the job has been intercepted. The criteria defined in the criteria table establishes which job are intercepted.

If IBM Workload Scheduler has been set up to handle job interception, it periodically runs its own job to retrieve a list of intercepted jobs and reschedules them to be relaunched. This job can be referred to as the *interception collector* job.

Job interception with the BC-XBP 2.0 interface is based on the single extended agent workstation, whereas with the BC-XBP 3.0 interface, job interception is based on the currently active job interception criteria profile.



Note:

- Jobs launched by IBM Workload Scheduler, or by any other external scheduler using the BC-XBP interface, can be intercepted provided the **job_interceptable** option in the common options file is set to `ON`, and the **-flag enable_job_interceptable** keyword is included in the job definition.
- Ensure that the job interception and job throttling features are not running at the same time. The interception collector jobs fail if a job throttler instance is running. To stop the job throttler, refer to [Step 5. Starting and stopping the job throttling feature on page 336](#).

The following are the high-level steps required to implement job interception for both the BC-XBP 2.0 and 3.0 interfaces.

Job interception and the BC-XBP 2.0 interface

About this task

To set up IBM Workload Scheduler to handle job interception in an SAP environment with the BC-XBP 2.0 interface, implement the following steps:

1. Install the BC-XBP 2.0 interface. Refer to SAP Note 604496 to know if your SAP system already has the BC-XBP 2.0 interface, or which SAP support package you need to install to enable it.
2. Define an IBM Workload Scheduler job to periodically collect the intercepted SAP jobs.
3. Specify interception criteria in the SAP system.
4. Specify interception criteria in IBM Workload Scheduler from the Monitor Workstations portlet on the Dynamic Workload Console. The criteria is set at workstation level.
5. Activate the job interception feature of the BC-XBP 2.0 interface.

Job interception and the BC-XBP 3.0 interface

About this task

To set up IBM Workload Scheduler to handle job interception in an SAP environment with the BC-XBP 3.0 interface, implement the following steps:

1. Verify if the BC-XBP 3.0 interface is already installed on the SAP system.
2. Define an IBM Workload Scheduler job to periodically collect the intercepted SAP jobs.
3. Specify interception criteria in the SAP system.
4. Specify interception criteria in IBM Workload Scheduler from the Manage SAP Criteria Profiles portlet on the Dynamic Workload Console.
5. Activate the job interception feature of the BC-XBP 3.0 interface.

Collecting intercepted jobs periodically for BC-XBP 2.0

With the BC-XBP 2.0 interface, you can configure the job interception collector using an IBM Workload Scheduler job that periodically retrieves intercepted jobs and relaunches them.

About this task

Define an IBM Workload Scheduler job that uses the SAP interception collector task to collect intercepted jobs and restart them.

To define an IBM Workload Scheduler job that collects intercepted job and relaunches them, use the following syntax:

```
ENGINE_NAME_HOSTING_XA#[folder/]JOBNAME
SCRIPTNAME "TWA_home/methods/r3batch -t HIJ -c UNIQUE_ID"
DESCRIPTION "Collects intercepted jobs on SAP XA XA_Unique_ID"
STREAMLOGON TWSuser
RECOVERY STOP
```

Where:

ENGINE_NAME_HOSTING_XA

The name of the engine workstation hosting the XA workstation with the r3batch access method that communicates with the SAP system.

JOBNAME

Name of the IBM Workload Scheduler job and the folder in which it is defined, if any.

TWA_home

Fully qualified path to your IBM Workload Scheduler installation.

XA_Unique_ID

The unique identifier for the extended agent workstation. See [UNIQUE_ID on page 177](#) for more details about retrieving the unique identifier.

-t HIJ

This is the SAP task type to run the job interception collector. HIJ stands for `Handle Intercepted Jobs`.

TWSuser

Name of the IBM Workload Scheduler user that launches the access method.

The interception collector job runs at periodical intervals; for example, every 10 minutes. It retrieves all the jobs that have been intercepted since the last run of the interception collector, and launches them again according to a template.

Collecting intercepted jobs periodically for BC-XBP 3.0

With the BC-XBP 3.0 interface, you can configure the job interception collector using an IBM Workload Scheduler job that periodically retrieves intercepted jobs and relaunches them.

About this task

Because intercepted jobs remain in the `Released` and then `Intercepted` status until they are relaunched, you need to use the SAP interception collector task to collect and relaunch them.

To use the job interception collector BC-XBP 3.0 to collect and restart jobs, create the **r3batch_icp** folder in the *TWS_home*/methods/ path with the correct access rights.

To define an IBM Workload Scheduler job that collects and relaunches jobs use the following syntax:

```
ENGINE_NAME_HOSTING_XA#[folder/]JOBNAME
DOCOMMAND "TWA_home/methods/r3batch -t HIJ -c UNIQUE_ID -- \
  "-profile_id profile_ID_number\"
STREAMLOGON TWSuser
DESCRIPTION "Collects intercepted jobs on SAP XA XA_Unique_ID"
TASKTYPE UNIX
RECOVERY STOP
```

where,

ENGINE_NAME_HOSTING_XA

The name of the engine workstation hosting the XA workstation with the r3batch access method that communicates with the SAP system.

JOBNAME

Name of the IBM Workload Scheduler job and the folder within which it is defined, if any.

TWA_home

Fully qualified path to your IBM Workload Scheduler installation.

XA_Unique_ID

The unique identifier for the extended agent workstation. See [UNIQUE_ID on page 177](#) for more details about retrieving the unique identifier.

-t HIJ

This is the SAP task type to run the job interception collector. HIJ stands for `Handle Intercepted Jobs`.

- profile_id profile_ID_number

Specifies the identification number of the interception criteria profile on the SAP system for XBP 3.0.

TWSuser

Name of the IBM Workload Scheduler user that launches the access method.

The interception collector job runs at periodical intervals; for example, every 10 minutes. It retrieves all the jobs that have been intercepted since the last run of the interception collector, and launches them again according to a template.



Note: If the interception collector is configured for XBP 3.0 job interception, but the XBP 2.0 interface is configured on the SAP system, the collector fails. Ensure the XBP interface versions are synchronized.

Setting interception criteria on the SAP system

About this task

In SAP, the interception criteria are held in table `TBCICPT1`. Only jobs that match the criteria of this table are intercepted, when their `start` conditions are fulfilled. All the other jobs are run normally.

You can maintain the entries in this table by using transaction `se16` and setting the following:

- Client number
- Job mask
- User mask

Setting interception criteria on IBM Workload Scheduler

About this task

In IBM Workload Scheduler, interception criteria are defined and used by setting:

Table criteria**For BC-XBP 2.0**

You use the **Monitor Workload** of Dynamic Workload Console to set table criteria.

For details about how you set table criteria, see [Setting SAP table criteria on the extended agent workstation on page 310](#).

For BC-XBP 3.0

You set table criteria from the **Administration > Workload Design > Manage SAP Criteria Profiles** panel from the Dynamic Workload Console.

For details about how you set table criteria, see [Setting SAP criteria in the job interception criteria profile on page 311](#).

Template files (optional)

For details about how you create template files, see [Using template files on page 313](#).

Setting SAP table criteria on the extended agent workstation

About this task

To set table criteria with the BC-XBP 2.0 interface on an SAP job using the **Monitor Workload** of the Dynamic Workload Console, follow these steps:

1. Log in to the Dynamic Workload Console.
2. From the **Monitoring and Reporting** menu, click **Orchestration Monitor**.
3. In the **Monitor Workload** window select the engine, enter **Workstation** in the **Object Type** field, and select the plan to display the list of workstations you want to monitor. Click **Run**.
4. Select an extended agent workstation in the table of displayed workstations, and click **More Actions > Table Criteria...** from the toolbar.
5. The **Table Criteria** panel displays. From this panel you can add, delete, edit, or refresh criteria.

Figure 21. The Table Criteria panel

	Client	Job Name	Job Creator	Job Template
☐	001	ICP_TEST*	ALTRICHTER	
☐	500	0	0	
☐	888	INVOICE*	ALTRICHTER	

6. Specify the criteria:

- In **Client**, specify the client workstation of the SAP job.
- In **Job Name**, specify a filter to match a set of SAP jobs. Use the asterisk (*) wildcard character to match a set of jobs.
- In **Job Creator**, specify a filter to match a set of SAP job creator. Use the asterisk (*) wildcard character to match a set of jobs.
- Optionally, in **Job Template**, specify the template file that contains instructions for the interception collector about how to run the intercepted SAP job under control of IBM Workload Scheduler. For more information about template files, see [Using template files on page 313](#).
- In **Job Class**, specify the class assigned to the job that represents the priority with which the job runs on the SAP system.

7. Click **OK**.

Setting SAP criteria in the job interception criteria profile

Setting criteria to intercept jobs and relaunch them.

About this task

To set the criteria that defines which SAP jobs to intercept and relaunch with the BC-XBP 3.0 interface using the Dynamic Workload Console, perform the following steps:

1. In the Navigation bar at the top, click **Design > SAP > Manage SAP Criteria Profiles**.
2. In **Workstation name**, type the name of the workstation where the SAP job runs. This is the workstation with the r3batch access method that communicates with the remote SAP system. If you do not know the name of the workstation, click the **Lookup Workstations** icon to enter your filter criteria and click **Search**. If you enter a string representing part of the workstation name, it must be followed by the asterisk (*) wildcard character. Both the question mark (?) and asterisk (*) are supported as wildcards. You can also simply use the asterisk wildcard character (*) to display all workstations. Optionally, specify any of the other search criteria available and click **Search**.

Result

From the results displayed, select the workstation and click **OK**.

3. From the Criteria Manager main view, click **New** to create a criteria profile.
4. Select **Interception** as the type of criteria profile you want to create.
5. Enter descriptive text that enables you to easily identify the criteria profile in the table of criteria profiles. Avoid using special characters such as, <(less than), > (greater than), or the ' (apostrophe) in this field.
6. Click **Save**.

Result

The criteria profile is displayed in the list of criteria profiles and it is not yet active.

7. On the **Details** tab in the upper-right pane, define the criteria that intercepted jobs must match. For example, to intercept jobs with a job name beginning with "ICP", specify the following criteria:

- a. Click  to define a new criterion.
- b. In **Description**, type `Criterion 1`.
- c. In **JOB NAME**, click  to specify the value for the JOB NAME field.
- d. Leave the default value **Select** to indicate to use the selection criterion specified when intercepting jobs.
- e. In **Options**, select **Pattern** and in **Single Value or Lower Limit**, type `ICP*`.

Result

This sets the condition for the job name.

- f. Click **Save** to save the criterion definition.
8. Define the criteria that must be matched to relaunch intercepted jobs. Click the **Job Relaunch Criteria** tab.
 - a. Click  to define a new criteria that determines which jobs are relaunched.
 - b. In **Client**, specify the client workstation of the SAP job.
 - c. In **Job Name**, specify a filter to match a set of SAP jobs. Use the asterisk (*) wildcard character to match a set of jobs.
 - d. In **Job Creator**, specify a filter to match a set of SAP job creator. Use the asterisk (*) wildcard character to match a set of jobs.

- e. Optionally, in **Job Template**, specify the template file that contains instructions for the interception collector about how to run the intercepted SAP job under control of IBM Workload Scheduler. For more information about template files, see [Using template files on page 313](#).
 - f. In **Job Class**, specify the class assigned to the job that represents the priority with which the job runs on the SAP system.
9. Click **OK**.
 10. You can continue to define more criteria and then save the criteria profile.
 11. When you are done defining the criteria, save the criteria profile.
 12. Select the criteria profile and then click **Activate** from the toolbar. 

Results

The status of the criteria profile is updated to show that it is now active. The criteria profile can now begin to intercept jobs according to the specifications of the criteria hierarchy and relaunch them as defined in the IBM Workload Scheduler job. If another criteria profile of the same criteria type was active, its status changes to inactive.

Using template files

About this task

A template is a file with extension `.jdf` located in the same directory as the interception criteria file (`TWA_DATA_DIR/methods/r3batch_icp`). The template file contains instructions for the interception collector about how to run the intercepted SAP job under control of IBM Workload Scheduler. Its syntax corresponds to the syntax of `docommand` in `conman`. You can use any text editor to maintain this file. Ensure that the user, `LJUser`, is able to read and write to this file.

If the user template file is empty, a template file named `default.jdf` is used. If `default.jdf` does not exist, the following instructions are used:

```
alias=SAP_$RUN_$JOBNAME_$JOBCOUNT
```

This means that the intercepted SAP jobs are to be restarted immediately, because of the absence of the `at=` job option. Their IBM Workload Scheduler names are composed of the string `SAP_`, the current run number of the interception collector, and the name and ID of the SAP job.

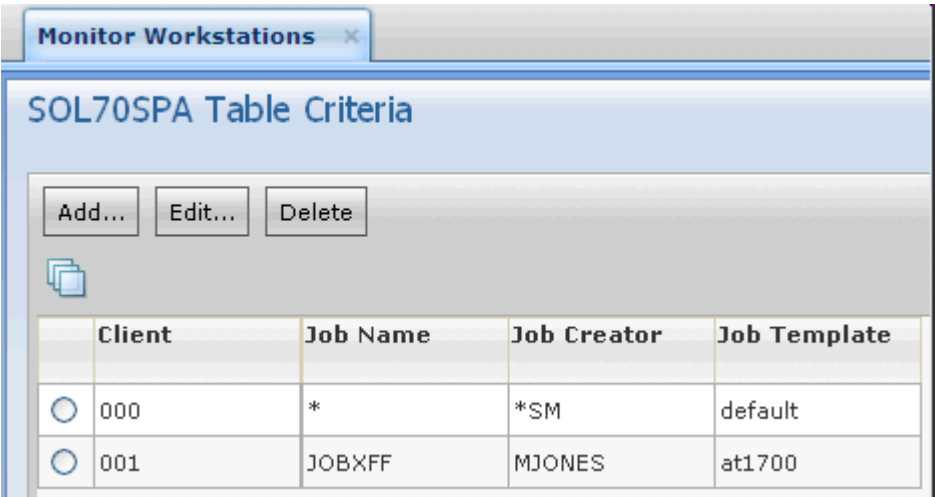
The instruction set for restarting an intercepted SAP job is retrieved in the following order:

1. From the template file, if an existing template is specified in the interception criteria file.
2. From the default template file, if the template is specified in the interception criteria file but does not exist, or if the template is not specified in the interception criteria file.
3. From the default instruction set, if the default template file does not exist.

Job interception example

The following example demonstrates how different template files can be used to determine when an intercepted SAP job is restarted. The interception criteria table contains the following entries:

Figure 22. The Table Criteria panel



The table criteria specified, implies the following:

Client 000

All jobs started in client 000 by SAP users whose user name begins with sm, will be intercepted. The interception collector restarts the jobs using the instructions from the default template file default.jdf. If the default template file does not exist, then the SAP jobs are restarted immediately as specified in the default instruction set:

```
alias=SAP_$RUN_$JOBNAME_$JOBCOUNT
```

Client 001

The job named, JOBXFF, started in client 001 by SAP user named, MJONES, will be intercepted. The interception collector restarts the jobs using the instructions from the template file at1700.jdf. The SAP jobs are restarted at 17:00 with a random name, because of the alias command. The template file at1700.jdf contains the following entry:

```
alias;at=1700
```

Using placeholders

In the template files you can use a number of placeholders that are replaced by the interception collector at run time. They are listed in [Table 31: Placeholders for job interception template files on page 314](#).

Table 31. Placeholders for job interception template files

Placeholder	Description
\$CPU	Name of the extended agent workstation where the interception collector runs.
\$CLIENT	Client number of the intercepted SAP job.
\$JOBNAME	Name of the intercepted SAP job.
\$JOBCOUNT	Job ID of the intercepted SAP job.

Table 31. Placeholders for job interception template files (continued)

Placeholder	Description
\$USER	Name of the user who launched the SAP job.
\$JOBNUM	Job number of the interception collector.
\$RUN	Current run number of the interception collector.
\$SCHED	Schedule name of the interception collector.
\$RAND	Random number.

The template:

```
alias=ICP_$RAND_$JOBNAME_$JOBCOUNT_$CLIENT;at=1000
```

instructs the interception collector to restart the SAP job named `DEMO_JOB` with job ID `12345678` on client `100` at `10:00` as IBM Workload Scheduler job `ICP_1432_DEMO_JOB_12345678_100`.

Activating the job interception feature

Activate the job interception feature for the appropriate BC-XBP interface.

About this task

To enable the job interception feature: .

1. Run ABAP report `INITXBP2`.

Result

This report shows you the current status of the job interception and parent-child features, and allows you to toggle the status of both features.

2. Select the BC-XBP interface version as appropriate:
 - **Activate 3.0**
 - **Activate 2.0**
3. Save the changes.

The parent-child feature

In some situations, an SAP job dynamically spawns a number of other jobs; for example, to distribute the workload to the free application servers. Prominent examples are the mass activity jobs of the SAP `FI-CA` component. Before BC-XBP 2.0, it was difficult for external schedulers to handle this situation, because the business process does not usually end with the end of the initial job (parent job), but with the end of all subjobs (child jobs).

The BC-XBP 2.0 interface allows you to determine if a job has launched subjobs, together with their names and IDs, and so it is now possible to track them.

To activate this feature, use the `INITXBP2` ABAP report, which you can also use to toggle the status of job interception.

When the parent-child feature is active, IBM Workload Scheduler considers an SAP job as finished only after all its child jobs have ended. The status of the IBM Workload Scheduler job remains as `EXEC` while the parent job or any of its child jobs are running.

The status of the IBM Workload Scheduler job becomes `succ` if the parent job and all child jobs end successfully. If any of the jobs ended with an error, the status of the IBM Workload Scheduler job becomes `ABEND`.



Note: The parent-child feature can interfere with job interception because, although the parent job cannot be intercepted, any of its child jobs can be intercepted if they match the interception criteria. In this case, the IBM Workload Scheduler job remains in the `EXEC` status until the intercepted child job has been relaunched and has ended.

The joblogs of the child jobs are appended in the IBM Workload Scheduler `stdlist` after the joblog of the parent job.

Using Business Information Warehouse

Business Information Warehouse (BIW) is a data warehouse solution tailored to SAP.

Business Information Warehouse (BIW) allows business reporting and decision support.

To use the InfoPackages component, you must have the SAP Business Warehouse Systems, version 2.0B or later installed.

To use the Process Chains component, you must have the SAP Business Warehouse Systems, version 3.0B or later installed.

The Support Package 9 (SAPKW31009) for SAP Business Warehouse version 3.1 is required so that `SAP` can launch process chains.

Business Warehouse components

SAP supports two main Business Warehouse components, InfoPackages and Process Chains.

An InfoPackage is the entry point for the loading process from a specific **InfoSource** (a logical container of data source, generically named **InfoObject**). Technically, an InfoPackage is an SAP job whose aim is to load data. Like any other SAP job, it contains job-specific parameters such as start time, and dependencies.

A Process Chain is a complex chain of different processes and their relationships. The processes within a process chain are not limited to data load processes, or InfoPackages, but also include:

- Attribute/Hierarchy Change run
- Aggregate rollup
- ABAP program
- Another process chain
- Customer build process

Defining user authorizations to manage SAP Business Warehouse InfoPackages and process chains

What you need to use SAP R/3 Business Warehouse InfoPackages and process chains.

Access method for SAP can manage SAP Business Warehouse InfoPackages and process chains. To use the SAP Business Warehouse functions, you must define an IBM Workload Scheduler user within SAP with full authorization for the ABAP Workbench object S_DEVELOP.

The user must also belong to the following profiles:

- S_BI-WHM_RFC (for Business Information Warehouse version 7.0, or later)
- S_RS_ALL
- Z_MAESTRO

Managing SAP Business Warehouse InfoPackages and process chains

You can manage existing InfoPackages and process chains on SAP systems from SAP.

Business Warehouse InfoPackages and process chains can only be created from the SAP environment. However, the Dynamic Workload Console supports pick lists of InfoPackages and process chains, so that you can also define IBM Workload Scheduler jobs for these existing objects.

You can create IBM Workload Scheduler job definitions that map to SAP jobs that already exist on SAP systems in the following environments:

- Distributed
- z/OS

The SAP jobs can run on extended agent workstations, dynamic agent workstations, dynamic pools, and z-centric workstations depending on the type of job definition you choose to create.

This section describes how to perform tasks such as creating the IBM Workload Scheduler job definitions that map to SAP jobs, how to display the details of these jobs, and how to rerun a process chain job.

Creating an IBM Workload Scheduler job that contains InfoPackages or process chains

Creating a job with InfoPackages or process chains.

About this task

This section describes how to create an IBM Workload Scheduler SAP job definition that references a Business Warehouse InfoPackage or Process Chain SAP job.

SAP job definitions can be created using both a distributed or z/OS engine and they can be scheduled to run on the following workstations with the r3batch access method:

- An IBM Workload Scheduler extended agent workstation. A workstation that is hosted by a fault-tolerant agent or master workstation.
- A dynamic agent workstation.
- A dynamic pool.
- A z-centric workstation.

Refer to the Dynamic Workload Console online help for a complete description of all UI elements for both engine types and all supported workstation types.

Take into consideration that:

- To be able to schedule InfoPackages using IBM Workload Scheduler, the scheduling options of the InfoPackage must have:
 - Start type set to **Start later in background process**.
 - Start time set to **Immediate**.
- To be able to control process chains using IBM Workload Scheduler, the scheduling options of the process chain must be **Start Using Meta Chain or API**. If the process chain is set to **Direct Scheduling**, it starts immediately when activated in the SAP system or transported to another SAP system.
- If you are using an operating system that does **not** support Unicode, set the **TWSXA_LANG** option. For details about the operating systems that support Unicode, see [Unicode support on SAP on page 228](#). For details about the **TWSXA_LANG** option, see [Setting National Language support options on page 374](#).

You can create a SAP job definition to reference an InfoPackage or process chain using the Dynamic Workload Console.

The following procedure creates an IBM Workload Scheduler SAP job definition and references an InfoPackage or process chain in the IBM Workload Scheduler database:

1. Click **IBM Workload Scheduler>Design>Workload Designer**.
2. Select a an engine. The **Workload Designer** is displayed.
3. From the Working List pane, click:
 - z/OS engine: **New > ERP**
 - Distributed engine: **New > Job Definition > ERP**
4. Select the SAP job definition in accordance with the engine and type of agent on which the job runs.

z/OS engine

SAP

This job definition references an existing job on the SAP system and can run on dynamic agent workstations, dynamic pools, and z-centric workstations.

Distributed engine

SAP Job on Dynamic Workstations

This job definition can run on dynamic agent workstations, dynamic pools, and z-centric workstations.

SAP Job on XA Workstations

This job definition can run on extended agent workstations. A workstation that is hosted by a fault-tolerant agent or master workstation.

5. In the Workspace pane, specify the properties for the job definition you are creating using the tabs available. The tabs for each type of SAP job definition are similar, but there are some differences depending on the type of engine you

selected and the type of workstation on which the job runs. For more detailed information about the UI elements on each tab, see the Dynamic Workload Console online help.

The **General** page requires information regarding the workstation that connects to the remote SAP system. If a default SAP connection is already configured, then these fields are already prefilled, otherwise, you can specify the required information on the General page or you can configure a default connection to be used each time it is required in a definition, see [Setting the SAP data connection on page 256](#) for more information.

On the **Task** page, in **Subtype**, specify either **BW Process Chain** or **BW InfoPackage**.

6. Click **Save** to add the SAP job definition to the IBM Workload Scheduler database.

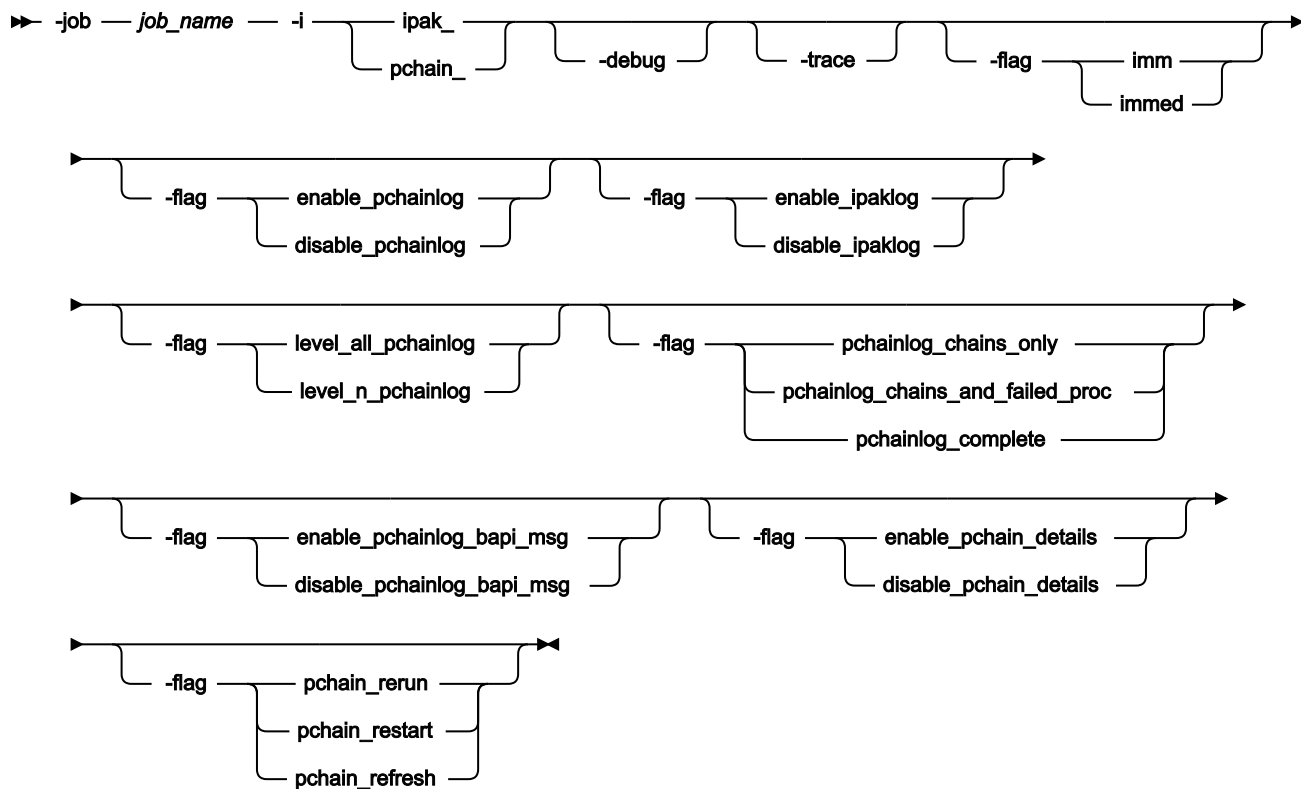
Task string to define Business Warehouse InfoPackages and process chain jobs

This section describes the task string parameters that control the running of the Business Warehouse InfoPackages and process chain jobs. You must specify them in the following places when you define their associated IBM Workload Scheduler jobs:

- If you use the Dynamic Workload Console, in the **SAP command line** field of the Task page of the SAP job definition panel.
- As arguments of the `scriptname` keyword in the job definition statement, if you use the IBM Workload Scheduler command line.
- As arguments of the `JOBCMD` keyword in the `JOBREC` statement in the `SCRIPTLIB` of IBM Z Workload Scheduler, if you are scheduling in an end-to-end environment.

The string syntax is the following:

Figure 23. Job definition syntax



The parameters are described in [Table 26: Task string parameters for SAP jobs on page 265](#).

Table 32. Task string parameters for SAP jobs

Parameter	Description	GUI Support
-job <i>job_name</i>	The name of the task to be run. It is either an InfoPackage technical field name, or a process chain name. This parameter is mandatory.	✓
-i {ipak_ pchain_}	One of the following: ipak_ Target job is an InfoPackage pchain_ Target job is a process chain	✓
-debug	Turns on the most verbose <code>r3batch</code> trace. This option is for debugging the extended agent and should not be used in standard production.	✓
-trace	Turns on the <code>SAP RFC</code> trace.	✓

Table 32. Task string parameters for SAP jobs (continued)

Parameter	Description	GUI Support
	When you use this option, a trace file is created in the IBM Workload Scheduler <code>methods</code> directory. In UNIX, this trace file is called <code>dev_rfc</code>. In Windows, the file is called <code>rfcXXXXX_XXXX.trc</code>. The <code>methods</code> directory is located in: On UNIX operating systems <code>TWA_DATA_DIR/methods</code> On Windows operating systems <code>TWA_home\methods</code> This option is for debugging the extended agent and should not be used in standard production. Ensure that you delete the trace option from the job after you have performed debug procedures. The trace file can become very large and unmanageable.	
-flag {imm immed}	Specifies to launch the job immediately, meaning that if there are no spare work processes, the job fails.	✓
-flag {enable_pchainlog disable_pchainlog}	Enables or disables retrieval and appending of the process chain job log in the IBM Workload Scheduler <code>stdlist</code> . Disable if the size of the log affects performance. A related configuration option can be set for this purpose at a more general level. See Table 24: r3batch common configuration options on page 237 .	✓
-flag {enable_ipaklog disable_ipaklog}	Enables or disables retrieval and appending of the InfoPackage job log in the IBM Workload Scheduler <code>stdlist</code> . Disable if the size of the log affects performance. A related configuration option can be set for this purpose at a more general level. See Table 24: r3batch common configuration options on page 237 .	
-flag {level_n_pchainlog level_all_pchainlog}	Allows for retrieval of process chain logs down to the process chain level you specify. level_n_pchainlog Specifies that the process chains are logged down to, and including, the level represented by number <i>n</i>.	

Table 32. Task string parameters for SAP jobs (continued)

Parameter	Description	GUI Support
	level_all_pchainlog Specifies that all the process chains are logged. The default is <code>level_1_pchainlog</code> . A related configuration option can be set for this purpose at a more general level. See Table 24: r3batch common configuration options on page 237 .	
-flag {pchainlog_chains_only pchainlog_chains_and_failed_proc pchainlog_complete}	Specifies what type of process chain-related logs will be retrieved. pchainlog_chains_only Only the process chains are logged. pchainlog_chains_and_failed_proc In addition to the process chains, all the processes that failed are also logged. pchainlog_complete The process chains and all processes are logged. The default is pchainlog_complete . A related configuration option can be set for this purpose at a more general level. See Table 24: r3batch common configuration options on page 237 .	
-flag {enable_pchainlog_bapi_msg disable_pchainlog_bapi_msg}	Enables or disables retrieval of additional messages from the BAPI calls from the SAP Business Warehouse process chains and appends them to the IBM Workload Scheduler <code>stdlist</code> .	
-flag {enable_pchain_details disable_pchain_details}	Enables or disables the display of details about the process chain job. A related configuration option can be set for this purpose at a more general level. See Table 24: r3batch common configuration options on page 237 .	✓
-flag {pchain_rerun pchain_restart pchain_refresh}	Determines the action that IBM Workload Scheduler performs when you rerun a job that submits a process chain.	✓

Table 32. Task string parameters for SAP jobs (continued)

Parameter	Description	GUI Support
	pchain_rerun	
	IBM Workload Scheduler creates another process chain instance and submits it to be run again.	
	pchain_restart	
	IBM Workload Scheduler restarts the original process chain from the failing processes to the end.	
	pchain_refresh	
	IBM Workload Scheduler updates the status and details of the original process chain.	
	For more details about rerunning a process chain, refer to Rerunning a process chain job on page 326 .	



Note: Typically, the `-debug` and `-trace` options are for debugging the extended agent and should not be used in standard production.

The following is an example for an InfoPackage job whose technical field name is `ZPAK_3LZ3JRF29AJDQM65ZJBJF50MY`:

```
-job ZPAK_3LZ3JRF29AJDQM65ZJBJF50MY -i ipak_
```

Displaying details about Business Warehouse InfoPackages

About this task

To display details about a Business Warehouse InfoPackage, perform the following steps:

1. From the **Design** menu, click **Manage Jobs on SAP**.
2. In **Engine name**, select the name of the IBM Workload Scheduler engine connection from which you want to view SAP job details.
3. In **SAP Job Type**, select **Business Warehouse InfoPackage**.
4. In **Workstation name**, specify the workstation where the SAP job runs. If you do not know the object name, click the ... (Browse) button. In the **Name** and **Location** panel, enter some characters of the object name (asterisk is supported as a wildcard) and click **Start**. From the displayed list, select the workstation you want to use, and click **OK**.
5. Click **Display**. The list of available jobs of type Business Warehouse InfoPackage for the specified engine is displayed.

6. Select the job for which you want to display the details and click **Details**.
7. When you have finished viewing the details for the job, click **OK** to return to the list of SAP jobs on the workstation specified.

Displaying details about a process chain job

You can view the details for a process chain job including any local subchains contained in the process chain.

Before you begin

Ensure you have performed the following steps before running this procedure:

- Set the **pchain_details** option to **on** in the common options file. For more information about this option, refer to [Defining the common options on page 237](#).
- In a distributed environment, customize the Browse Jobs tasks that you created *before* installing IBM Workload Scheduler 8.4 Fix Pack 1 to show the **Job Type** column. For details about how to customize the task properties, refer to the Dynamic Workload Console online help.
- In a z/OS environment, you must customize the task properties to display the **Advanced Job Type** column that indicates the job type. For details about how to customize the task properties, refer to the Dynamic Workload Console online help.

About this task

To display details about an SAP Process Chain that you scheduled as an IBM Workload Scheduler job, perform the following steps from the Dynamic Workload Console.

1. Click **Monitoring and Reporting > Workload Monitoring > Monitor Workload**.
2. In **Engine name**, select the name of the IBM Workload Scheduler engine connection from where you want to work with SAP jobs.
3. In **Object Type**, leave the default selection **Job**.
4. Click **Run**.
5. The table of results corresponding to the search criteria is displayed:

Figure 24. Dynamic Workload Console - Table of results

Status	Internal Status	Job	Job Type	Workstation (Job)	Job Stream
Successful	SUCC	PC_-1967534151	SAP Process Chain	W6470AGE	JOBS
Successful	SUCC	PC_301475144	SAP Process Chain	W6470AGE	JOBS
Successful	SUCC	PC_-109893088	SAP Process Chain	W6470AGE	JOBS
Error	ABEND	PC_-1347740193	SAP Process Chain	W6470AGE	JOBS
Error	ABEND	PC_646387164	SAP Process Chain	W6470AGE	JOBS
Error	ABEND	PC_-1513805554	SAP Process Chain	W6470AGE	JOBS
Error	ABEND	PC_1682550152	SAP Process Chain	W6470AGE	JOBS
Error	ABEND	PC_-573813697	SAP Process Chain	W6470AGE	JOBS
Successful	SUCC	PC_2145163497	SAP Process Chain	W6470AGE	JOBS
Successful	SUCC	PC_-1340397696	SAP Process Chain	W6470AGE	JOBS
Successful	SUCC	PC_-355196273	SAP Process Chain	W6470AGE	JOBS
Successful	SUCC	PC_1301210520	SAP Process Chain	W6470AGE	JOBS

6. Select a process chain job. For each process chain job, a hyperlink named **SAP Process Chain** is displayed.

Distributed environment

The **Job Type** column displays **SAP Process Chain** to help you identify SAP process chain jobs.

z/OS environment

The **Advanced Job Type** column displays **SAP Process Chain** to help you identify SAP process chain jobs.

Click the hyperlink for the job whose details you want to display.

7. The details for the process chain are displayed:

Figure 25. Dynamic Workload Console - Details of a process chain job

'PCHAINTEST01' SAP Process Chain

Monitor Jobs > Active Tasks (1) > All Jobs in plan (Distributed) (Owner... > 'PCHAINTEST01' SAP Process Chain

Refresh Close View Expand All Collapse All

'PCHAINTEST01' SAP Process Chain

Process	Type	Link	Status	Start Time	End Time	Description	ID	Process ID
▼ STARTER_PCHAIN_01	Start		Completed	2/13/06 1:15 PM CET	2/13/06 1:15 PM CET	STRATER PCHAIN 01	CXX9B4FFSGF7AH8EUL7FH30GQ0	0
NODELEV2	Program	Green	Completed	2/13/06 1:15 PM CET	2/13/06 1:15 PM CET	NODE_02_LEV_01	BI_PROCESS_ABAP_13154100	1
NODELEVE1	Program	Green	Completed	2/13/06 1:15 PM CET	2/13/06 1:15 PM CET	NODE_01_LEV_01	BI_PROCESS_ABAP_13154200	2

IBM Workload Scheduler monitors the process chain job until the job completes. The details shown reflect the last monitoring process performed. Perform a restart of the process chain indicating a refresh operation to synchronize the details with those on the remote SAP system to have the most updated information possible. If the process chain contains local subchains, a hyperlink is displayed for each one. Click the hyperlink you want, to display details about the corresponding subchain job. Alternatively, you can display the process chain details by clicking the hyperlink for the job and display the job properties panel. Click the hyperlink shown under **SAP Job Details**. The details for the process chain are displayed.

Rerunning a process chain job

Process chain jobs can be rerun from the start, rerunning the entire process chain, or they can be restarted from a specific process. Restarting a process enables you to restart without rerunning the whole process chain again. You can choose to either restart from the failed processes in a process chain, or restart a specific process indicating the related process ID.

To rerun an SAP job that submits a process chain, you can use one of the following user interfaces:

conman

For details, refer to the *IBM Workload Scheduler User's Guide and Reference*.

Dynamic Workload Console

See [Procedure for rerunning a process chain job on page 330](#) for information about performing this task from the console.

For information about rerunning an SAP Standard R/3 job, see [Rerunning a standard SAP job on page 280](#).

In general, when you rerun a process chain job, the new job is assigned the name of the alias you specify. To keep the original job name, set the IBM Workload Scheduler global option **enRetainNameOnRerunFrom** to **yes**. For details about this option, see *IBM Workload Scheduler Administration Guide*.

On extended agents, an alias is mandatory for each action you perform on the process chain job and the action itself, is the prefix of the alias name. For example, if you choose to restart a process chain from the failed processes, and assign PCHAIN1 as the alias for the process chain job, then the new job name is `Restart_PCHAIN1`.

In a z/OS environment, the process chain job maintains the same name and the Monitor Jobs view always displays the status for the last action performed on the job. Every time a rerun is performed on a process chain job, a new instance is generated each with a different ID.

 Note:

1. By default, if you do not specify any setting, rerunning a process chain job corresponds to submitting a new process chain instance.
2. If you kill an IBM Workload Scheduler job that submits a process chain, the process chain is removed from schedule in the SAP Business Information Warehouse system. To restart the same process chain instance with `r3batch`, you require at least the following SAP Business Information Warehouse versions:
 - 3.0 with SP25
 - 3.1 with SP19
 - 3.5 with SP10
 - 7.0

If your version of SAP Business Information Warehouse is earlier, you can restart the process chain only manually, through the SAP graphical interface.

[Table 33: Actions performed when you rerun a process chain job on page 327](#) shows the action performed when you rerun an IBM Workload Scheduler job that submits a process chain, depending on the settings you specify. These are the actions performed when you submit the rerun operation using the Rerun button from the Monitor Jobs view.

Table 33. Actions performed when you rerun a process chain job

Action performed	Description and setting
A new process chain instance is submitted	IBM Workload Scheduler creates another process chain instance and submits it to be run again. This action occurs when: • On extended agents, you specify <code>RERUNvalue</code> as the step to rerun, where <code>value</code> is any value you want. This setting overrides the settings in the job definition and options file, if any. In an end-to-end environment, you can perform this action on a centralized job by adding the following parameter to the script file: <pre>-flag pchain_rerun</pre>

Table 33. Actions performed when you rerun a process chain job (continued)

Action performed	Description and setting
	- In the job definition, you set <code>-flag pchain_rerun</code>. This setting overrides the setting in the options file, if any. For a description of this parameter, see Table 32: Task string parameters for SAP jobs on page 320. - In the options file, you set the <code>pchain_recover</code> option to <code>rerun</code>. For a description of this option, refer to Table 24: r3batch common configuration options on page 237.
The original process chain is rerun from the failed processes	IBM Workload Scheduler restarts the original process chain from the failed processes to the end. In this way, after you detected the error that caused the failure and performed the recovery action, you can rerun the process chain job from the failed processes and have its run completed. This action is performed only if at least one process in the process chain did not complete successfully. It occurs when: - On extended agents, you specify <code>RESTARTvalue</code> as the step to rerun, where <code>value</code> is any value you want. This setting overrides the settings in the job definition and options file, if any. In an end-to-end environment, you can perform this action on a centralized job by adding the following parameter to the script file: <pre>-flag pchain_restart</pre> - In the job definition, you set <code>-flag pchain_restart</code>. This setting overrides the setting in the options file, if any. For a description of this parameter, see Table 32: Task string parameters for SAP jobs on page 320. - In the options file, you set the <code>pchain_recover</code> option to <code>restart</code>. For a description of this option, refer to Table 24: r3batch common configuration options on page 237.
The process that you specify is restarted	IBM Workload Scheduler restarts the process of the original process chain that you specify, and monitors the process chain run until its final state. On extended agents, this action occurs when you specify <code>PROCESSprocessID</code> as the step to rerun, where <code>processID</code> is the identifier of the process you want. For example, if the process ID is <code>3</code>, you must specify <code>PROCESS3</code> as the step. You can view the process IDs in the following ways: Dynamic Workload Console, version 8.5 or later. From the panel where the details about the process chain are displayed, see the column named Process ID. For details about how to display the process chain details, refer to Displaying details about a process chain job on page 324.

Table 33. Actions performed when you rerun a process chain job (continued)

Action performed	Description and setting
	IBM Workload Scheduler job log, as follows: <pre>+++ EEW01071I Start of process chain PCHAIN1 Process Chain PCHAIN1 (Log ID:D3C0ZWAYESD58PX0YPE0GNZK7) . ----- Process Type: TRIGGER. Process Variant: PCHAIN1_STARTER. Actual State: F. ... >> Process ID: 3. Process Type: ABAP. Process Variant: Z_PCHAIN1_NODE3. Actual State: F. Instance: D3C0ZXL3IJ8LR509Q1D9A4Y4N. >> Process ID: 4. Process Type: ABAP. Process Variant: Z_PCHAIN1_NODE1. Actual State: . Instance: D3C0ZZKS0RR88DKRJQ09Z1WW7. +++ EEW01072I End of process chain PCHAIN1</pre>

The following list shows the meaning of the alphabetic value used as the actual state in the job log:

Actual state	Meaning
A	Active
F	Completed
G	Successfully completed
P	Planned
Q	Released
R	Ended with errors

Table 33. Actions performed when you rerun a process chain job (continued)

Action performed	Description and setting
S	Skipped
X	Canceled
Y	Ready
blank	Undefined
The status and details of the original process chain are updated	In an end-to-end environment, you can perform this action on a centralized job by adding the following parameter to the script file:
	<code>-pchain_pid processID</code>
	IBM Workload Scheduler monitors the original process chain until its final status.
	This action occurs when:
	On extended agents, you specify <code>REFRESHvalue</code> as the step to rerun, where <code>value</code> is any value you want. This setting overrides the setting in the job definition, if any.
	In an end-to-end environment, you can perform this action on a centralized job by adding the following parameter to the script file:
	<code>-flag pchain_refresh</code>
	In the job definition, you set <code>-flag pchain_refresh</code>. For a description of this parameter, see Table 32: Task string parameters for SAP jobs on page 320.

Procedure for rerunning a process chain job

You can rerun all of the processes in the process chain from the Dynamic Workload Console or you can rerun at a process level.

Before you begin

In z/OS environments, you need to set the status of the job to **Ready** before you can rerun the job.

1. Select a job and click **Set Status**.
2. In **Change Status**, select **Ready**.
3. Click **OK** to return to the list of jobs.

About this task

To rerun a process chain SAP job, perform the following steps:

1. Click **Monitoring and Reporting > Workload Monitoring > Monitor Workload**.
2. In **Engine name**, select the name of the IBM Workload Scheduler engine connection from where you want to work with SAP jobs.
3. In **Object Type**, leave the default selection **Job**.
4. Click **Run**.
5. A list of jobs is displayed. Select a process chain job.

Distributed environment

The **Job Type** column displays **SAP Process Chain** to help you identify SAP process chain jobs.

z/OS environment

The **Advanced Job Type** column displays **SAP Process Chain** to help you identify SAP process chain jobs. To display the **Advanced Job Type** column in the table, edit the **Task Properties** and in **Column Definition**, add the **Advanced Job Type** column to the **Selected Columns** list. Move the column up to define the order of the column in the table and make it more visible.

6. Rerun the job.
 - a. Click **More Actions > Restart Process Chain**.
 - b. Select the action you want to perform on the selected process chain:

Rerun

Reruns the entire process chain. The process chain ID on the SAP system remains the same, as well as the job identifier on z/OS systems.

Specify an alias to identify the new job. In distributed systems the rerun process chain is identified with this alias name prefixed by **RERUN**.

Refresh

Refreshes the Dynamic Workload Console view with the latest updates on the remote SAP system so that the two views are synchronized.

Specify an alias to identify the new job. In distributed systems the refreshed process chain is identified with this alias name prefixed by **REFRESH**.

Restart from the failed processes

Action available only for process chains in error state. Rerun only some steps of the process chain, starting from the failed processes.

Specify an alias to identify the new job. In distributed systems the restarted process chain is identified with this alias name prefixed by **RESTART**.

Restart from a specific process

Action available only for process chains in error state. Rerun only some steps of the process chain, starting from the process specified in the **SAP Process ID** field. You can find the process ID by opening the job log or viewing the job type details from the table of results of your monitor job task.

In distributed systems the restarted process chain is identified with this alias prefixed by `PROCESS.`

7. Click **OK** to perform the selected action on the process chain.

Results

The job reruns immediately.

Business scenario: rerunning the original process chain job from the failed process

As a scheduling administrator, you are responsible for managing batch jobs in both SAP and non-SAP systems. The workflow is one or more job streams in IBM Workload Scheduler. A job stream contains jobs that collect and prepare data for month-end closing over all sales channels. The month-end closing report requires data to be collected from several sales and distribution systems. Data is collected using local and remote process chains in the SAP Business Intelligence system. The process chains include a set of Infopackages, ABAP reports, and operating system jobs to sort the report data by a logical hierarchy.

To administer from a single point of control, you link the SAP process chains to IBM Workload Scheduler through IBM Workload Scheduler.

During batch processing, an IBM Workload Scheduler job comprising a process chain, failed. Optionally, you can see which processes failed either from the Dynamic Workload Console (for details, see [Displaying details about a process chain job on page 324](#)) or in the job log. You ask the SAP administrator to fix the cause of the error, then, on an extended agent, you rerun the IBM Workload Scheduler job by setting the step as `RESTARTva Lue`. In this way, the original process chain is restarted from the failed processes and continues until the ending step.

Alternatively, you can select the process chain job from the Monitor Jobs view on the Dynamic Workload Console and then select **More Actions > Restart Process Chain** and then select the **Restart from the failed processes** option.

Business scenario: restarting a specific process of the process chain

You might decide to restart a single process as a preparation step before restarting the failed processes of a process chain. A failed process might have corrupted some data, so you run the single process to restore the data and set up the required system state before you rerun the other processes in the process chain.

Suppose you are using InfoPackages and process chains to extract data from one or several sources and you want to transform this data into managerial reports, for example by using aggregate functions. If the process that transforms this data fails, it might corrupt the data that the preceding InfoPackage process had successfully extracted. After fixing the problem with the transformation process, you must restart the InfoPackage extraction process to reload the data, even

though this extraction process had completed successfully before. Restart the failed transformation process only after the data has been reloaded, either by restarting the failed processes of the process chain or by restarting just the failed transformation process.

On an extended agent, from the Monitor Jobs view on the Dynamic Workload Console, select the process chain and click Rerun, then specify `PROCESSprocessID` as the step to rerun, where `processID` is the identifier of the process you want to restart.

To restart a specific process of the process chain, from the Monitor Jobs view on the Dynamic Workload Console, select the process chain and click **More Actions > Restart Process Chain** and then select the **Restart from a specific process** option, specifying the process ID in the **SAP Process ID** field.

Job throttling feature

Learn how the job throttling feature helps you to improve the efficiency of your scheduling on SAP systems and reduce the batch window for your SAP jobs to a minimum.

Using advanced XBP 2.0 and 3.0 functions, such as the job interception and parent-child, the job throttler ensures that the SAP system is not overloaded and the number of released jobs does not exceed the total number of SAP background work processes in the system.

You can also configure the job throttler to send data related to its activity to the SAP Computing Center Monitoring System (CCMS) for monitoring purposes.

Business scenario

You manage your Internet sales through an application software that verifies that data is correct, checks the availability of the item, and validates the order. To process all the orders received, you scheduled an IBM Workload Scheduler job to run every 12 hours, connect to SAP, and generate a child job for every order to process. Child jobs are in charge of creating shipping bills, checking destination address, and forwarding the orders to the appropriate carrier, thus optimizing the delivery process. A potential overload of the system might occur during peak times, for example over Christmas, and could risk the late delivery of orders, damaging your business. To manage the submission of jobs and activate an advanced management of their priority class (for both parent and child jobs), enable the job throttling feature.

Additionally, you might want to set a policy so that an SAP CCMS alert is raised each time the number of jobs to be released under the control of the job throttler exceeds a certain threshold. To do this, you enable the job throttler to send data to the SAP CCMS monitoring architecture. At job throttler startup, an MTE that monitors the number of jobs to be released by the job throttler is created. By including the MTE in a monitoring set and specifying the related threshold, you are alerted each time the threshold is exceeded.

Software prerequisites

To use job throttling, you must have the SAP JCo 3.0.2 libraries or later (`dll` and `jar` files) installed in the `<data_dir>/methods/throttling/lib` directory. To download JCo 3.0.x, visit the SAP Service Marketplace web site.

Setting and using job throttling

The job throttler enqueues intercepted jobs and releases them when the background work processes that they need on the SAP server or SAP server group are available. The queue of intercepted jobs is sorted by scheduling time and priority of SAP jobs. When the SAP parent-child feature is enabled, child jobs inherit their progenitor's priority so that new urgent jobs are run before other planned jobs.

The following sections describe the steps to operate job throttling.

Step 1. Setting the options in the options file

About this task

To define the behavior of the job throttling feature, set the following options in the options file. For detailed information about the options, see [Table 24: r3batch common configuration options on page 237](#).

- `throttling_enable_job_class_inheritance`
- `throttling_enable_job_interception`
- `throttling_interval`
- `throttling_max_connections`
- `throttling_release_all_on_exit`

Step 2. Enabling and configuring the job interception feature

About this task

As a prerequisite, the job throttler requires that the job interception feature is enabled in the SAP system. To enable and configure job interception, follow these steps.



Note: Ensure that the job throttling and job interception features are not running at the same time. The job throttler cannot start if interception collector jobs are running.

1. Enable job interception, either automatically or manually, as follows:

Automatic activation (meaning that the job throttler enables the job interception on SAP system)

In the options file of the workstation with the r3batch access method you are using, set

`throttling_enable_job_interception=on` (this is the default).

Manual activation

- a. In the SAP system, run the INITXBP2 ABAP program in the transaction `se38` and enable job interception.
- b. In the options file of the workstation with the r3batch access method you are using, set `throttling_enable_job_interception=off`.



Note: When you stop the job throttler, the setting for the job interception feature that was previously configured on the SAP system is restored.

2. In the SAP system, configure the job interception criteria as follows:
 - a. Launch the transaction `se16` to access the table TBCICPT1, where the interception settings are maintained.
 - b. Set the job name, creator, and client related to the jobs you want to intercept. To intercept all SAP jobs, specify the wildcard * (asterisk) for the job name, creator, and client.
 - c. Save your settings and close the dialog.

SAP will intercept all the jobs matching the selection criteria, and the job throttling will release all the jobs that were intercepted.

Step 3. Enabling job class inheritance

About this task

You can configure the job throttler to have the intercepted job inherit the priority class from its progenitor (the top-level job in the hierarchy), if the progenitor class is higher than the intercepted job class. To do this, in the options file set `throttling_enable_job_class_inheritance=on`; this setting automatically enables the parent-child feature on the SAP system.



Note: When you stop the job throttler, the setting for the parent-child feature that was previously configured on the SAP system is restored.

Step 4. Configuring the logging properties

About this task

You can configure the trace properties of the job throttler by editing the logging configuration file `jobthrottling.properties` located in `<data_dir>/methods/throttling/properties`.

To configure the trace level, follow the procedure.

1. Set the trace level property. The supported trace levels are: `DEBUG_MIN`, `DEBUG_MID`, and `DEBUG_MAX`, where `DEBUG_MAX` is the most verbose trace level.
2. Save the changes.

Results

When making changes to the trace level setting, the changes are effective immediately after saving the `.properties` file. Other changes might require a restart to make them effective.

What to do next

You can also configure the name, number, and size of the trace file. By default, the job throttler generates a maximum of 3 files of 5 MB in the `<data_dir>/methods/traces` directory.



Note: The job throttler creates the `<data_dir>/methods/traces` directory as soon as it is started.

Step 5. Starting and stopping the job throttling feature

About this task

To start job throttling, run the `jobthrottling` executable file related to the operating system you are using. Optionally, you can create an IBM Workload Scheduler job that starts the job throttler.



Note: On Windows systems using a single-byte character language, to start job throttling from a command prompt ensure that the DOS shell font is *not* Lucida Console. Ensure also that you set the IBM Workload Scheduler environment by entering the following command:

```
TWA_home\twc_env.cmd
```

From a command prompt, enter:

UNIX operating systems

```
TWA_home/methods/jobthrottling.sh {XA_Unique_ID|base_options_filename} [-scratch]
```

Windows operating systems

```
TWA_home\methods\jobthrottling.bat {XA_Unique_ID|base_options_filename} [-scratch]
```

Where:

XA_Unique_ID

The unique identifier for the extended agent workstation you are using. See [UNIQUE_ID on page 177](#) for details about retrieving the unique identifier for a workstation.

base_options_filename

For dynamic and z-centric agents, the file name of the options file without the extension, defined on the engine workstation hosting the workstation with the r3batch access method.

-scratch

If you enabled the job throttler to send data to CCMS (for details, see [Sending data from job throttling to the CCMS Monitoring Architecture on page 337](#)), the job throttler starts and resets the attribute MTE named `JT total released jobs` to 0. If you do not specify **-scratch**, the job throttler starts and increments the `JT total released jobs`.

This parameter is optional, and has effect only if the job throttler sent its data to CCMS at least once before.

To know the syntax for the `jobthrottling` command, run the command as follows:

```
➤ jobthrottling — -u ➤
```


To stop the job throttler, enter the following command (optionally, you can create an IBM Workload Scheduler job that stops the job throttler):

UNIX operating systems

```
TWA_home/methods/stop-jobthrottling.sh {XA_Unique_ID|base_options_filename}
```

Windows operating systems

```
TWA_home\methods\stop-jobthrottling.bat {XA_Unique_ID|base_options_filename}
```

Alternatively, you can enter the following command (you must be connected as *TWSUser* and have read and write permissions on the `txt` file):

```
echo shutdown > TWA_home/methods/{XA_Unique_ID|base_options_filename}_jobthrottling_cmd.txt
```

The job throttler stops:

- When the timestamp of `{XA_Unique_ID|base_options_filename}_jobthrottling_cmd.txt` is later than the time when the job throttler started.
- Within the time interval you specified in the `throttling_interval` option.

Sending data from job throttling to the CCMS Monitoring Architecture

About this task

You can configure the job throttler to send data related to its activity to the SAP Computing Center Monitoring System (CCMS) for monitoring purposes. Sending data from the job throttler to CCMS is supported if you have at least the SAP Web Application Server 6.20, Support Package 12 installed.

In the options file, set the following options (for details, see [Table 24: r3batch common configuration options on page 237](#)):

```
throttling_send_ccms_data
throttling_send_ccms_rate
```

In this way, at job throttler startup the following monitoring tree elements (MTE) are created:

- A context MTE named **ITWS for Apps**.
- An object MTE with the same name as the IBM Workload Scheduler extended agent where the job throttler is running. This object MTE belongs to the context MTE **ITWS for Apps**.
- The following attribute MTEs:

JT total released jobs

The total number of jobs that the job throttler has released since startup. This value depends on the `-scratch` option you set at job throttler startup; for details, see [Step 5. Starting and stopping the job throttling feature on page 336](#).

JT queue

The number of enqueued intercepted jobs to be released.

JT released jobs per cycle

The number of released jobs in the latest run. This value depends on the `throttling_send_ccms_rate` setting; for details, see [Table 24: r3batch common configuration options on page 237](#).



Note: By default `throttling_release_all_on_exit` is set to `ON`, meaning that when you stop the job throttler, all the intercepted jobs are released. However, these jobs are not considered when updating the **JT total released jobs**, **JT queue**, and **JT released jobs per cycle** MTEs.

To begin monitoring, include the MTEs in the monitoring set you want, and set the thresholds to generate an alert.

You can define an IBM Workload Scheduler event rule based on the CCMS alerts; for detailed information, refer to [Defining event rules based on CCMS Monitoring Architecture alerts on page 361](#).

For example, to define an event that monitors the attribute MTE **JT total released jobs**, on the extended agent workstation with unique identifier `SAP_XA`, connected to the SAP system ID `T01`, specify the following information:

XA Workstation

`SAP_XA`

MTE SAP System ID

`T01`

MTE Monitoring Context Name

`ITWS for Apps`

MTE Monitoring Object Name

`SAP_XA`

MTE Monitoring Attribute Name:

`JT total released jobs`

Deleting the monitoring tree elements

About this task

After you stopped the job throttling feature, if you configured it to send its status data to CCMS, you can delete one or more MTEs that were created. To do this:

1. From the SAP GUI, invoke the transaction `r220` to display a list of monitor sets.
2. Locate the monitor set named SAP CCMS Technical Expert Monitors, and expand it.
3. Locate the monitor named All Monitor Contexts, and double-click it to open it.
4. From the action menu, select **Extras -> Activate Maintenance Functions**.

5. Locate the MTE named **ITWS for Apps** and select it.
6. Right-click the MTE and select **Delete**. You are prompted to choose one of the delete options.
7. Select the option you want. The MTE is deleted accordingly.



Note: Deleting **ITWS for Apps** from the All Monitor Contexts monitor, deletes also all the copies that you might have created in other monitors.

Exporting SAP factory calendars

This section describes how to export SAP factory calendars into a file format that can be processed by the IBM Workload Scheduler **composer** command line, to add the exported calendar definitions to the IBM Workload Scheduler database.

Business scenario

About this task

You might want to configure your IBM Workload Scheduler scheduling activities based on the schedule calendar in your SAP system. To do this, use the `r3batch` export function to export the SAP calendar definitions into a file whose format is compatible with the IBM Workload Scheduler **composer** command line. Based on the parameters you specify, you create a file that contains only the SAP calendar definitions that meet your scheduling requirements. Use this file as input for the **composer add** command, to import the calendar definitions into the IBM Workload Scheduler database. Your IBM Workload Scheduler and SAP calendars are now synchronized.

To keep the IBM Workload Scheduler and SAP calendar definitions synchronized and avoid duplicating data maintenance in the two environments, you can schedule to export the calendar definitions from SAP and import them to IBM Workload Scheduler on a regular basis using a dedicated job.

Exporting and importing SAP factory calendars

Refer to the following sections:

- [Exporting factory calendars on page 339](#) for an explanation about how you use the **r3batch** export function to access and download factory calendars available in an SAP system. The main purpose of this function is to create an output file that can be used by the **composer** to synchronize IBM Workload Scheduler calendars with existing SAP factory calendars, integrating the calendar definitions from SAP into IBM Workload Scheduler.
- [Importing factory calendars on page 342](#) for an explanation about how you import the exported calendar definitions into the IBM Workload Scheduler database.

For details about the IBM Workload Scheduler calendar definitions, see *User's Guide and Reference*.

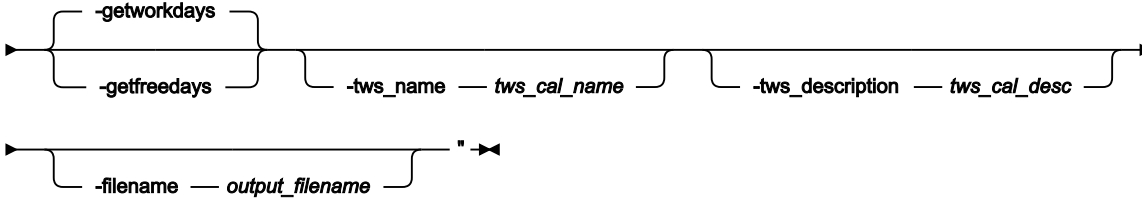
Exporting factory calendars

About this task

To export an SAP calendar, from `TWA_home/methods` (where `TWA_home` is the complete path where you installed IBM Workload Scheduler) enter the following command:

Figure 26. Command syntax

►► `r3batch` — `-t RSC` — `-c` — `XA_Unique_ID` — `-calendar_ID` — `calendarID` — `-year_from` — `yyyy` — `-year_to` — `yyyy` ►



Where:

-t RSC

The identifier of the task to be performed, in this case RSC (Retrieve SAP Calendars). This parameter is required.

-c XA_Unique_ID

The unique identifier for the extended agent workstation connected to the SAP system where the calendar data to export is located. The SAP system must be configured as a workstation to IBM Workload Scheduler. This parameter is required. For information about retrieving the unique identifier for the extended agent workstation, see [UNIQUE_ID on page 177](#).

-calendar_id calendarID

The identifier of the SAP calendar to be exported, which consists of two alphanumeric characters. This parameter is required.

-year_from yyyy

The year of the calendar from when to start exporting dates, in the format yyyy. This parameter is required.

-year_to yyyy

The year of the calendar when to stop exporting dates, in the format yyyy. This parameter is required.

-getworkdays | -getfreedays

Specify `getworkdays` to create the IBM Workload Scheduler calendar definition based on the working days of the SAP calendar. In this way, each date of a working day is stored in the output file.

Specify `getfreedays` to create the IBM Workload Scheduler calendar definition based on the holidays of the SAP calendar. Each date of a non-working day is stored in the output file.

These parameters are optional and mutually exclusive. If you do not specify either, the default is `getworkdays`.

-twc_name twc_cal_name

The IBM Workload Scheduler name for the exported SAP factory calendar. It is stored in the output file.

You can specify up to eight alphanumeric characters. This parameter is optional, the default is `SAPXX_calendarID`, where:

XX

Corresponds to WK if the calendar includes only working days or FR if the calendar includes only non-working days.

calendarID

The identifier of the SAP calendar.

For example, the default IBM Workload Scheduler name for an exported calendar, whose identifier is 04, that includes only working days, is `SAPWK_04`.

-twc_description tws_cal_desc

The description of the IBM Workload Scheduler calendar. It is stored in the output file. You can specify up to 120 alphanumeric characters. If the description contains blanks, it must be enclosed between single quotes. This parameter is optional.

-filename output_filename

The name of the output file that is to contain the calendar definitions. This file is written in a scheduling language that can be processed by the **composer** when you add the calendar data to the IBM Workload Scheduler database.

You can specify a file name with its complete or partial path; if you do not specify any path, the file is created in the current directory. If the path you specify does not exist, it is created, provided that you have the appropriate access rights. Otherwise, the command returns an error message and is not performed.

You can specify up to the maximum number of characters allowed by your operating system. If the name of the file contains blanks, it must be enclosed between single quotes. If another file with the same name exists, it is overwritten.

This parameter is optional. The default value is `tws_name.txt`, where `tws_name` is the value you set for the `tws_name` parameter.

The following is an example of an SAP factory calendar export command:

```
r3batch -t RSC -c horse10 -- " -calendar_id 01 -year_from 2007
-year_to 2010 -tws_name CAL1 -twc_description 'SAP Calendar 01'
-getworkdays -filename 'my dir/calendar_01.dat' "
```

This command exports the SAP calendar named 01, located on the SAP system named horse10. The dates exported begin from year 2007, until year 2010, considering only working days. The IBM Workload Scheduler name used for the calendar is CAL1, and the description written in the output file is SAP Calendar 01. The output file is named `calendar_01.dat`, stored in `<data_dir>/methods/my_dir`, and its content looks like the following

```
$CALENDAR
CAL1
"SAP Calendar 01"
```

```

01/02/2007 01/03/2007 01/04/2007 01/05/2007 01/08/2007 01/09/2007 01/10/2007
01/11/2007 01/12/2007 01/15/2007 01/16/2007 01/17/2007 01/18/2007 01/19/2007
01/22/2007 01/23/2007 01/24/2007 01/25/2007 01/26/2007 01/29/2007 01/30/2007
01/31/2007 02/01/2007 02/02/2007 02/05/2007 02/06/2007 02/07/2007 02/08/2007
.....
11/24/2010 11/25/2010 11/26/2010 11/29/2010 11/30/2010 12/01/2010 12/02/2010
12/03/2010 12/06/2010 12/07/2010 12/08/2010 12/09/2010 12/10/2010 12/13/2010
12/14/2010 12/15/2010 12/16/2010 12/17/2010 12/20/2010 12/21/2010 12/22/2010
12/23/2010 12/24/2010 12/27/2010 12/28/2010 12/29/2010 12/30/2010 12/31/2010

```

Importing factory calendars

About this task

To import the exported calendar definitions into the IBM Workload Scheduler database, copy the output file from the extended agent for SAP to the master workstation and from the **composer** command line on the master workstation, enter the following command:

```
-add output_filename
```

where *output_filename* is the name of the exported file, with its complete path.

For example, to import the *twc_calendar_01.dat* file exported in the previous example, copy the file to the master workstation. From the **composer** command line on the master workstation, enter:

```
-add TWA_home/methods/my_dir/twc_calendar_01.dat
```

where *TWA_home* is the complete path where you installed IBM Workload Scheduler.

Defining internetwork dependencies and event rules based on SAP background events

This section describes how to define internetwork dependencies and event rules for IBM Workload Scheduler based on SAP background events.



Note: To be able to define and monitor event rules, you must configure your environment as described in [Configuring SAP event monitoring on page 251](#).

Defining internetwork dependencies based on SAP background events

Dependencies are prerequisites that must be satisfied before a job or job stream can start. Internetwork dependencies are dependencies checked by the extended agent workstation to which they belong. In response to an internetwork dependency, the SAP extended agent checks for the occurrence of the SAP background event specified in the dependency. As soon as the SAP event is raised, the SAP extended agent commits the event and instructs IBM Workload Scheduler to resolve the corresponding internetwork dependency.

For more details about internetwork dependencies, refer to the *IBM Workload Scheduler: User's Guide and Reference*. For more details about how to raise SAP events, see [Raising an SAP event on page 278](#).

To define SAP background events as internetwork dependencies, XBP versions 2.0 and 3.0 are supported, with the following differences:

XBP version 2.0

SAP background events can release IBM Workload Scheduler internetwork dependencies only if the dependencies are created or checked *before* the SAP event is raised. An event history is ignored, therefore an SAP event raised before the internetwork dependency is created, is not considered.



Note: Because an SAP event history is ignored, for each SAP background event to be checked, a placeholder SAP job is created. This is a dummy job whose running depends on the SAP background event, therefore an SAP event is considered raised as soon as the corresponding placeholder job has completed.

XBP version 3.0 (supported by SAP NetWeaver 7.0 with SP 9, or later)

Only the SAP background events stored in the SAP event history table are considered by IBM Workload Scheduler to check for internetwork dependencies resolution. As a prerequisite, the SAP administrator must create the appropriate event history profiles and criteria on the target SAP system.

To avoid performance reduction, run reorganization tasks against the SAP event history.



Note: Some SAP systems providing XBP version 3.0 still return XBP version as 2.0. To check if your SAP system provides XBP 3.0, invoke the transaction `se37` and search for the function module `BAPI_XBP_BTC_EVTHISTORY_GET`. If your system contains the module, set the `xbpversion` option to 3. In this way, **r3batch** will ignore the XBP value returned by the SAP system. For details about the `xbpversion` option, refer to [Table 24: r3batch common configuration options on page 237](#).

To define an SAP background event as an internetwork dependency, use the following parameters:

Table 34. Parameters to define an SAP internetwork dependency

Parameter	Description	GUI support
-evtid <i>sap_event_name</i>	The name of the SAP background event, up to 32 characters. If the name contains blanks, enclose it between single quotes. This parameter is required.	✓
-evtpar <i>sap_event_parm</i>	The SAP event parameter, up to 64 characters. If the parameter contains blanks, enclose it between single quotes. This parameter is optional.	✓
-commit	Defines that the SAP background event is committed immediately after the internetwork dependency has been resolved. If you do not specify <code>-commit</code> , the event must be committed by running the <code>r3batch</code> task PI. The default is that <code>-commit</code> is not specified. For details about the PI task, refer to Committing SAP background events by an external task on page 345 . In addition to this parameter, you can set as default that the system commits internetwork dependencies immediately by specifying <code>commit_dependency=on</code> in	✓

Table 34. Parameters to define an SAP internetwork dependency (continued)

Parameter	Description	GUI support
	the options file. For details about the commit_dependency option, see Table 24: r3batch common configuration options on page 237 .	
	 Note: With XBP version 2.0, defining two internetwork dependencies on the same SAP event might lead to an error, if -commit is specified. For example, suppose you define an internetwork dependency for the SAP event SAPEVT, with or without setting -commit. After this definition, the SAP event SAPEVT is raised. Then you define a second internetwork dependency based on SAPEVT, specifying -commit. The second dependency immediately commits the SAP event, with the consequence that the first dependency becomes impossible to resolve. Therefore, when the first job checks for the internetwork dependency, an error is issued.	

The following example shows how to define an internetwork dependency based on the SAP background event named SAP_TEST with the parameter 12345678. After its processing, the event is not immediately committed.

```
-evtid SAP_TEST -evtpar 12345678
```

The resulting internetwork dependency looks like the following, where SAPWS is the name of the extended agent workstation that connects to the SAP background processing system where the event runs:

```
follows SAPWS::-evtid SAP_TEST -evtpar 12345678"
```

The following example shows how to define an internetwork dependency based on the SAP background event named SAP_TEST, without parameter. As soon as the internetwork dependency is resolved, the event is committed.

```
-evtid SAP_TEST -commit
```

The resulting internetwork dependency looks like the following, where SAPWS is the name of the extended agent workstation that connects to the SAP background processing system where the event runs:

```
follows SAPWS::-evtid SAP_TEST -evtpar 12345678"
```

[Table 35: Internetwork dependency definition and possible resolution on page 344](#) shows the correspondence between the definition and possible resolution of an internetwork dependency that depends on an SAP event, with or without parameters assigned. In this table, SAP_TEST is used as the event name and 12345678 or ABCDEFG as the event parameter.

Table 35. Internetwork dependency definition and possible resolution

IBM Workload Scheduler internetwork dependency specified	SAP event raised in SAP system	SAP event pa rameter	IBM Workload Scheduler internetwork dependency resolved
-evtid SAP_TEST	none	none	No

Table 35. Internetwork dependency definition and possible resolution (continued)

IBM Workload Scheduler internetwork dependency specified	SAP event raised in SAP system	SAP event pa rameter	IBM Workload Scheduler internetwork dependency resolved
-evtid SAP_TEST	END_OF_JOB	none	No
-evtid SAP_TEST	SAP_TEST	none	Yes
-evtid SAP_TEST	SAP_TEST	12345678	Yes
-evtid SAP_TEST -evtpar 12345678	SAP_TEST	none	No
-evtid SAP_TEST -evtpar 12345678	SAP_TEST	12345678	Yes
-evtid SAP_TEST -evtpar 12345678	SAP_TEST	ABCDEFGF	No

Committing SAP background events by an external task

About this task

SAP events defined as IBM Workload Scheduler internetwork dependencies, by default are not automatically committed after their processing. You can modify this default by specifying the `-commit` parameter. Otherwise, if you leave the default, you must commit the processed event by using the external task Put Information (PI).

The PI task commits all the processed events that meet the given criteria. For this reason, it is recommended that you run this task at the end of the working day. By doing so, internetwork dependencies that are already resolved are not reset and the objects depending on them are not blocked until they are resolved again.

From a command line, enter the following command:

Figure 27. Command syntax

```

▶▶ -r3batch — -t PI — -c — XA_Unique_ID — -" -t CE — -evtid — sap_event_name —
                                     └─ -evtpar — sap_event_parm ─┘
▶ — " —▶

```

Where:

-t PI

The identifier of the task to be performed, in this case PI (Put Information). This parameter is required.

-c XA_Unique_ID

The unique identifier for the extended agent workstation connected to the SAP background processing system where the event is run. This parameter is required. For information about retrieving the unique identifier for the extended agent workstation, see [UNIQUE_ID on page 177](#).

-t CE

The identifier of the task to be performed, in this case CE (Commit Event). This parameter is required.

-evtid sap_event_name

The name of the SAP event running on the background processing system. If the name contains blanks, enclose it between single quotes. This parameter is required.

-evtpar sap_event_parm

The parameter of the SAP event running on the background processing system. If the parameter contains blanks, enclose it between single quotes. This parameter is optional. If you do not specify it, all the SAP events with the name you specified, with or without a parameter, are committed on the target system.

The following is an example of how to commit the SAP event named `SAP_TEST`, with parameter `1234567`, with extended agent workstation with unique identifier `horse10` connected to the background processing system:

```
r3batch -t PI -c horse10 -- " -t CE -evtid SAP_TEST -evtpar 1234567"
```

Defining internetwork dependencies based on SAP background events with the Dynamic Workload Console

About this task

To define an SAP background event as an internetwork dependency with the Dynamic Workload Console, perform the following steps:

1. Launch the **Workload Designer** from the Dynamic Workload Console. From the **Design** menu, click **Workload Designer** page.
2. Search for and open the job stream you want to manage.
 - a. Select the **Job stream card** to display all job streams present in the selected folder. Otherwise, you can select the **Job stream card** and then type the job stream name in the **Search** bar.
 - b. Select the job stream and click **Edit**. The job stream and its contents are displayed in the **Workspace area**.
3. In the job stream subrow, click **Add Dependency** and select **Internetwork**.
4. Specify the properties for the internetwork dependency.
 - a. In the **Network Agent** field, enter the name of the agent workstation connected to the SAP background processing system where the event runs.
 - b. In the **Dependency** field, enter the parameters to define the internetwork dependency. For a description of the parameters allowed, refer to [Table 34: Parameters to define an SAP internetwork dependency on page 343](#).
5. Click **Save** to save the changes to the job stream.

Results

The local job or job stream now has a dependency on a SAP background event. You can also perform this procedure from the graphical view available from the job stream row in the **Workspace area**. For more information about adding dependencies and editing objects in the graphical view, refer to the Dynamic Workload Console User's Guide.

Defining event rules based on SAP background events

A scheduling event rule defines a set of actions to run when specific event conditions occur. The definition of an event rule correlates events and triggers actions.

An event rule is identified by a rule name and by a set of attributes that specify if the rule is active, the time frame of its validity, and other information required to decide when actions are triggered. It includes information related to the specific events (`eventCondition`) that the rule must detect and the specific actions it is to trigger upon their detection or timeout (`ruleAction`). Complex rules might include multiple events and multiple actions.

If you are using XBP 3.0, only the SAP background events that are stored in the event history table are considered by IBM Workload Scheduler.

To define event rules, you can use either of the following:

The composer command line

You edit the rules with an XML editor of your choice. For details about how to use the composer to define event rules, see the *IBM Workload Scheduler User's Guide and Reference*.

The Dynamic Workload Console

For information about creating an event rule, see the section about creating an event rule in *Dynamic Workload Console User's Guide*.

The SAP background event is identified by the following information:

SAP Event ID

The name identifying the SAP event. Wildcards are not allowed.

If you are using the Dynamic Workload Console, you can type the event name in the SAP Event ID field. This field does not support wildcard characters (* and %), nor the following special characters: asterisk (*), question mark (?), and backslash (\). Note that for supported special characters, the escape character (\) must not be used.

Alternatively, you can use the lookup function to search for and select the event name. When specifying the string to search for that represents the SAP Event ID, wildcard characters are supported, (* and %). For example, if you specify "myevent*", then results can include events such as "myevent", "myevent%", and "myevents".

Event parameter

The parameter associated with the SAP event, if any. Wildcards are not allowed.

If you are using the Dynamic Workload Console, the following special characters are not supported when specifying the event parameter: asterisk (*), question mark (?), and backslash (\).

Extended or dynamic agent workstation

The name of the extended agent workstation or the name of the dynamic agent workstation running event monitoring.



Note:



1. If you specify a pattern with the wildcard asterisk (*), all the agents whose name matches the pattern will monitor the specified event.
2. As a best practice, define that an event belonging to an SAP system is monitored by one agent workstation only. If the same SAP event is monitored by more than one agent, you might either be notified multiple times for the same event occurrence or the first agent that notifies the event occurrence makes that event unavailable to the other agents.
3. If you modify the extended agent configuration in the `r3batch` option files, to make the changes effective you must stop and restart the agent.
4. For dynamic agents you can specify the name of a local options file. In the Properties section of the Create Event Rules window of the Dynamic Workload Console a lookup button provides a list of all the local options files associated with that agent. If you do not specify the name of a local options file, the global options file is used by default in the rule definition.

SAP events matching criteria

The SAP background events specified in the event rule are matched with the events raised in the SAP system, according to the following criteria. Depending on the parameters you set:

The SAP event ID and parameter are specified in the event rule

To match, the SAP event ID and parameter must be the same as the event ID and event parameter raised in the SAP system. Also, the event state must be N (New). SAP events with a different parameter or without any parameter are ignored.

The information collected about the matching SAP event is sent by the `r3evmon` process to IBM Workload Scheduler. If the notification is successfully sent, the event is committed on the SAP system and its state changed to C (Confirmed).

For example, you define an event rule in your IBM Workload Scheduler plan based on the following SAP event:

SAP event ID

SAP_TEST

SAP event parameter

ABCDEF

Workstation

An extended agent named GENIUS

According to these settings, a file named `GENIUS_r3evmon.cfg` is created on `GENIUS`. It contains the following `!R3EVENT` keyword:

```
!R3EVENT 0008SAP_TEST0006ABCDEF
```

Monitoring of the `SAP_TEST` event with parameter `ABCDEF` is automatically started. Suppose that the following SAP events were raised on the SAP system:

Table 36. History table of the SAP events raised

EVENT GUID	SAP EVENT ID	EVENT PARM	EVENT SERVER	EVENT TIMESTAMP	EVENT STATE	PROCESS STATE	COUNT OF JOBS
1234	SAP_TEST	ABC123	...	20070925 13:00	C	OK	1
2345	SAP_TEST	ABCD	...	20070925 14:00	N	OK	2
3456	SAP_TEST		...	20070925 15:00	N	OK	3
4567	SAP_TEST	ABCDEF	...	20070925 16:00	N	OK	4

Only the following SAP event is notified to IBM Workload Scheduler:

Table 37. SAP event matching with the event rule defined

EVENT GUID	SAP EVENT ID	EVENT PARM	EVENT SERVER	EVENT TIMESTAMP	EVENT STATE	PROCESS STATE	COUNT OF JOBS
4567	SAP_TEST	ABCDEF	...	20070925 16:00	N	OK	4

If the notification is successfully sent, the event is committed on the SAP system and its state changed to C (Confirmed).

Only the SAP event ID is specified in the event rule

To match, the SAP event ID must be the same as the ID of the events raised in the SAP system whose state is N (New). The parameters of the SAP events, whether specified or not, are not taken into account.

The information collected about all the matching SAP events is sent by the `r3evmon` process to IBM Workload Scheduler. Each event successfully notified is committed on the SAP system and its status changed to C (Confirmed).

For example, you define an event rule in your IBM Workload Scheduler plan based on the following SAP event:

SAP event ID

SAP_TEST

Workstation

GENIUS

According to these settings, a file named `GENIUS_r3evmon.cfg` is created on `GENIUS`. It contains the following

`!R3EVENT` keyword:

```
!R3EVENT 0008SAP_TEST
```

Monitoring of the `SAP_TEST` event is automatically started. Suppose that the following SAP events were raised on the SAP system:

Table 38. History table of the SAP events raised

EVENT GUID	SAP EVENT ID	EVENT PARM	EVENT SERVER	EVENT TIMESTAMP	EVENT STATE	PROCESS STATE	COUNT OF JOBS
1234	SAP_TEST	ABC123	...	20070925 13:00	C	OK	1
2345	SAP_TEST	ABCD	...	20070925 14:00	N	OK	2
3456	SAP_TEST		...	20070925 15:00	N	OK	3
4567	SAP_TEST	ABCDEF	...	20070925 16:00	N	OK	4

Only the following SAP events are notified to IBM Workload Scheduler:

Table 39. SAP events matching with the event rule defined

EVENT GUID	SAP EVENT ID	EVENT PARM	EVENT SERVER	EVENT TIMESTAMP	EVENT STATE	PROCESS STATE	COUNT OF JOBS
2345	SAP_TEST	ABCD	...	20070925 14:00	N	OK	2
3456	SAP_TEST		...	20070925 15:00	N	OK	3
4567	SAP_TEST	ABCDEF	...	20070925 16:00	N	OK	4

Each event whose notification is successfully sent is committed on the SAP system and its state changed to C (Confirmed).

Setting a filter for SAP background events in the security file

In the security file, you can filter the SAP background events that can be used to define event rules. By doing this, you restrict the use of certain SAP events to specific users. For example, assume that you want your USA department to manage only the SAP events whose ID begins with `SAP_USA`, and your Italy department to manage all events except those beginning with `SAP_USA`. In the security file that defines the user access for the USA department, define the `CUSTOM` keyword for the `EVENT` object as follows:

```
EVENT      PROVIDER=@ +CUSTOM=SAP_USA@  ACCESS=USE
```

where:

PROVIDER=@

Specifies that the user can use the events coming from any provider.

+CUSTOM=SAP_USA@

Specifies that the user can use only the SAP events whose ID begins with `SAP_USA`.

This keyword applies only to the SAP provider (SapMonitor).

ACCESS=USE

Sets the user access to the object to `USE`.

In the security file that defines the user access for the Italy department, define the `CUSTOM` keyword for the `EVENT` object as follows:

```
EVENT      PROVIDER=@ ~CUSTOM=SAP_USA@  ACCESS=USE
```

where:

PROVIDER=@

Specifies that the user can use the events coming from any provider.

~CUSTOM=SAP_USA@

Specifies that the user can use all SAP events, except those whose ID begins with `SAP_USA`.

This keyword applies only to the SAP provider (SapMonitor).

ACCESS=USE

Sets the user access to the object to `USE`.

For more details about the security file and how to set up user authorizations, see the *IBM Workload Scheduler: Administration Guide*.

Defining event rules based on IDoc records

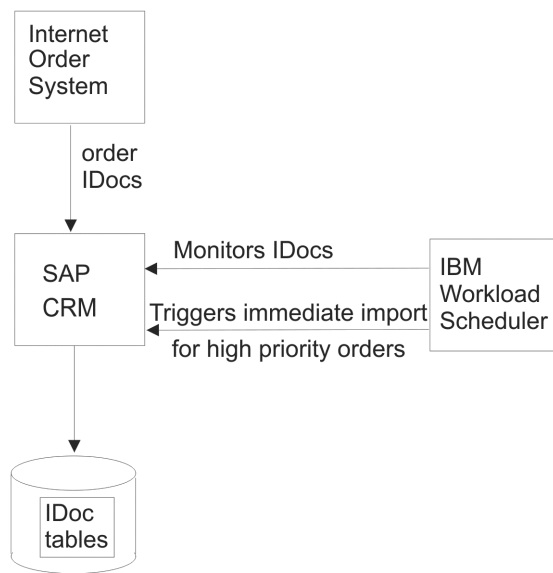
You can use IBM Workload Scheduler to monitor Intermediate Document (IDoc) records in SAP systems and forward events to the IBM Workload Scheduler event integration framework.

To do this, you define an event condition that contains the criteria that the IDocs must match to be forwarded to IBM Workload Scheduler. When the event condition occurs, the action that you associated with it (for example, running a job) is performed.

Business scenario

You connected your Internet sales application to your SAP Customer Relationship Management (CRM) system, which receives the orders as incoming IDocs. The orders are classified as emergency and ordinary, and therefore have different IDoc message types. You want the emergency orders to be imported into the CRM system directly, and the ordinary orders to be processed in batch mode. To do this, in IBM Workload Scheduler, you define an event rule that monitors the IDoc message types corresponding to emergency orders and sends an event to IBM Workload Scheduler. In IBM Workload Scheduler, you define a job to be released when this type of event is received and is linked to an SAP job that runs an import ABAP report for these specific types of IDocs.

Figure 28. Managing high priority IDocs overview



Creating event rules based on IDocs

About this task

To define event rules based on IDocs, specify the fields to be used as matching criteria during IDoc monitoring. For details about these fields, refer to [Events matching criteria on page 353](#). To create the event rules, you can use either of the following:

The composer command line

You edit the rules with an XML editor of your choice. For a general explanation about how to use the composer to define event rules, see the *IBM Workload Scheduler: User's Guide and Reference*. The event condition requires:

- `SAPMonitor` as event monitor provider.
- `IDOCEventGenerated` as event type.

For a list of the values that you can specify in the **attributeFilter** name when defining the event condition, refer to [Table 42: Parameters of IDOCEventGenerated event type on page 355](#).

The Dynamic Workload Console

For information about creating an event rule, see the section about creating an event rule in *Dynamic Workload Console User's Guide*.



Note:



1. To be able to define and monitor event rules, ensure that you configured your environment as described in [Configuring SAP event monitoring on page 251](#).
2. To configure how IBM Workload Scheduler retrieves the IDoc monitors, set **idoc_no_history** and **idoc_shallow_result** in the options file. For details about these options, refer to [Defining the common options on page 237](#).

Events matching criteria

[Table 40: IBM Workload Scheduler fields used to define event rules based on IDocs on page 353](#) lists the IBM Workload Scheduler fields corresponding to the fields in the IDoc record that you want to search. During monitoring, each IDoc matching the search criteria generates an event that is sent to IBM Workload Scheduler.

Table 40. IBM Workload Scheduler fields used to define event rules based on IDocs

Composer property	Console property	IDoc field
SAPClient	SAP client	MANDT
SAPIDocStatus	Status	STATUS
SAPDirectionIDocTransmission	Direction	DIRECT
SAPReceiverPort	Receiver port	RCVPOR
SAPReceiverPartnerFunction	Receiver partner function	RCVPFC
SAPReceiverPartnerType	Receiver partner type	RCVPRT
SAPReceiverPartnerNumber	Receiver partner number	RCVPRN
SAPSenderPort	Sender port	SNDPOR
SAPSenderPartnerType	Sender partner type	SNDPRT
SAPSenderPartnerFunction	Sender partner function	SNDPFC
SAPSenderPartnerNumber	Sender partner number	SNDPRN
SAPLogicalMessageType	Logical message type	MESTYP
SAPNameOfBasicType	Name of basic type	IDOCTP
SAPLogicalMessageCode	Logical message code	MESCOD
SAPLogicalMessageFunction	Logical message function	MESFCT
SAPTestFlag	Test flag	TEST
SAPOutputMode	Output mode	OUTMOD

Optionally, you can define also correlation rules by using the fields listed in [Table 41: IBM Workload Scheduler fields used to define correlation rules for IDoc events on page 354](#). Date and time values are specified in GMT time zone.

Table 41. IBM Workload Scheduler fields used to define correlation rules for IDoc events

Composer property	Console property	IDoc field
SAPIDocNumber	IDoc number	DOCNUM
SAPReleaseForIDoc	IDoc SAP release	DOCREL
SAPIDocType	IDoc type	DOCTYP
SAPReceiverAddress	Receiver SADR address	RCVSAD
SAPReceiverSADRClient	Receiver SADR client	RCVSMN
SAPFlagForInternationalReceiverAddress	Receiver SADR flag	RCVSNA
SAPReceiverCommunicationType	Receiver SADR communication type	RCVSCA
SAPDefaultFlagForReceiverAddress	Receiver SADR default flag	RCVSDF
SAPReceiverAddressSequentialNumber	Receiver SADR sequential number	RCVSLF
SAPReceiverLogicalAddress	Receiver logical address	RCVLAD
SAPEDIStandard	EDI Standard	STD
SAPEDIStandardVersion	EDI standard version	STDVRS
SAPEDIMessageType	EDI message type	STDMES
SAPSenderAddress	Sender SADR address	SNDSAD
SAPSenderSADRClient	Sender SADR client	SNDSMN
SAPFlagForInternationalSenderAddress	Sender SADR flag	SNDSNA
SAPSenderCommunicationType	Sender SADR communication type	SNDSKA
SAPDefaultFlagForSenderAddress	Sender SADR default flag	SNDSDF
SAPSenderAddressSequentialNumber	Sender SADR sequential number	SNDSLK
SAPSenderLogicalAddress	Sender logical address	SNDLAD
SAPReferenceToInterchangeFile	Interchange file reference	REFINT
SAPReferenceToMessageGroup	Message group reference	REFGRP
SAPReferenceToMessage	Message reference	REFMES
SAPEDIArchiveKey	EDI archive key	ARCKEY
SAPIDocCreationDate	IDoc creation date	CREDAT
SAPIDocCreationTime	IDoc creation time	CRETIM
SAPExtension	Extension	CIMTYP
SAPEDIALESerializationField	EDI/ALE Serialization field	SERIAL
SAPOverridingInInboundProcessing	Overriding in inbound processing	EXPRSS

Table 41. IBM Workload Scheduler fields used to define correlation rules for IDoc events (continued)

Composer property	Console property	IDoc field
SAPIDocChangeDate	IDoc last update date	UPDDAT
SAPIDocChangeTime	IDoc last update time	UPDTIM

Based on the defined rule, the `r3evmon` process of IBM Workload Scheduler monitors the events related to IDoc records according to a polling rate. To customize this polling rate, use the `evmon_interval` option; for details, see [Defining the common options on page 237](#).

[Table 42: Parameters of IDOCEventGenerated event type on page 355](#) lists the values that you can specify as attribute filter name when defining the event condition.

Table 42. Parameters of IDOCEventGenerated event type

Property name	Description	Type	Filtering allowed	Required	Multiple values allowed	Wildcard allowed	Length (min-max)
SAPClient	SAP client number	numeric (0-9)	✓	✓		✓	1 3
	IDoc status information						
	For a list of allowed values, refer to Table 43: Standard outbound IDoc statuses on page 357 and Table 44: Standard inbound IDoc statuses on page 358 .						
SAPIDocStatus		numeric	✓	✓	✓		1 2
		numeric					
SAPDirectionIDocTransmission	IDoc direction	Value can be 1 (outbound) or 2 (inbound).	✓	✓			1 1
SAPReceiverPort	Receiver port. SAP system, EDI subsystem	string	✓				1 10
SAPReceiverPartnerFunction	Partner function of receiver	string	✓				1 2
SAPReceiverPartnerType	Partner type of receiver	string	✓				1 2
SAPReceiverPartnerNumber	Partner number of receiver	string	✓				1 10
SAPSenderPort	Sender port. SAP system, EDI subsystem	string	✓				1 10
SAPSenderPartnerType	Partner type of sender	string	✓				1 2

Table 42. Parameters of IDOCEventGenerated event type

(continued)

Property name	Description	Type	Filtering allowed	Required	Multiple values allowed	Wildcard allowed	Length (min-max)
SAPSenderPartnerFunction	Partner function of sender	string	✓				1 2
SAPSenderPartnerNumber	Partner number of sender	string	✓				1 10
SAPLogicalMessageType	Logical message type	string	✓			✓	1 30
SAPNameOfBasicType	Name of basic type	string	✓			✓	1 30
SAPLogicalMessageCode	Logical message code	string	✓				1 3
SAPLogicalMessageFunction	Logical message function	string	✓				1 3
SAPTestFlag	Test flag	string	✓				1 1
		string					
		Value can be 2					
SAPOutputMode	Output Mode	(immediate	✓				1 1
		sending) or 4					
		(collected sending).					

Table 43: Standard outbound IDoc statuses on page 357 lists the standard outbound IDoc statuses and Table 44: Standard inbound IDoc statuses on page 358 lists the standard inbound IDoc statuses. Optionally, you can activate a check to prevent event rule definitions with inconsistent IDoc status list and direction. If you activate the check and specify inconsistent values when defining a rule (for example, 02 as status and 2 as direction), you receive an error message and you cannot save the rule definition. To activate the check, perform the following steps:

1. In the `TWA_home>\eventPlugIn` directory on Windows™ and in the `TWA_DATA_DIR/eventPlugIndirectory` on UNIX®, create the `SapMonitorPlugIn.properties` file.
2. Edit `SapMonitorPlugIn.properties` to set the following configuration property:

```
TWSPlugIn.event.idoc.consistency.check = true
```
3. From `conman`, stop and restart the event processing server by using, respectively, the **stopeventprocessor** and **starteventprocessor** commands.

The default value is `false`.

To have predictable event action results, when defining event rules consider using only non-transitory statuses that allow user checks.

Table 43. Standard outbound IDoc statuses

Status	Description
01	IDoc generated
02	Error passing data to port
03	Data passed to port
04	Error within control information of EDI subsystem
05	Error during translation
06	Translation
07	Error during syntax check
08	Syntax check
09	Error during interchange
10	Interchange handling
11	Error during dispatch
12	Dispatch OK
13	Retransmission OK
14	Interchange acknowledgement positive
15	Interchange acknowledgement negative
16	Functional acknowledgement positive
17	Functional acknowledgement negative
18	Triggering EDI subsystem OK
19	Data transfer for test OK
20	Error triggering EDI subsystem
22	Dispatch OK, acknowledgement still due
23	Error during retransmission
24	Control information of EDI subsystem OK
25	Processing despite syntax error
26	Error during syntax check of IDoc
27	Error in dispatch level (ALE service)
29	Error in ALE service
30	IDoc ready for dispatch (ALE service)
31	Error no further processing

Table 43. Standard outbound IDoc statuses (continued)

Status	Description
32	IDoc was edited
33	Original of an IDoc which was edited
34	Error in control record of IDoc
36	Electronic signature not performed (timeout)
37	IDoc added incorrectly
38	IDoc archived
39	IDoc is in the target system (ALE service)
40	Application document not created in target system
41	Application document created in target system
42	IDoc was created by test transaction

Table 44. Standard inbound IDoc statuses

Status	Description
50	IDoc added
51	Application document not posted
52	Application document not fully posted
53	Application document posted
54	Error during formal application check
55	Formal application check OK
56	IDoc with errors added
57	Error during application check
58	IDoc copy from R/2 connection
60	Error during syntax check of IDoc
61	Processing despite syntax error
62	IDoc passed to application
63	Error passing IDoc to application
64	IDoc ready to be transferred to application
65	Error in ALE service
66	IDoc is waiting for predecessor IDoc (serialization)

Table 44. Standard inbound IDoc statuses (continued)

Status	Description
68	Error - no further processing
69	IDoc was edited
70	Original of an IDoc which was edited
71	IDoc reloaded from archive
73	IDoc archived
74	IDoc was created by test transaction

For example, you define a rule with the following attributes:

Workstation

A dynamic agent named SAPCPU

SAP client number

001

IDoc status list

56,60

IDoc direction

2 (inbound)

After saving the rule according to these settings, when the rule becomes active a file named `SAPCPU_r3evmon.cfg` is created on SAPCPU. It contains the following `!IDOC` keyword:

[illegible]

IDoc monitoring is automatically started. When the event condition is verified, the action defined in the rule is triggered

For an explanation of the `!IDOC` keyword format, refer to [Table 51: Miscellaneous troubleshooting items on page 376](#).

Examples of event rules based on IDocs

The following example applies to the scenario described in [Business scenario on page 351](#). It shows an event rule that triggers an import ABAP report when an IDoc is added with a message type corresponding to emergency orders.

```
<?xml version="1.0" encoding="UTF-8"?>
<eventRuleSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://www.abc.com/xmlns/prod/tws/1.0/event-management/rules"
  xsi:schemaLocation="http://www.abc.com/xmlns/prod/tws/1.0/
    event-management/rules EventRules.xsd">
  <eventRule name="scenario1_IDoc" ruleType="filter" isDraft="no">
    <eventCondition name="IDocEventRaised1" eventProvider="SapMonitor"
      eventType="IDocEventGenerated">
      <scope>
        001 ON SAPCU WITH 2
      </scope>
    </eventCondition>
  </eventRule>
</eventRuleSet>
```

```

    <filteringPredicate> <attributeFilter name="Workstation" operator="eq">
      <value>SAPCPU</value>
    </attributeFilter>
    <attributeFilter name="SAPClient" operator="eq">
      <value>001</value>
    </attributeFilter>
    <attributeFilter name="SAPIDocStatus" operator="eq">
      <value>50</value>
    </attributeFilter>
    <attributeFilter name="SAPDirectionIDocTransmission" operator="eq">
      <value>2</value>
    </attributeFilter>
    <attributeFilter name="SAPLogicalMessageType" operator="eq">
      <value>EORD1</value>
    </attributeFilter>
  </filteringPredicate>
</eventCondition>
<action actionProvider="TWSaction" actionType="sbj"
  responseType="onDetection">
  <description>Trigger immediate report for high priority orders
</description>
  <parameter name="JobDefinitionWorkstationName">
    <value>MASTER84</value>
  </parameter>
  <parameter name="JobDefinitionName">
    <value>triggerimport</value>
  </parameter>
</action>
</eventRule>
</eventRuleSet>

```

The following example shows an event rule defined to create a ticket for failing IDocs in the SAP Solution Manager or any other problem management system: when an IDoc with a syntax error is detected, the engine submits a job to create a ticket for the failing IDoc.

```

<?xml version="1.0" encoding="UTF-8"?>
<eventRuleSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://www.abc.com/xmlns/prod/tws/1.0/event-management/rules"
  xsi:schemaLocation="http://www.abc.com/xmlns/prod/tws/1.0/
    event-management/rules EventRules.xsd">
  <eventRule name="scenario1_IDoc" ruleType="filter" isDraft="no">
    <eventCondition name="IDocEventRaised1" eventProvider="SapMonitor"
      eventType="IDocEventGenerated">
      <filteringPredicate> attributeFilter name="Workstation" operator="eq">
        <value>SAPCPU</value>
      </attributeFilter>
      <attributeFilter name="SAPClient" operator="eq">
        <value>001</value>
      </attributeFilter>
      <attributeFilter name="SAPIDocStatus" operator="eq">
        <value>60</value>
      </attributeFilter>
      <attributeFilter name="SAPDirectionIDocTransmission" operator="eq">
        <value>2</value>
      </attributeFilter>
      <attributeFilter name="SAPLogicalMessageType" operator="eq">
        <value>MYORD1</value>
      </attributeFilter>
    </filteringPredicate>
  </eventCondition>
</eventRule>
</eventRuleSet>

```



```

    </filteringPredicate>
  </eventCondition>
  <action actionProvider="TWSaction" actionType="sbj"
    responseType="onDetection">
    <description>Create a ticket for failing IDocs
    </description>
    <parameter name="JobDefinitionWorkstationName">
      <value>MASTER84</value>
    </parameter>
    <parameter name="JobDefinitionName">
      <value>createticket</value>
    </parameter>
  </action>
</eventRule></eventRuleSet>

```

Defining event rules based on CCMS Monitoring Architecture alerts

Use CCMS functions to check the performance of the various SAP system components, diagnose potential problems, and be alerted about error and warning conditions.

The SAP Computing Center Monitoring System (CCMS) is a centralized monitoring architecture that provides a set of monitors for monitoring the SAP environment. Using the CCMS functions you can check the performance of the various SAP system components, diagnose potential problems, and be alerted about error and warning conditions. The monitors provide you with the information you require to fine tune the SAP system and the operating modes, and hence optimize system performance.

With IBM Workload Scheduler, you can integrate the CCMS monitoring functions into your management infrastructure by defining event rules based on the alerts raised in the SAP system.

Business scenarios

The following sections describe:

- [Business scenario: defining an event rule to process alerts related to IDocs on page 361](#)
- [Business scenario: defining an event rule to process alerts related to operating system on page 362](#)

Business scenario: defining an event rule to process alerts related to IDocs

You connected your Internet sales application to your SAP Customer Relationship Management (CRM) system, which receives the orders as incoming IDocs. You want to import the orders into the CRM system when their number exceeds a specified threshold, therefore you configured your SAP CCMS monitoring architecture to generate an alert when the number of incoming IDocs exceeds a certain value. To automatically start a task that imports the orders:

1. In your SAP CCMS monitoring architecture, identify the element related to the alert that you configured for the incoming order IDocs.
2. In IBM Workload Scheduler, define an event rule, to be active during the timeframe when inbound order traffic is heavy, which monitors the element identified in step 1. As soon as an alert is generated for the element, a CCMS event is sent to IBM Workload Scheduler.
3. In IBM Workload Scheduler, define a job to be submitted when the CCMS event is received, to run an SAP job that runs an import ABAP report for the order IDocs.

Business scenario: defining an event rule to process alerts related to operating system

As an IBM Workload Scheduler administrator, you are in charge of taking the appropriate action in the IBM Workload Scheduler plan when a critical situation occurs in the SAP system. You have an SAP extended agent workstation dedicated to submit Business Intelligence tasks, whose activity you want to suspend every time the SAP BI system faces a critical situation (for example, the SAP system is running out of space). To do this:

1. In your SAP CCMS monitoring architecture, identify the element related to the SAP system you want to monitor.
2. In IBM Workload Scheduler, define an event rule that monitors the element and sends an event to IBM Workload Scheduler when an alert is generated for it. Associate with this event an action that sets the limit of the agent workstation to 0, and sends a mail to the SAP administrator to notify the details of the critical situation.
3. As soon as the SAP administrator resolves the problem, you set the limit of the agent workstation back to its original value to resume the scheduling activities.

Creating event rules based on CCMS alerts

About this task

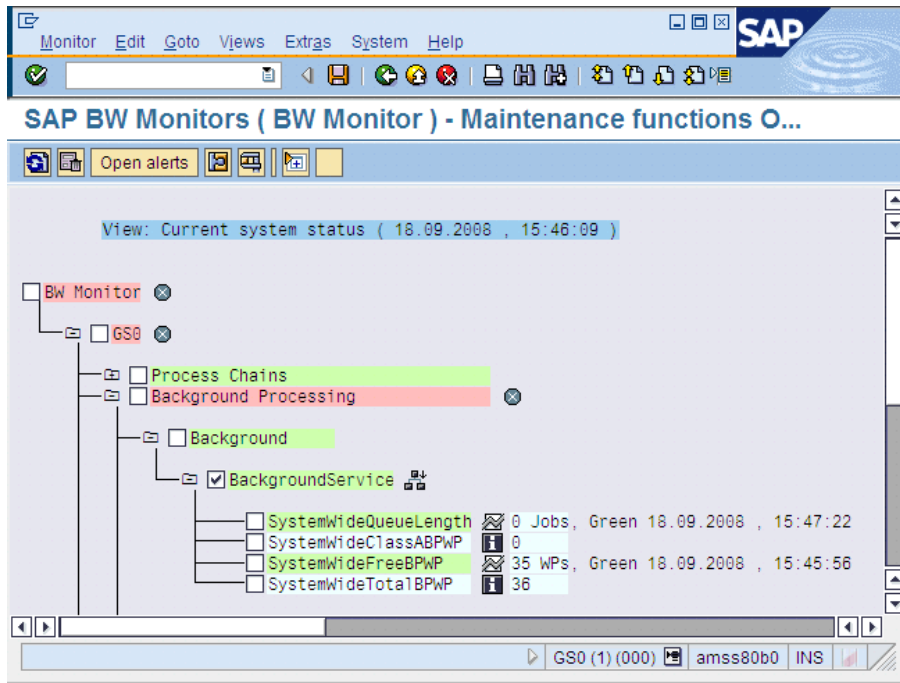
SAP systems are shipped with a predefined set of monitors, grouped in *monitor sets*. A monitor set contains a list of monitors, each monitor contains a set of *monitoring trees*. A monitor is a set of *monitoring tree elements* (MTEs) that are arranged in a hierarchical structure, named *alert monitoring tree*. You can define event rules based on the alert generated for a specific MTE.



Note: To be able to define and monitor event rules, ensure that you configured your environment as described in [Configuring SAP event monitoring on page 251](#).

Figure 29: A monitor and its MTEs - © SAP AG 2009. All rights reserved. on page 363 shows the monitor named **BW Monitor** (belonging to the monitor set **SAP BW Monitor**) and its associated monitor tree elements (MTEs).

Figure 29. A monitor and its MTEs - © SAP AG 2009. All rights reserved.



To configure how IBM Workload Scheduler retrieves the CCMS alerts, set **ccms_alert_history** in the options file. For details about this option, refer to [Defining the common options on page 237](#).

To create event rules, you can use either of the following:

The composer command line

You edit the rules with an XML editor of your choice. For a general explanation about how to use the composer to define event rules, see the *User's Guide and Reference*.

The Dynamic Workload Console

For information about creating an event rule, see the section about creating an event rule in *Dynamic Workload Console User's Guide*.

To define the CCMS event for your rule, specify the following information. For more details about how you separate the MTE name into the individual IBM Workload Scheduler fields, see [Mapping between the MTE name and IBM Workload Scheduler fields on page 365](#).

Extended or dynamic agent workstation

The name of the extended agent workstation or the name of the dynamic agent workstation running event monitoring.



Note:



1. If you specify a pattern with the wildcard asterisk (*), all the agents whose name matches the pattern will monitor the specified event.
2. As a best practice, define that an event belonging to an SAP system is monitored by one agent workstation only. If the same SAP event is monitored by more than one agent, you might either be notified multiple times for the same event occurrence or the first agent that notifies the event occurrence makes that event unavailable to the other agents.
3. If you modify the extended agent configuration in the `r3batch` option files, to make the changes effective you must stop and restart the agent.
4. For dynamic agents you can specify the name of a local options file. In the Properties section of the Create Event Rules window of the Dynamic Workload Console a lookup button provides a list of all the local options files associated with that agent. If you do not specify the name of a local options file, the global options file is used by default in the rule definition.

MTE SAP System ID

Name of the SAP system where the MTE is located (for example, `GS0` in [Figure 29: A monitor and its MTEs - © SAP AG 2009. All rights reserved. on page 363](#)). This field is required. Wildcards are not allowed, you can specify up to eight characters.

MTE Monitoring Context Name

Name of the monitoring context to which the MTE belongs. This field is required. A monitoring context is a logically connected group of monitoring objects that are ordered together under one summary in the monitoring tree (for example, `Background` in [Figure 29: A monitor and its MTEs - © SAP AG 2009. All rights reserved. on page 363](#)).

Wildcards are not allowed, you can specify up to 40 characters.

MTE Monitoring Object Name

Name of the monitoring object in the alert monitor. This field is required. A monitoring object is a component or property of the system that is to be monitored (for example, `BackgroundService` in [Figure 29: A monitor and its MTEs - © SAP AG 2009. All rights reserved. on page 363](#)). If you choose not to specify a value, you must leave the value NULL, which is the default.

Wildcards are not allowed, you can specify up to 40 characters.

MTE Monitoring Attribute Name

Name of the monitoring attribute in the alert monitor. In the monitoring tree, a monitoring attribute is always an end node in the hierarchy (for example, `SystemWideFreeBPWP` in [Figure 29: A monitor and its MTEs - © SAP AG 2009. All rights reserved. on page 363](#)). This field is required. If you choose not to specify a value, you must leave the value NULL, which is the default.

Wildcards are not allowed, you can specify up to 40 characters.

Alert Value

Numeric value that indicates the color of the alert generated for the MTE. This field is optional. You can specify one or a combination of the following values:

1

Green, meaning Everything OK.

2

Yellow, meaning Warning.

3

Red, meaning Problem or error.

If you do not specify any value, all the alerts generated for the MTE are considered.

Alert Severity

Severity of the alert. It can be a number between 0 (lowest) and 255 (highest), or a range among these values.

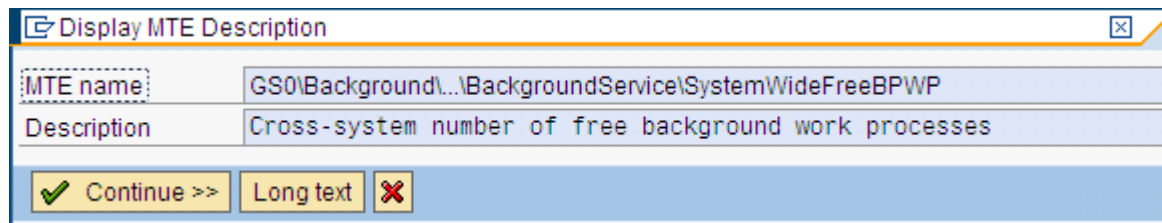
This field is optional. Alert severity is assigned during alert configuration; the SAP standard configuration is 50.

Mapping between the MTE name and IBM Workload Scheduler fields

About this task

Within SAP, MTEs are identified by a name made up of several tokens, separated by backslashes (\). To display the complete MTE name, select the MTE and click **Properties** or press **F1**:

Figure 30. Name and description of an MTE - © SAP AG 2009. All rights reserved.



According to the type of MTE that you want to monitor, you must fill in each IBM Workload Scheduler field with a specific token of the MTE name (to know your MTE type, select the MTE and click **Legend**):

If you are using Dynamic Workload Console V8.5.1, or later

1. In the **Event Rule Editor** panel, from the **Properties** section, click **Autofill MTE Tokens**. The **MTE Name** window opens.
2. In the **MTE Name** field, write the name of the MTE to monitor and click **OK**. You are returned to the **Event Rule Editor** panel, where the IBM Workload Scheduler fields are filled in accordingly.

If you are using Dynamic Workload Console prior to V8.5.1

Refer to the instructions provided in the following sections:

- [Context MTE on page 366](#)
- [Object MTE on page 367](#)
- [Attribute MTE on page 367](#)



Note: Virtual MTEs cannot be monitored.

Context MTE

A context MTE is the uppermost node of a monitoring tree; it contains all the associated object MTEs and attribute MTEs. Context nodes can be either of the following types:

Root

Belongs only to the All Monitoring Contexts monitor. According to the SAP version you are using, a root context MTE name can have either of the following formats:

```
tokenA\tokenB\...  
- OR -  
tokenA\tokenB
```

For example:

```
T10\SystemConfiguration\...
```

Refer to [Table 45: Mapping between root context MTE name and IBM Workload Scheduler fields on page 366](#) for an explanation about how you report this type of MTE in the IBM Workload Scheduler fields:

Table 45. Mapping between root context MTE name and IBM Workload Scheduler fields

IBM Workload Scheduler field	Token of MTE name	In this example...
MTE SAP System ID	<i>tokenA</i>	T10
MTE Monitoring Context Name	<i>tokenB</i>	SystemConfiguration
MTE Monitoring Object Name	N/A	NULL
MTE Monitoring Attribute Name	N/A	NULL

Summary

According to the SAP version you are using, a summary context MTE name can have either of the following formats:

```
tokenA\tokenB\...\tokenC\...  
- OR -  
tokenA\tokenB\tokenC
```

For example:

```
T10\SystemConfiguration\...\InstalledSupportPackages\...
```

Refer to [Table 46: Mapping between summary context MTE name and IBM Workload Scheduler fields on page 367](#) for an explanation about how you report this type of MTE in the IBM Workload Scheduler fields:

Table 46. Mapping between summary context MTE name and IBM Workload Scheduler fields

IBM Workload Scheduler field	Token of MTE name	In this example...
MTE SAP System ID	<i>tokenA</i>	T10
MTE Monitoring Context Name	<i>tokenB</i>	SystemConfiguration
MTE Monitoring Object Name	<i>tokenC</i>	InstalledSupportPackages
MTE Monitoring Attribute Name	N/A	NULL

Object MTE

According to the SAP version you are using, an object MTE name can have either of the following formats:

```
tokenA\tokenB\tokenC\tokenD
- OR -
tokenA\tokenB\...\tokenD
```

For example:

```
PR0\amsp53_PR0_11\R3Services\Background\
```

Refer to [Table 47: Mapping between object MTE name and IBM Workload Scheduler fields on page 367](#) for an explanation about how you report this type of MTE in the IBM Workload Scheduler fields:

Table 47. Mapping between object MTE name and IBM Workload Scheduler fields

IBM Workload Scheduler field	Token of MTE name	In this example...
MTE SAP System ID	<i>tokenA</i>	PR0
MTE Monitoring Context Name	<i>tokenB</i>	amsp53_PR0_11
MTE Monitoring Object Name	<i>tokenD</i>	Background
MTE Monitoring Attribute Name	N/A	NULL

Attribute MTE

According to the SAP version you are using, an attribute MTE name can have either of the following formats:

```
tokenA\tokenB\tokenC\tokenD\tokenE
- OR -
tokenA\tokenB\...\tokenD\tokenE
```

For example:

```
PR0\amsp53_PR0_11\R3Services\Background\AbortedJobs
```

Refer to [Table 48: Mapping between attribute MTE name and IBM Workload Scheduler fields on page 368](#) for an explanation about how you report this type of MTE in the IBM Workload Scheduler fields:

Table 48. Mapping between attribute MTE name and IBM Workload Scheduler fields

IBM Workload Scheduler field	Token of MTE name	In this example...
MTE SAP System ID	<i>tokenA</i>	PR0
MTE Monitoring Context Name	<i>tokenB</i>	amsp53_PR0_11
MTE Monitoring Object Name	<i>tokenD</i>	Background
MTE Monitoring Attribute Name	<i>tokenE</i>	AbortedJobs

Setting correlation rules and action parameters

Optionally, you can use the alert properties listed in [Table 49: Alert properties for correlations on page 368](#) to:

- Define correlation rules between CCMS events.
- Specify additional parameters for the action that is associated with the event rule.

Date and time values are specified in GMT time zone.

Table 49. Alert properties for correlations

CCMS alert property	Console property	Composer property
MTSYSID for event definition	MTE SAP System ID	InputSAPSystemID
MTMCNAME for event definition	MTE Monitoring Context Name	InputMonitoringContextName
OBJECTNAME for event definition	MTE Monitoring Object Name	InputMonObjectName
FIELDNAME for event definition	MTE Monitoring Attribute Name	InputMonFieldName
ALERTDATE	Alert Date	AlertDate
ALERTTIME	Alert Time	AlertTime
ALINDEX	Alert Index	AlertIndex
ALSYSID	Alert SAP System ID	AlertSAPSystemID
ALUNIQNUM	Alert UID	AlertUID
EXTINDEX	Alert MT Ext Index	AlertMTExtIndex
FIELDNAME	Alert Monitoring Attribute Name	AlertMonFieldName
MANDT	Client	Client
MSCGLID	Alert Msg Container ID	AlertMsgContainerID
MSEGNAME	Alert MTE Segment	AlertMTESegment
MSG	Alert Message	AlertMessage

Table 49. Alert properties for correlations (continued)

CCMS alert property	Console property	Composer property
MSGCLASS	XMI Ext Company Name	XMIExtCompanyName
MSGID	XMI Log Msg ID	XMILogMsgID
MTCLASS	Alert MT Class	AlertMTClass
MTINDEX	Alert MT Index	AlertMTIndex
MTMCNAME	Alert Monitoring Context Name	AlertMTEContext
MTNUMRANGE	Alert MTE Range	AlertMTERange
MTSYSID	Alert MTE System	AlertMTESys
MTUID	Alert MT Type ID	AlertMTTypeID
OBJECTNAME	Alert Monitoring Object Name	AlertMonObjName
RC	Alert Return Code	AlertReturnCode
REPORTEDBY	Alert Reported By	AlertReportedBy
SEVERITY	Alert Severity	AlertSeverity
STATCHGBY	Alert Changed By	AlertChangedBy
STATCHGDAT	Alert Change Date	AlertChangeDate
STATCHGTIM	Alert Change Time	AlertChangeTime
STATUS	Alert Status	AlertStatus
USERID	User ID	UserID
VALUE	Alert Value	AlertValue

Getting alert status and committing alerts by an external task

Learn how to get CCMS alert status and commit CCMS alerts.

Refer to the following sections for details about:

- [Getting CCMS alert status on page 369](#)
- [Committing CCMS alerts on page 371](#)

Getting CCMS alert status

About this task

To get the current status of a CCMS alert from IBM Workload Scheduler, use the external task Get Information (GI). To replace the command arguments with the actual values, refer to the output returned by the event rule you defined. For details

about the correspondence between the CCMS properties and the Console and composer properties, see [Table 49: Alert properties for correlations on page 368](#).

From a command line, enter the following command:

Figure 31. Command syntax

```
►► -r3batch — -t GI — -c Agent_name — -- "-t GAS — -alsysid sap_system_ID — -msegname alert_mte_segment →
    ► — -aluniquum alert_UID — -alindex alert_index — -alertdate alert_date — -alerttime alert_time — " -►◄
```

Where:

-t GI

Identifier of the task to be performed, in this case GI (Get Information).

-c *Agent_name*

The name of the dynamic agent workstation, or the unique identifier for the extended agent workstation connected to the SAP system where the MTE for which the alert was raised is located. For information about retrieving the unique identifier for the extended agent workstation, see [UNIQUE_ID on page 177](#).

-t GAS

Identifier of the task to be performed, in this case GAS (Get Alert Status).

-alsysid *sap_system_ID*

Identifier of the SAP system where the MTE for which the alert was raised is located. If the name contains blanks, enclose it between single quotes.

-msegname *alert_monitoring_segment*

Name of the alert monitoring segment. You can specify from 1 to 40 characters.

-aluniquum *alert_UID*

Unique identifier of the alert, made up of 10 characters.

-alindex *alert_index*

Alert index, made up of 10 characters.

-alertdate *alert_date*

Date of the alert, in the format *yyyymmdd*.

-alerttime *alert_time*

Time of the alert, in the format *hhmmss*.

The following is an example of how to retrieve the current status of a CCMS alert:

```
r3batch -t GI -c horse10 -- " -t GAS -alsysid T10
-msegname SAP_CCMS_horse10_T10_00 -aluniquum 0017780869
-alindex 0000000104 -alertdate 20081007 -alerttime 040356"
```

You are returned the current status of the alert.

Committing CCMS alerts

About this task

The CCMS alerts that you defined as IBM Workload Scheduler events are not automatically committed after their processing. To commit an alert that was processed by IBM Workload Scheduler, use the external task Put Information (PI).

To replace the command arguments with the actual values, refer to the output returned by the event rule you defined. For details about the correspondence between the CCMS properties and the Console and composer properties, see [Table 49: Alert properties for correlations on page 368](#).

From a command line, enter the following command:

Figure 32. Command syntax

```
►► -r3batch — -t PI — -c — XA_Unique_ID — -" -t CA — -alsysid — sap_system_ID — -msegrname — alert_mte_segment →
    ► — -aluniquum — alert_UID — -alindex — alert_index — -alertdate — alert_date — -alerttime — alert_time — " —►
```

Where:

-t PI

Identifier of the task to be performed, in this case PI (Put Information).

-c *Agent_name*

The name of the dynamic agent workstation or the unique identifier for the extended agent workstation connected to the SAP system where the MTE for which the alert was raised is located. For information about retrieving the unique identifier for the extended agent workstation, see [UNIQUE_ID on page 177](#).

-t CA

Identifier of the task to be performed, in this case CA (Commit Alert).

-alsysid *sap_system_ID*

Identifier of the SAP system where the MTE for which the alert was raised is located. If the name contains blanks, enclose it between single quotes.

-msegrname *alert_monitoring_segment*

Name of the alert monitoring segment. You can specify from 1 to 40 characters.

-aluniquum *alert_UID*

Unique identifier of the alert, made up of 10 characters.

-alindex *alert_index*

Alert index, made up of 10 characters.

-alertdate *alert_date*

Date of the alert, in the format *yyyymmdd*.

-alerttime alert_time

Time of the alert, in the format *hhmmss*.

The following is an example of how to commit a CCMS alert:

```
r3batch -t PI -c horse10 -- " -t CA -alsysid T10
-msegrname SAP_CCMS_horse10_T10_00 -aluniquum 0017780869
-alindex 0000000104 -alertdate 20081007 -alerttime 040356"
```

You are returned with the message `The CCMS alert was successfully confirmed.`

Example of an event rule based on CCMS alerts

The following example shows an event rule defined to monitor the yellow alerts raised on the MTE named `GS0\ALE/EDI` `GS0(000)` `Log.sys` `TVALE\Inbound` `IDoc` `ORDER_IDOC\Inbound: IDoc generated`. The MTE is configured to generate a yellow alert when the number of IDocs representing orders ready to process exceeds a specified threshold. If this condition occurs, the following actions are triggered:

- An IBM Workload Scheduler job is submitted to process the order IDocs.
- An IBM Workload Scheduler job, with priority 10, is submitted to confirm the alert.

```
<?xml version="1.0" encoding="UTF-8"?>
<eventRuleSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://www.abc.com/xmlns/prod/tws/1.0/event-management/rules"
  xsi:schemaLocation="http://www.abc.com/xmlns/prod/tws/1.0/event-management/rules
    http://www.abc.com/xmlns/prod/tws/1.0/event-management/rules/EventRules.xsd">
  <eventRule name="SCENARIO1_XAL" ruleType="filter" isDraft="yes">
    <eventCondition name="MTEEventRaised1" eventProvider="SapMonitor"
      eventType="CCMSEventGenerated">
      <filteringPredicate>
        <attributeFilter name="Workstation" operator="eq"
          <value>SAP_XA</value>
        </attributeFilter>
        <attributeFilter name="InputSAPSystemID" operator="eq"
          <value>GS0</value>
        </attributeFilter>
        <attributeFilter name="InputMonitoringContextName" operator="eq"
          <value>ALE/EDI GS0(000) Log.sys TVALE</value>
        </attributeFilter>
        <attributeFilter name="InputMonObjectName" operator="eq"
          <value>Inbound IDoc ORDER_IDOC</value>
        </attributeFilter>
        <attributeFilter name="InputMonFieldName" operator="eq"
          <value>Inbound: IDoc generated</value>
        </attributeFilter>
      </filteringPredicate>
    </eventCondition>
    <action actionProvider="TWSAction" actionType="sbj" responseType="onDetection">
      <parameter name="JobUseUniqueAlias">
        <value>>false</value>
      </parameter>
      <parameter name="JobDefinitionWorkstationName">
        <value>SAP_XA</value>
      </parameter>
    </action>
  </eventRule>
</eventRuleSet>
```

```

        <parameter name="JobAlias">
            <value>IDOC_{MTEEventRaised1.AlertUID}/value>
        </parameter>
        <parameter name="JobDefinitionName">
            <value>PROCESS_ORDER/value>
        </parameter>
    </action>
    <action actionProvider="TWSAction" actionType="sbd" responseType="onDetection">
        <parameter name="JobUseUniqueAlias">
            <value>>false/value>
        </parameter>
        <parameter name="JobWorkstationName">
            <value>TWS_HOST_FTA/value>
        </parameter>
        <parameter name="JobTask">
            <value>C:\TWA_home\methods\r3batch -t PI
                -c {MTEEventRaised1.Workstation} -- "
                -t CA -ALSYID {MTEEventRaised1.AlertSAPSystemID}
                -MSEGNAM {MTEEventRaised1.AlertMTESegment}
                -ALUNIQNUM {MTEEventRaised1.AlertUID}
                -ALINDEX {MTEEventRaised1.AlertIndex}
                -ALERTDATE {MTEEventRaised1.AlertDate}
                -ALERTTIME {MTEEventRaised1.AlertTime} "
            </value>
        </parameter>
        <parameter name="JobPriority">
            <value>10</value>
        </parameter>
        <parameter name="JobType">
            <value>Command</value>
        </parameter>
        <parameter name="JobAlias">
            <value>CONFIRM_{MTEEventRaised1.AlertUID}</value>
        </parameter>
        <parameter name="JobStreamName">
            <value>CONFIRM_STREAM</value>
        </parameter>
        <parameter name="JobLogin">
            <value>twuser</value>
        </parameter>
    </action>
</eventRule>
</eventRuleSet>

```

National Language support

The National Language support feature allows you to install SAP on a localized IBM Workload Scheduler workstation and use localized characters for IBM Workload Scheduler job names, job streams, and SAP variants.

Using the local and global configuration files, you can set up SAP to use different code pages and languages for both its output and its connection with a remote SAP system.

As described in [Unicode support on page 227](#), this version of Access method for SAP features Unicode, which is widely supported by SAP systems since version 4.7. However, if either the workstation running SAP or the target SAP systems do not support Unicode, this section describes how you configure code pages and national languages for SAP.

Setting National Language support options

The following options control the code page and national language used by Access method for SAP, when Unicode support is not used:

TWSXA_CP

The code page used to establish the connection between `r3batch` and the target SAP system.

If you are running a non-Unicode version of `r3batch`, set this option to the code page installed on the SAP system (for a list of the valid code pages, refer to [SAP supported code pages on page 375](#)). The default value is the SAP code page 1100, similar to the standard ISO8859-1. In all other cases, this option is ignored.

TWSXA_LANG

The language that `r3batch` uses to log in. It can be one of the following (DE, EN, and JA can be set from the Option Editor. The other languages can be set using any text editor):

- Brazilian Portuguese (pt_BR)
- English (EN, the default value)
- French (FR)
- German (DE)
- Italian (IT)
- Japanese (JA)
- Korean (KO)
- Simplified Chinese (zh_CN)
- Spanish (ES)
- Traditional Chinese (zh_TW)



Note: If you are working with InfoPackages and process chains on operating systems that do not support Unicode, this option must be set.

TWSMETH_CP

The code page that `r3batch` uses for its output. The default is the code page used by the IBM Workload Scheduler workstation that hosts `r3batch`.

Ensure that the `TWSMETH_CP` and `TWSMETH_LANG` options are consistent.

TWSMETH_LANG

The catalog language used by `r3batch`. The default is the language used by the IBM Workload Scheduler workstation that hosts `r3batch`.

Ensure that the `TWSMETH_CP` and `TWSMETH_LANG` options are consistent.

SAP supported code pages

To communicate with SAP systems, Access method for SAP uses the following code pages. Use these values to set option `TWSXA_CP`, only when `r3batch` does not support Unicode.

Table 50. SAP supported code pages

SAP code pages	Description
1100	8859-1, this is the default value
1103	MS 850
8000	SJIS: Shift JIS
8300	BIG5: Traditional Chinese
8400	GBK: Simplified Chinese

Troubleshooting

Learn what to do if you get any problems while installing or using IBM Workload Scheduler access methods or plug-ins.

Troubleshooting the SAP connection

If you are unable to submit SAP jobs using IBM Workload Scheduler after the R/3 configuration, perform the following tests:

- Ensure that you can ping the SAP system from the IBM Workload Scheduler system. This shows basic network connectivity.
- Note that using the SAP routers to access the R/3 system could exceed the size of internal buffers of the RFC library used to store the hostname of the SAP system. When this occurs, the hostname gets truncated, causing the connection to the R/3 system to fail. To work around this problem, do not fully qualify the name of the SAP routers or alternatively use the IP addresses.
- Run the following **telnet** command to verify connectivity:

```
telnet systemname 33xx
```

where *systemname* is the system name or IP address of the SAP server and *xx* is the SAP instance.

If the command fails to complete, this means that communication between `r3batch` and the SAP application server is down.

- Log on to the SAP system as an administrator and verify that the IBM Workload Scheduler RFC user (created in the [Creating the IBM Workload Scheduler RFC user on page 217](#)) exists.
- If the SAP gateway truncates the connection string, replace the host name with the IP address.
- If `r3batch` runs on an AIX® system that does not use U.S. English, make sure that the U.S. Language Environment® is installed on both the IBM Workload Scheduler workstation and the SAP database workstation. Otherwise the error `BAD TEXTENV` (or a similar error message) might appear in the `dev_rfc` trace file and connections to SAP fail.

Other known problems

[Table 51: Miscellaneous troubleshooting items on page 376](#) lists miscellaneous troubleshooting problems.

Table 51. Miscellaneous troubleshooting items

Area	Item
r3batch and r3event: output contains unreadable characters	Symptom: When you enter the r3batch and r3event commands interactively (for example, to export an SAP calendar) the output is returned in UTF-8 format.
	Solution: To resolve this problem, you can either use a shell that supports the UTF-8 code page or redirect the output to a file and open it with a text editor that supports the UTF-8 format.
r3batch: SAP jobs contain quotation marks (") or reverse quotes (`)	Symptoms: SAP jobs whose names contain quotation marks or reverse quotes are not displayed in the pick list of the Dynamic Workload Console. -OR- You have an IBM Workload Scheduler job that tries to submit an SAP job whose name contains quotation marks, but it abends with an error. The following message might be displayed: <pre>EEW00439E The required options are not specified either in the global or in the local options file.</pre>
	Solution: In your SAP system, make a copy of the SAP job and assign it a name that does not contain quotation marks or reverse quotes.
r3batch: SAP job containing Arabic characters.	Symptom: An SAP job abends when the job contains Arabic characters.
	Solution: If you run an SAP job that contains Arabic characters, you must set the local codepage of the agent workstation hosting the r3batch access method to the Arabic codepage. Refer to the twsmeth_cp keyword in the common options file, Defining the common options on page 237 .
r3batch: error messages submitting a job on dynamic agents.	Symptom: When working with dynamic workstations and performing actions such as: displaying a process chain, restarting a process chain, or retrieving the spool list, the following messages might be displayed from the Dynamic Workload Console: <pre>EEW00439E The required options are not specified either in the global or in the local options file.</pre> <pre>EEW01065W The environment variable UNISON_JOB is not set. The process chain cannot be restarted.</pre>
	Solution: These messages might indicate that the requested action is not supported on dynamic workstations. Refer to the <i>IBM Workload Scheduler Release Notes®</i> for more information about IBM Workload Scheduler features and minimum required versions for compatibility.

Table 51. Miscellaneous troubleshooting items

(continued)

Area	Item
r3batch: r3batch hangs when performing actions from the Dynamic Workload Console.	Symptoms: r3batch hangs when performing actions from the Dynamic Workload Console such as selecting from a pick list, submitting a job, or similar actions that require connection to the SAP system. The IBM Workload Scheduler joblog might also contain multiple "Timer expired" messages.
	Solution: This problem is caused by the IBM Workload Scheduler logging and tracing component. There are two possible solutions: • Deactivate the tracing utility as described in the following technote: http://www.ibm.com/support/docview.wss?uid=swg21503284. - OR • Modify the <code>r3batch.properties</code> files. Locate the <code>ther3batch.trace.handlers.traceFile.MPFileSemKeyproperties</code> setting, and then either comment this property setting out or use a different value. Choose any numeric value and retry the operation.
r3batch: Submit same process chain in parallel fails.	Symptom: The SAP system returns an error message RFC_ERROR_SYSTEM_FAILURE when starting an SAP process chain.
	Solution: Verify if the corrections stated in SAP note 1723482 are applied to your SAP Business Warehouse system or avoid running the same process chain more than once simultaneously.
r3batch: When you restart the process of a subchain, the status of the original process chain is not changed to active	Symptom: When you restart the process of a subchain, the status of the original process chain is not changed to active.
	Solution: Refer to SAP Note 1075876.
r3batch: Refresh an SAP process chain after a kill action on a running job instance.	Symptom: If you perform a kill action on an IBM Workload Scheduler job instance running on a dynamic workstation which monitors an SAP process chain, and then subsequently perform a Refresh operation on this job, the job fails.
	Solution: You cannot perform a Refresh operation after having performed a kill action on an IBM Workload Scheduler job instance running on a dynamic workstation which monitors an SAP process chain. Verify the status of the SAP process chain on the SAP system, and then set the IBM Workload Scheduler job status accordingly.

Table 51. Miscellaneous troubleshooting items

(continued)

Area	Item
r3batch: Wrong characters are displayed in the Criteria Manager profile description.	Symptom: Special characters such as, < (less than), > (greater than), or the ' (apostrophe) specified in the Description field of the Create criteria profile dialog are displayed incorrectly. Solution: Avoid using special characters in the Description field when creating a new criteria profile.
r3evmon: monitoring events is not started, stopped, or performed	Symptom: You cannot start or stop event monitoring, or event monitoring is not performed. Solution: Ensure that <i>TWSuser</i> is the owner of the following files, and that the user has read and write permissions: • <code>/TWA_DATA_DIR/pids/XAname_r3evmon.pid</code> • <code>/TWA_DATA_DIR/EIF/XAname_r3evmoncache.dat</code> • <code>/TWA_DATA_DIR/EIF/XAname_r3evmoneif.conf</code> • <code>/TWA_DATA_DIR/methods/r3evmon_cfg/XAname_r3evmon.cfg</code> • <code>/TWA_DATA_DIR/methods/r3evmon_cfg/XAname_r3idocmon.cfg</code> • <code>/TWA_DATA_DIR/methods/r3evmon_cfg/XAname_r3xalmon.cfg</code> • <code>/TWA_DATA_DIR/methods/r3evmon_cfg/XAname_r3evmon.lck</code> On Windows™ workstations, these files are located in the <i>TWA_home</i> directory and not in the <i>TWA_DATA_DIR</i> directory.
r3batch: monitoring SAP events is not performed	Symptom: The SAP event on which the event rule is based is neither monitored nor committed. Solution: Ensure that the extended agent workstation you specified in the SAP event definition exists. When you define an SAP event within an event rule, no check on the extended agent workstation is made: if the workstation does not exist, the event rule is saved and activated but it will never be resolved.
r3batch: monitoring SAP events is not performed	Symptom: With XBP 3.0, the SAP event is raised but IBM Workload Scheduler is not notified and therefore does not act as expected. Solution: Ensure that the SAP event was not excluded from logging in the SAP event history table.
r3batch: monitoring SAP events is not performed	Symptom: The SAP events on which the event rule is based are not monitored nor committed. Solution: The SAP events being monitored are listed in the following file: <pre>TWA_DATA_DIR/monconf/XAname_r3evmon.cfg</pre> where <i>XAname</i> is the name of the SAP extended agent workstation.

Table 51. Miscellaneous troubleshooting items

(continued)

Area	Item
	Check that the file is updated and contains the current monitoring plan. The SAP events are indicated by the following keyword (one for each SAP event on the same extended agent): <pre data-bbox="407 485 1459 516">!R3EVENT SAP_event_name_lengthSAP_event_name[SAP_event_parm_lengthSAP_event_parm]</pre> where: <i>SAP_event_name_length</i> The length of the SAP event name to monitor, in the format <i>nnnn</i>. For example, 0008, if the event name is SAP_TEST. <i>SAP_event_name</i> The name of the SAP event to monitor. <i>SAP_event_parm_length</i> The length of the parameter associated with the SAP event to monitor, if any. The format is <i>nnnn</i>. For example, 0007, if the event name is SAP_PAR. <i>SAP_event_parm</i> The parameter associated with the SAP event to monitor, if any. This value is optional, but omitting it identifies an SAP event with no parameter associated. For details about how the events are matched between <code>r3evmon.cfg</code> and the SAP system, see SAP events matching criteria on page 348. For each configuration file, an <code>r3evmon</code> process is started to monitor the SAP events listed. To start an <code>r3evmon</code> monitoring process for a specific extended agent workstation, enter the following command.  Note: 1. For UNIX® only, <code>r3evmon</code> must be entered by the owner of the IBM Workload Scheduler installation: 2. If you run <code>r3evmon</code> from a Windows™ DOS shell, the command prompt is not returned until the process completes. <pre data-bbox="407 1671 1459 1703">r3evmon -t SEM -c XA_Unique_ID -- "[-EIFSRV EIF_server -EIFPORT EIF_port]"</pre> where:

Table 51. Miscellaneous troubleshooting items

(continued)

Area	Item
	<i>XA_Unique_ID</i> The unique identifier of the extended agent workstation. For information about retrieving the unique identifier for the extended agent workstation, see UNIQUE_ID on page 177. <i>EIF_server</i> The host name or IP address of the master domain manager. <i>EIF_port</i> The port that the master domain manager uses to receive the event notification.
r3batch:IDoc monitoring is not performed	Symptom: The events on which the event rule is based are not monitored or no event is generated during IDoc monitoring. Solution: The events being monitored are listed in the following file: <pre>TWA_DATA_DIR/monconf/XAname_r3evmon.cfg</pre> where <i>XAname</i> is the name of the SAP extended agent workstation. It is the same file that is used to monitor SAP events in general. Check that the file is updated and contains the current monitoring plan. The events corresponding to the <code>IDOCEventGenerated</code> event type are indicated by the following keyword (one for each event on the same extended agent): <pre>!IDOC nnnn<Client Number>nnnn<IDoc Status List>nnnn<Direction>nnnn<Receiver Port> nnnn<Receiver Partner Type>nnnn<Partner Function of Receiver> nnnn<Partner Number of Receiver>nnnn<Sender Port>nnnn<Sender Partner Type> nnnn<Partner Function of Sender>nnnn<Partner Number of Sender> nnnn<Message Type>nnnn<IDoc Type>nnnn<Logical Message Variant> nnnn<Logical Message Function>nnnn<Test Flag>nnnn<Output Mode></pre> where: <i>nnnn</i> The length of the IDoc field. For example, <code>0005</code> indicates the value of an IDoc status list corresponding to <code>56, 60</code>. <i>< ></i> Contains the value of the field associated with the IDoc to be monitored. For a list of the supported IDoc fields, refer to Table 40: IBM Workload Scheduler fields used to define event rules based on IDocs on page 353.

Table 51. Miscellaneous troubleshooting items

(continued)

Area	Item
	For each configuration file, an <code>r3evmon</code> process is started to monitor the events listed. Make sure that an <code>r3evmon</code> monitoring process is started for the involved extended agent workstation.
<code>r3evmon</code> : monitoring SAP and IDoc events increases memory consumption	Symptom: Memory consumption increases continuously during monitoring of IDoc and standard SAP events.
	Solution: Refer to SAP Notes® 1021071 and 1109413.
<code>r3batch</code> : Duplicated events generated during IDoc monitoring	Symptom: The action defined in an event rule with <code>IDocEventGenerated</code> event type is unexpectedly repeated.
	Solution: Reset the start date and time for the next monitoring loop. These values are stored in the following file: <pre><data_dir>/methods/r3evmon_cfg/XAname_r3idocmon.cfg</pre> where <code>XAname</code> is the name of the SAP extended agent workstation. Therefore you can either: • Stop <code>r3evmon</code>, delete the <code>XAname_r3idocmon.cfg</code> file and then start <code>r3evmon</code> again. - OR - • Stop <code>r3evmon</code>, set the date and time in the <code>XAname_r3idocmon.cfg</code> file to the values you want, and start <code>r3evmon</code> again. Use the following format for the start date and time: <pre>start_date=YYYYMMDD start_time=HHMMSS</pre> For example: <pre>start_date=20080307 start_time=115749</pre> Check the value of the idoc_no_history option: • If it is set to OFF and no <code>XAname_r3idocmon.cfg</code> file exists, then all matching IDocs are retrieved, not only the current ones. • If it is set to ON (default value), check the date and time in the <code>XAname_r3idocmon.cfg</code> file.
<code>r3batch</code> : No event is generated during IDoc monitoring	Symptom: The expected event actions are not triggered.
	Solution: Check the value of the idoc_no_history option; if it is set to ON (default value), check the date and time in the <code>XAname_r3idocmon.cfg</code> file.

Table 51. Miscellaneous troubleshooting items

(continued)

Area	Item
Error defining an internetwork dependency based on SAP event	Symptom: If you work with XBP 2.0, when you try to define an internetwork dependency based on an SAP event, the following error message is displayed: <pre data-bbox="402 491 1349 632">*** ERROR 778 *** EEW00778E An internal error has occurred. The program could not modify the following job: Job name: Job ID: %CJ ERROR</pre> Solution: Perform the following steps: 1. Check if the BCTEST report is defined in your SAP system by invoking either one of the following transactions: sa38 Enter BTC* and click the picklist button. In the panel that opens, click the picklist button and check if BTCTEST is shown in the list that is displayed. se38 Enter BTC* and click the picklist button. Check if BTCTEST is shown in the list that is displayed. 2. If report BTCTEST is not found in the list, you can either: ◦ Choose another existing report, and, in the local options file, set the placeholder_abap_step option to the name you chose. Because the report assigned to the placeholder job is run when the corresponding event is raised, ensure that you choose a dummy report. For details about the placeholder_abap_step option, see Table 24: r3batch common configuration options on page 237. - OR - ◦ Set the placeholder_abap_step option to a custom developed ABAP code of your choice.
r3batch: error message when scheduling SAP jobs	Symptom: When creating an SAP job, the following message is displayed while trying to view the details of an ABAP's variant: <pre data-bbox="402 1629 1333 1709">AWS00101E Missing ABAP routine. J_101_REPORT_ALL_SELECTIONS please install the latest ABAP routine for Maestro!!</pre> Solution: This defect is caused by an error in an SAP function module. SAP describes this problem and possible solutions in the SAP Notes® 0351293 and 0116354.

Table 51. Miscellaneous troubleshooting items

(continued)

Area	Item
r3batch: modify job step error	You change print parameters with the <code>BAPI_XBP_MODIFY_JOB_STEP</code> function module, and subsequently, they are incorrect. As a consequence, r3batch gets error 221: <pre>MSG_CANNOT_GET_PRIARC_PARAMS: "Retrieving new print and archive parameters failed"</pre> The problem is solved by installing SAP Note 758829.
r3batch: modify job step error	The <code>BAPI_XBP_MODIFY_JOB_STEP</code> function module always uses the name of the logged-on user as the name for the step user. In this case, when submitting a job with the <code>-vx</code> options, r3batch creates a job by copying all the data from the original template, except the variant name of the first step (which is provided as the option parameter). This procedure is usually referred to as the "old copy". However, when adding a step to a new job, the XBP 2.0 interface ignores the user parameter passed by r3batch. The problem is solved by installing SAP note 758829.
r3batch: does not start after installation on Windows™	Symptom: After installing or upgrading the SAP R/3 access method to version 8.5 on a Windows™ operating system, you try to start r3batch but nothing happens. The following message is displayed: <pre>The application failed to initialize properly. Click on OK to terminate the application.</pre> Solution: Ensure that you applied the SAP Note 684106 to install the required Microsoft™ DLLs.
r3batch: IBM Workload Scheduler environment variables are not resolved when specified in the task string for an R/3 batch job.	Symptom: When IBM Workload Scheduler environment variables are used in the task string for an R/3 batch job and the job is launched, the environment variables are not resolved. The exact string used to specify the variable is used instead. Solution: To leverage IBM Workload Scheduler environment variables, you must modify the access method as follows: - In the <code>TWA_DATA_DIR/methods</code> directory, create a file named, r3batch.cmd (on Windows™) or r3batch.sh (on UNIX®) as required, containing the following content: <pre>@echo off set METHODSPATH=%~dp0 call "%METHODSPATH:="%r3batch.exe" %*</pre> - Modify the CPU XAGENT definition from r3batch to r3batch.cmd. An example follows: <pre>CPUNAME NW1 DESCRIPTION "r3batch" OS OTHER NODE none TCPADDR 31111 FOR MAESTRO HOST STROMBOLI ACCESS "r3batch.cmd" TYPE X-AGENT AUTOLINK OFF BEHINDFIREWALL OFF</pre>

Table 51. Miscellaneous troubleshooting items

(continued)

Area	Item
	<pre>FULLSTATUS OFF END</pre> 3. To modify the CPU access method in the Symphony® file, run JnextPlan as follows: <pre>JnextPlan -for 0000</pre>
@longlink file present in installation directory	Symptom: After installing IBM® Workload Scheduler on a computer with an AIX® operating system where a master domain manager is already installed, a @longlink file containing the following is present in the installation directory: <pre>methods/_tools/_jvm_64/lib/desktop/icons/HighContrastInverse/48x48/mimetypes/ gnome-mime-text-x-java.png</pre> Solution: The file can be ignored. It does not present any problems for the proper functioning of the product.
Job throttling does not start	Symptom: When you start the job throttling feature, nothing happens and the following error message is displayed: <pre>EEW0JTR0207E Error, another job throttler instance is already running against the same SAP system. Foreign job throttler registration is: Client ID="clientID", Name="TWS4APPS_JOBTHROTTLER",Host="hostname", UID "UniqueID"</pre> Cause and Solution: Possible causes are: • You are running job interception collector jobs, but the job interception and job throttling features cannot run at the same time. Choose which feature to start. For detailed information, refer to Job interception and parent-child features on page 305 and Job throttling feature on page 333. • Another job throttler instance is running against the same SAP system. You can start only one job throttler instance. • A previous job throttler instance created an exclusive lock object on the SAP system that could have become permanent. To verify it, use transaction <code>sm12</code> and query for the lock object named <code>TWS4APPS_JOBTHROTTLER</code>. If the lock object exists, and you are not running any job throttler or job interception instance, remove the lock manually and restart the job throttler.
Job throttling does not start	Symptom: When you start the job throttling feature, nothing happens and the following error message is displayed: <pre>EEW0JT0209E Error, the password format is not valid.</pre> Cause and Solution: Your password is encrypted in old format. To encrypt the password with the correct encryption version, use the enigma or pwdcrypt programs. For details about how to encrypt the password, see Encrypting SAP user passwords on page 249.

Table 51. Miscellaneous troubleshooting items

(continued)

Area	Item
Job throttling does not stop	Symptom: When you stop the job throttling feature, nothing happens.
	Cause and Solution: You are connected as a <i>TWSUser</i> who does not have write permission on the <code>XAname_jobthrottling_cmd.txt</code> file. To solve this problem, delete the <code>XAname_jobthrottling_cmd.txt</code> file and enter the command again. For detailed information about stopping the job throttler, refer to Step 5. Starting and stopping the job throttling feature on page 336 .
Job throttling: alerts for MTEs are not generated according to the threshold values set	Symptom: Alerts for the MTEs created by the job throttler are generated without respecting the threshold values that are set.
	Cause and Solution: You started a new job throttler instance, which, being enabled to send data to CCMS, created the related MTEs. When you include the MTEs in your monitoring set, the threshold values are automatically set according to the existing MTE class. Nevertheless, alerts are generated without respecting these values. To solve this problem, edit the MTE properties and save them again, even if you do not change anything.
Job throttling: saving MTE properties generates an informational message	Symptom: When you edit and save the properties of MTEs generated by the job throttler, the following informational message is displayed:
	Message does not exist. Cause and Solution: In the pop-up window that displays the message, click Continue and close the Properties window. Your settings are saved.
The system cannot intercept jobs	Symptom: Although the job interception feature is active on the SAP system, the intercepted jobs are kept in scheduled state.
	Cause and Solution: The job throttler feature or the Java™ Virtual Machine used by the job throttler might still be active. On each extended agent where the job throttler was started at least once, ensure that: 1. You stopped the feature. For details, see Step 5. Starting and stopping the job throttling feature on page 336. 2. The Java™ Virtual Machine used by the job throttler was stopped by the process. To search for Java™ processes, use: On Windows™ The Process Explorer

Table 51. Miscellaneous troubleshooting items

(continued)

Area	Item
	On UNIX® The command <code>ps -ef grep throttling</code> If a Java™ Virtual Machine instance related to the job throttler is found, kill it.
access method executables: <code>r3batch, r3event,</code> psagent: permission denied messages in the job log.	Symptom: The job log reports multiple "Permission denied" messages. Cause and Solution: The root cause might be that the access method executable, for example, <code>r3batch</code>, is submitted by the <code>root</code> user and not the <code>twuser</code>. This creates directories and files with the wrong ownership and file permissions. Verify the ownership of the following directories and files if you are running the product on UNIX® platforms. Ensure that the <code>twuser</code> is the owner of the files and that the user has both read and write permissions on the files, and execute permission on the directories. <pre>TWA_DATA_DIR/methods/traces TWA_DATA_DIR/methods/traces/*.log</pre>
psagent: misleading message displayed if the local options file has no right permissions	Symptom: The job log shows the following message: <pre>EEW00439E You did not specify the required options either in the global or in the local options file.</pre> but all the mandatory options were correctly set in the options file. Solution: Check that the options file has <code>read</code> and <code>write</code> permissions available to the user who is trying to launch the job.
No messages written in the job log	Symptom: IBM Workload Scheduler does not write any messages in the job log if the file system for tracing is full or the <code>ljuser</code> does not have the correct permission to write in the trace directory.
The submission of a PeopleSoft job fails	Symptom: The submission of a PeopleSoft job fails and the IBM Workload Scheduler job log contains a Java™ exception similar to the following: <pre>Exception in thread "3194" java.lang.ExceptionInInitializerError at bea.jolt.JoltSessionAttributes.<clinit>(JoltSessionAttributes.java:183) at psft.pt8.net.JoltSessionPool.createConnection(JoltSessionPool.java:363) at psft.pt8.net.JoltSessionPool.getJoltSession(JoltSessionPool.java:220)</pre> Cause and Solution: The <code>psjoe.jar</code> path contains special characters. Define a path without special characters.
The submission of an Oracle job fails	Symptom: The submission of an Oracle job fails and the IBM Workload Scheduler job log shows the following information:

Table 51. Miscellaneous troubleshooting items

(continued)

Area	Item
	EEWP0017 Child MCMLJ exited normally. Exit code: 1.EEWP0027 Error - Launch job failed Solution: Submitting an Oracle job might fail because there is a connection problem to the Oracle database. Verify that your Oracle naming methods are set correctly. For details about how to configure naming methods, refer to the <i>Oracle Net Services Administrator's Guide</i>.
To upgrade the SAP environment, perform the following steps:	1. Delete the TWS ABAP module. 2. Upgrade SAP. 3. Install TWS ABAP module.

Scheduling jobs on IBM Workload Scheduler from SAP Solution Manager

IBM Workload Scheduler and SAP Solution Manager are integrated to allow the IBM Workload Scheduler engine to run the job scheduling tasks available from the Solution Manager user interface.

The integration is provided by the SMSE Adapter, which runs on the master domain manager. The SMSE Adapter uses the SAP Solution Manager Scheduling Enabler (SMSE) interface provided by SAP to enable *external schedulers* to run the scheduling for Solution Manager.

With this integration, when you schedule a job from the **Scheduling** panel of Solution Manager, IBM Workload Scheduler takes charge of the job scheduling, monitoring, and management tasks, as well as of job triggering and notification.

Under these conditions IBM Workload Scheduler acts as an RFC-Server with a common interface for scheduling jobs. It is identified through an RFC-Destination, registered in the SMSE. The interaction between Solution Manager and IBM Workload Scheduler is based on a PUSH mechanism implemented by the SMSE interface, whereby the master domain manager responds to requests solicited by the Solution Manager job scheduling functions.

Qualified as *external scheduler* by Solution Manager, the registered master domain managers, identified by their RFC destination names, can be called or administered from the **Process Scheduling Adapter** menu item in the Solution Manager GUI.

The jobs scheduled from Solution Manager on IBM Workload Scheduler must have been previously defined in the IBM Workload Scheduler database.

A job scheduled from the **Schedule Jobs** or **Job Documentation** panels in Solution Manager to be run by IBM Workload Scheduler, is automatically mapped in a job stream that is expressly created to include the job.

Registering the master domain manager on SAP Solution Manager

The first step to run the integration is to register the master domain manager on the SAP Solution Manager system.

To register master domain manager on the SAP Solution Manager system, you must:

1. Have established a connection based on RFC or Web Services between the master and the Solution Manager system.
2. Have the SAP JCo 3.1 Patch 3 (sapjco31P_3) libraries (jar files and, according to your operating system, dll, so, or sl) installed in the `<data_dir>/methods/smseadapter/lib` directory on the master domain manager. To download 3.1 Patch 3 (sapjco31P_3), visit the Sap Service Marketplace.



Attention: The libraries require the Microsoft Visual C++ Redistributable Package (vcredist) installed.

3. Configure the `smseadapter.properties` file located in the `<data_dir>/methods/smseadapter/lib` directory on the master.

The file contains a `SMSE_ADAPTER_CONNECTION_n` section that can be duplicated depending on the number of connections that you want to define. You can in fact set more connection definitions for the same master, where, for example, the following can vary:

- The SAP Solution Manager system.
- The agent that is to run the workload.
- The SAP user name.



Note: A master domain manager can have only one active connection at a time via the `smseadpter`. If the adapter finds more than one section with the `startAdapter` property set to `true` (or not set to `false`), it uses the first section of properties and ignores the others.

4. Stop and start WebSphere Application Server Liberty Base. For further information, see Application server - starting and stopping.

The following is an example of `smseadapter.properties` file:

```
[SMSE_ADAPTER_CONNECTION_1]
startAdapter =
ashost =
sysnr =
client =
sid =
user =
passwd =
lang =
destination =
setDestinationAsDefault =
jobStreamNamePrefix =
agentName =
notificationThreadCheckInterval =
adminConsoleHost =
adminConsolePort =
adminConsoleUser =
adminConsoleUserPassword =
maxRegistrationAttempts =
registrationAttemptInterval =
```

This section can be repeated as many times as needed in the `smseadapter.properties` file.

The properties are:

Table 52. Properties for the `smseadapter.properties` file.

Property	Description	Required	Notes
SMSE_ADAPTER_CONNECTION_1	This is the section header. If you have more sections the last digit should differ from one section and another. If two sections contain identical property values, only the first section read is considered, the other is ignored.	✓	
startAdapter	Specifies whether to connect or not to SAP Solution Manager. Can be true or false. Must be set to true to make the connection work. Set to false to temporarily suspend the connection.	✓	The default is true.
ashost	The host name of the SAP Solution Manager server on which the master domain manager registers. For example, / H/7.142.153.8/H/7.142.154.114.	✓	The master domain manager can connect to one Solution Manager system at a time.
sysnr	The SAP system number of the system that the master registers on. This value must have two digits. For example, 00.	✓	
client	The SAP client number. For example, 001.	✓	
sid	The SAP system identifier (SID) that the master registers on. For example, SM1.	✓	
user	The SAP user name that will be used during the notification process to log into SAP Solution Manager. For example, twsadmin.	✓	
passwd	The SAP password that will be used during the notification process to log into SAP Solution Manager. You can enter it in clear or in encrypted forms.	✓	To encrypt the password use the <code>enigma</code> program located in the <code>methods</code> folder on the master.
lang	The SAP logon language. For example, EN.	✓	
destination	A name entered here to identify the RFC Destination that will be used to connect	✓	This name defines the logical connection between the Solution Manager system

Table 52. Properties for the `smseadapter.properties` file.

(continued)

Property	Description	Required	Notes
	to SAP Solution Manager. For example, IWSM2.		and the master domain manager, referred to in Solution Manager as the <code>external scheduler</code> . The complete destination name will then be formed by: <code>destination@mdm_name</code> For example: IWSM2@MAS93WIN
	 Note: The destination name must be univocal.		 Note: The destination name must be univocal.
<code>setDestinationAsDefault</code>	Set to <code>true</code> to make this destination the default one. The default is <code>false</code> .		Use this property in a context where a Solution Manager system has more than one active destination defined (that is, more registered masters), to set the default external scheduler. If you do not set a default, and you have more external schedulers registered on an SM system, you will have to specify the destination at scheduling time.
<code>jobStreamNamePrefix</code>	A prefix of at least four letters that is to be added to the names of the job streams created when jobs are submitted. The first character must be a letter while the remaining characters can be alphanumeric.		The default prefix is SOLMAN.
<code>agentName</code>	The name of the IWS agent that will run the jobs. When you search for the job definition in the Scheduling dialog, the Search utility returns the names of the jobs defined to run on this agent.		If no agent name is specified, the Search utility returns the names of the jobs defined to run on all the agents attached to the master domain manager (unless you use filtering).
<code>notificationThreadCheckInterval</code>	The time interval, in seconds, between checks made by the notification thread on the status changes of a job. The default is 5 seconds.		The thread notifies Solution Manager with the status changes of a job.
<code>adminConsoleURL</code>	The protocol used (http or https) and the host name and port of the Dynamic Workload Console		The next four properties, all related to the Dynamic Workload Console, are optional, but if you specify one, you must specify all.

Table 52. Properties for the `smseadapter.properties` file.

(continued)

Property	Description	Required	Notes
	attached to the master. For example, <code>https://mydwc:port_number/abc/console.</code>		
<code>adminConsoleUser</code>	The username that logs onto the Dynamic Workload Console attached to the master.		
<code>adminConsoleUserPassword</code>	The password of the username that logs onto the Dynamic Workload Console attached to the master.		
<code>maxRegistrationAttempts</code>	The maximum number of attempts to connect to SAP. By default, it is set to 5 times.		
<code>registrationAttemptInterval</code>	The time after which a new attempt to connect to SAP is performed. By default, it is set to 5 seconds.		



Note: If the language configured for the master domain manager is different from the language configured for the Solution Manager system, the messages issued in the Solution Manager user interface may be displayed in mixed languages.

Scheduling

The `Job Management Administration` panel of Solution Manager has two entry points for scheduling jobs:

- The `Schedule Jobs` item in `Common Tasks`, a direct way of scheduling, where you pick the job from the IBM Workload Scheduler database and you set the scheduling options and time definitions.
- The `Job Documentation` object, where you can create and edit job documentation, schedule, monitor, and manage jobs.

The jobs scheduled from Solution Manager on IBM Workload Scheduler must have been previously defined in the IBM Workload Scheduler database.

A job scheduled from the `Schedule Jobs` or `Job Documentation` panels in Solution Manager to be run by IBM Workload Scheduler, is automatically mapped in a job stream that is expressly created to include the job. The job stream is (automatically) defined in the IBM Workload Scheduler database with a specific prefix defined in the [smseadapter.properties file on page 390](#).

Best practice for job security and stability

To ensure system stability and prevent known security conflicts, set the `streamlogon` user to match the agent installation owner (`wauser`) or `root` when submitting SAP jobs on the extended agent.

By aligning the *streamlogon* user with the owner of the TWSDATA directory, you avoid the permissions problem that occurs when the job attempts to write necessary diagnostic files.

If environmental constraints require using a *streamlogon* user that differs from the agent installation owner, the job successfully initializes but is expected to fail with *ABEND (Return Code 139)* upon execution.

This failure occurs because the non-default *streamlogon* user lacks the required write permissions to create and access essential log and trace files used by the `r3batch` method.

Required manual workaround

To resolve the *RC=139* error and allow the SAP job to complete, the administrator must manually adjust the file system permissions on the agent host **before** running the job. The administrator must grant full write access (`chmod 777`) to the *streamlogon* user for the following locations:

1. The `stdlist` folder within the `DATA_DIR`.
2. The `trace-r3batch.log` file located in the `TWA_DATA_DIR/methods/trace/` directory.

Scheduling jobs directly

In the Scheduling panel, before you can proceed to enter the job name and the scheduling details, you are asked to specify the identity of the scheduling system and the scheduler type, which must be SMSE. You can then specify the name of the job definition, and the job type, which can be any of the job types supported by IBM Workload Scheduler. The job is qualified by Solution Manager as an `external` job.

Select the `Status message` check box to enable monitoring tasks for the job.

In the `Start Conditions` section select when and how frequently the job will run and optionally make selections in the `Repeat every` and `Time Period` groups. Your selections are then mapped to matching run cycles, `valid from` and `valid to` dates, working days or non- working days, and time dependencies on IBM Workload Scheduler.



Note: The Extended Window start condition is not supported. All other start conditions are supported.

Use the tabs available on the uppermost part of the panel to manage jobs; for example, to copy, reschedule, release, kill, or cancel a job, create and see external notes, and check external logs.

If a scheduled job has not been started, you can change its start conditions or parameters and click **Schedule/Change Externally** again. Alternatively, you can change the start conditions and select **Reschedule** to reset the job to a new start time. In either case, IBM Workload Scheduler deletes the current job instance (that has not been started) and creates another one with the new characteristics.

On the other hand, you can click **Cancel** on a non-completed job that was already started. In this case, IBM Workload Scheduler deletes the running instance as expected.

As soon as the job is scheduled with success, the external job ID and the status are updated and you can view the job instance on the Dynamic Workload Console.

Scheduling from job documentation

With the Job Documentation option of the Job Management Administration panel, you can also create job documentation for jobs defined in IBM Workload Scheduler and scheduled from Solution Manager. From the Job Documentation menu you can view and edit job details, including job steps, basic business information, and scheduling information.

To create job documentation:

1. In the Job Documentation view, create job documentation **for a Job with detail UI**.
2. In the General pane of the new job documentation creation page, enter a job documentation name and select **Other** as *Job Type*. Selecting **Other**, *External Scheduler* is automatically selected.
3. Add a step in the Step Overview table.
4. Select a job definition type from a list of job types available from IBM Workload Scheduler.
5. Click **Save** on top of the job documentation creation page.
6. Select the **Systems** tab in the job documentation creation page and add a solution documentation, a logical component group, or a technical scenario for the new job documentation in the *Logical Component Groups*, *Solution Documentation* and *Technical Scenarios* table. Click **Save**.
7. Select **Scheduling** in the Systems table to set up scheduling definitions for the job associated with the new job documentation.

This action displays the same Scheduling panel described in [Scheduling jobs directly on page 392](#).

You can also select **Configure Monitoring** in the Systems table to set up monitoring specifications for the job.

Monitoring

Job status retrieval and job monitoring tasks are run by IBM Workload Scheduler, but you can configure and view them from the Solution Manager **Job Documentation** and **Monitoring** views. In Solution Manager to monitor a job you must configure a Business project monitoring object (BPmon). When monitoring data is requested for a job, Solution Manager through the SMSE adapter requests IBM Workload Scheduler for updates on the job status, logs, alerts.

To view the status of a job in the Solution Manager **Job Documentation** view, provided you selected the **Status message** checkbox in the **Scheduling** page, follow these steps:

1. Open the job documentation view for the job.
2. Select the **Systems** and the **Scheduling** tabs.
3. In the **Scheduling** page select the **External Log** button.

The job log is displayed in a pop-up window.

4. Select the **Refresh** button of the **External Job Status** field in the **Scheduling** page.

The current status of the job is displayed in the field.

To configure monitoring for a scheduled job with the `Status` message check box selected, go to the `Job Management Administration` panel of Solution Manager and open the `Job Documentation` view related to the job. There, select the `Systems` tab and in the ensuing page select `Configure Monitoring`.

1. In the `Identification` section of the `Job Monitoring Setup` window, input all mandatory fields.
2. Select the `Alert Setting and Data Collection` tab and configure alerts to your convenience.
3. Fill in the mandatory fields in the `Incidents and Notifications` and `Monitoring Activities` tabs.
4. Select the `Mass Generate and Activate` button on top to save and activate the monitoring object.

With the Push mechanism IBM Workload Scheduler forwards to Solution Manager the status changes that a job instance undergoes until it reaches a final status such as `complete`, `canceled`, `error`, or `killed`. IBM Workload Scheduler also forwards the following scheduling time information for the job instance:

- Estimated duration
- Actual start
- Actual completion

On the basis of this information, and according to the alert configuration you specified in the `Alert Setting and Data Collection` pane, Solution Manager triggers these alerts when any of the thresholds you specified are reached or exceeded. This grants you the means to keep the execution of your workload under control.

To view the alerts for a monitored job, select the `Unified Alert Inbox` view in the `Job Management Administration` panel:

1. Select the monitoring object for the job in the `Job Monitoring Standard View`.
2. Refresh the alert list table after some monitoring period.



Note: `Alert Inbox` and `Job Monitoring` panels can be also accessed from the `Home` in the `Job Management` section.

Setting the traces on the application server for the major IBM Workload Scheduler processes

About this task

The application server handles all communications between the IBM Workload Scheduler processes. The trace for these communications is set to `tws_info` by default (information messages only). The application server can be set to trace all communications, either for the whole product or for these specific groups of processes:

- Command line
- Connector
- Database
- Planner
- Utilities
- Dynamic workload broker

Significant impact on performance: Activating traces for the WebSphere Application Server Liberty Base leads to a significant impact on performance, especially if you set the tracing to `all`. Thus you are strongly advised to identify the process group where the problem that you want to trace is occurring, and only set the trace to that group.

To modify the trace level on the WebSphere Application Server Liberty Base, edit the `trace.xml` file as necessary.

Templates for the master domain manager are stored in the following paths:

On UNIX operating systems

`TWA_home/usr/servers/engineServer/configDropins/templates`

On Windows operating systems

`TWA_home\usr\servers\engineServer\configDropins\templates`

Templates for the Dynamic Workload Console are stored in the following paths:

On UNIX operating systems

`DWC_home/usr/servers/dwcServer/configDropins/templates`

On Windows operating systems

`DWC_home\usr\servers\dwcServer\configDropins\templates`

When you edit the file with your customized settings for the master domain manager, move it to the following paths:

On UNIX operating systems

`TWA_DATA_DIR/usr/servers/engineServer/configDropins/overrides`

On Windows operating systems

`TWA_home\usr\servers\engineServer\configDropins\overrides`

When you edit the file with your customized settings for the Dynamic Workload Console, move it to the following paths:

On UNIX operating systems

`DWC_DATA_dir/usr/servers/dwcServer/configDropins/overrides`

On Windows operating systems

`DWC_home\usr\servers\dwcServer\configDropins\overrides`

1. Copy the template file from the `templates` folder to a working folder.
2. Edit the template file in the working folder with the desired configuration.
3. Optionally, create a backup copy of the relevant configuration file present in the `overrides` directory in a different directory. Ensure you do not copy the backup file in the path where the template files are located.
4. Copy the updated template file to the `overrides` folder. Maintaining the original folder structure is not required.
5. Changes are effective immediately.

For example, to modify the trace level on WebSphere Application Server Liberty Base, perform the following steps:

1. Copy the `trace.xml` file from the `TWA_home/usr/servers/engineServer/configDropins/templates` folder to a working folder.
2. Edit the template file in the working folder by changing the following string:

```
<variable name="trace.specification" value="*info"/>
```

into

```
<variable name="trace.specification" value="com.ibm.tws.dao.model.  
=all:com.ibm.tws.dao.rdbms.=all"/>
```

3. Copy the updated template file to `TWA_DATA_DIR /usr/servers/engineServer/configDropins/overrides`. Changes are effective immediately.

Traces are stored in `DWC_DATA_dir/appserver/engineServer/logs`.

The **trace.specification** can be found in `trace.xml` files and can refer to a specific component (`tws_XXX`) or to the whole product, as follows:

tws_all

```
"com.ibm.tws.=all:org.apache.wink.server.=all:com.hcl.tws.*=all"
```

tws_alldefault

```
"com.ibm.tws.*=error=enabled"
```

tws_broker_all

```
"com.ibm.scheduling.*=all:TWSAgent=all"
```

tws_rest

```
"com.ibm.tws.conn.=all:com.ibm.tws.twsd.rest.=all:org.apache.wink.server.*=all"
```

tws_cli

```
"com.ibm.tws.cli.=all:com.ibm.tws.objects.=all"
```

tws_utils

```
"com.ibm.tws.util.*=all"
```

tws_conn

```
"com.ibm.tws.conn.=all:com.ibm.tws.objects.=all:com.ibm.tws.updatemanager.=all:com.ibm.tws.dao.plan.=a  
ll"
```

tws_db

```
"com.ibm.tws.dao.model.=all:com.ibm.tws.dao.rdbms.=all"
```

tws_planner

```
"com.ibm.tws.planner.=all:com.tivoli.icalendar.=all:com.ibm.tws.runcycles.=all:com.ibm.tws.conn.planner.=all  
:com.ibm.tws.cli.planner.*=all"
```

tws_secjni

```
"com.ibm.tws.audit.=all:com.ibm.tws.security.=all"
```

twS_engine_broker_all

```
"com.ibm.tws.=all:com.ibm.scheduling.=all:TWSAgent=all"
```

Editing the logging element above with the **traceSpecification** value to `twS_all`, enables

```
"com.ibm.tws.=all:org.apache.wink.server.=all:com.hcl.tws.*=all".
```

Other values are reported in variable tags above. You can also replace the value of the **trace.specification** parameter with a custom string.

Appendix E. Collecting job metrics

You can run the following SQL queries on the Workload Scheduler database to retrieve the number of jobs run by IBM Workload Scheduler over a period of time. One query determines the number of jobs run by specific workstations, while the other query determines the number of jobs run on the entire IBM Workload Scheduler domain.

You can run the queries from the command line interface of your database or you can add them in the Dynamic Workload Console to create your custom SQL report.

Job metrics queries for DB2 for zOS

Use the following SQL query to find the number of jobs run on specific workstations:

```
SELECT year(job_run_date_time) AS Year, month(job_run_date_time) AS Month,
cast (count(job_run_date_time) AS INT) AS JobNbr FROM mdl.job_history_v
WHERE workstation_name IN ('WKS_1', 'WKS_2', 'WKS_N')
GROUP BY year(job_run_date_time), month(job_run_date_time)
```

where 'WKS_1', 'WKS_2', 'WKS_N' are the names of the workstations that ran the jobs you want counted.

Use the following SQL query to find the number of jobs run on the entire IBM Workload Scheduler domain:

```
SELECT year(job_run_date_time) AS Year, month(job_run_date_time) AS Month,
cast (count(job_run_date_time) AS INT) AS JobNbr FROM mdl.job_history_v
GROUP BY year(job_run_date_time), month(job_run_date_time)
```

Job metrics queries for DB2

Use the following SQL query to find the number of jobs run on specific workstations:

```
SELECT year(job_run_date_time) AS Year, month(job_run_date_time) AS Month,
cast (count(job_run_date_time) AS INT) AS JobNbr FROM mdl.job_history_v
WHERE workstation_name IN ('WKS_1', 'WKS_2', 'WKS_N')or
(workstation_name = '-' and JOB_STREAM_WKS_NAME_IN_RUN in('WKS_1', 'WKS_2', 'WKS_N') )
GROUP BY year(job_run_date_time), month(job_run_date_time)
```

where 'WKS_1', 'WKS_2', 'WKS_N' are the names of the workstations that ran the jobs you want counted.

Use the following SQL query to find the number of jobs run on the entire IBM Workload Scheduler domain:

```
SELECT year(job_run_date_time) AS Year, month(job_run_date_time) AS Month,
cast (count(job_run_date_time) AS INT) AS JobNbr FROM mdl.job_history_v
GROUP BY year(job_run_date_time), month(job_run_date_time)
```

Notices

This document provides information about copyright, trademarks, terms and conditions for product documentation.

© Copyright IBM Corporation 1993, 2016 / © Copyright HCL Technologies Limited 2016, 2025

This information was developed for products and services offered in the US. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

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.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

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.

© (HCL Technologies Limited) (2025).

Portions of this code are derived from IBM Corp. Sample Programs.

© Copyright IBM Corp. 2016

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.

Adobe™, the Adobe™ logo, PostScript™, and the PostScript™ logo are either registered trademarks or trademarks of Adobe™ Systems Incorporated in the United States, and/or other countries.

IT Infrastructure Library™ is a Registered Trade Mark of AXELOS Limited.

Linear Tape-Open™, LTO™, the LTO™ Logo, Ultrium™, and the Ultrium™ logo are trademarks of HP, IBM® Corp. and Quantum in the U.S. and other countries.

Intel™, Intel™ logo, Intel Inside™, Intel Inside™ logo, Intel Centrino™, Intel Centrino™ logo, Celeron™, Intel Xeon™, Intel SpeedStep™, Itanium™, and Pentium™ are trademarks or registered trademarks of Intel™ Corporation or its subsidiaries in the United States and other countries.

Linux™ is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft™, Windows™, Windows NT™, and the Windows™ logo are trademarks of Microsoft™ Corporation in the United States, other countries, or both.



Java™ and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Cell Broadband Engine™ is a trademark of Sony Computer Entertainment, Inc. in the United States, other countries, or both and is used under license therefrom.

ITIL™ is a Registered Trade Mark of AXELOS Limited.

UNIX™ is a registered trademark of The Open Group in the United States and other countries.

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.

Index

A

- ABAP step definition
 - attribute 293
- ABAP/4 modules
 - SAP
 - importing
 - 221
- access method 166
 - agent
 - syntax 166
 - dynamic agent
 - option file 169
 - extended agent
 - option file 169
 - IBM Z Workload Scheduler Agent
 - option file 169
- interface 166
- PeopleSoft
- options
 - 188
 - SAP
 - 213
 - SAP
 - common options
 - 237
 - SAP
 - global configuration options
 - 231
 - SAP
 - local configuration options
 - 234
 - task options 166
- access method for
- dynamic agent
 - overview 165
- access method for extended agent
 - overview 165
- access method jobs 74
- activating
 - criteria profile 304
 - job interception 315
- Administering agents on AS/400 116
- Administering agents on i5/OS 116
- Administering agents on IBM i 116, 116
- Administering IBMi agents 116
- Administering IBMi jobs 116
- Administering jobs on AS/400 116
- Administering jobs on i5/OS 116
- advanced job types 63, 65
- agent
 - access method
 - syntax 166
- agent configuration
 - JobManager.ini file 18
 - maintenance 38
- agent log and trace files
 - twstrace syntax 25, 127
- agent logs encoding 22, 124
- agent logs setting 22, 124
- agent problems 158
- agent traces
 - modifying 25, 127
 - viewing settings 25, 127
- agent working directory
 - JobManager.ini file 18
- APARs
 - IJ30805 72

- IY92806 247, 318, 374
- IY97424 313, 313
- IZ03505 232, 232, 232, 232, 232, 235, 235,
235, 235, 235
- IZ12321 210
- IZ26839 376
- IZ33555 244
- IZ37273 270, 290
- IZ42262 273, 287, 293, 297
- PI54546 ix
- PK00789 47
- PK40356 47
- APPENDTEXT, keyword of JOBREC for file
transfer job types 87
- application server
 - SAP
 - 250
 - trace settings 394
- application servers
 - multiple,
PeopleSoft
 - 191
- AROPTS initialization statement
 - ENDTIME keyword 147
 - STARTTIME keyword 147
- AUTHALIAS, keyword of JOBREC 97
- authorization profile
 - SAP
 - 217
 - transaction PFCG 218
 - transaction su02 217

B

- balancing
 - SAP
 - workload using server groups
 - 276
- batch processing ID
 - PeopleSoft
 - 192
- BDC wait
 - R/3 304
- broker jobs promotion 113
- broker promotion variables 113
- broker workstation
 - defining 64
- broker workstations
 - creating 67
 - defining 67
 - jobs by reference 67
 - options 67
- building
 - criteria hierarchy 303
- Business Information Warehouse support
 - SAP
 - 316
- Business Warehouse components
 - InfoPackage 316
 - process chain 316
 - SAP
 - 316
- Business Warehouse InfoPackage
 - displaying details 323
- Business Warehouse InfoPackage and
process chain
 - managing 317

C

- call to a web service
 - sample JSDL files 17, 74
- CCMS
 - sending data from job throttler 337
- CCMS event
 - committing MTE alert 371
 - defining event rule, business scenario 361,
362
 - getting CCMS alert status 370
- changing password, RFC user
 - SAP
 - 225
- CHECKINTERVAL,
PeopleSoft
- option
 - 189
- child job on i5/OS 117
- child job on IBM i 117
- child job settings on AS/400 for
performances 117
- child job settings on i5/OS for
performance 117
- child job settings on IBM i for
performances 117
- child jobs settings for performances on
AS/400 117
- CLASSNAME, keyword of JOBREC 94
- CMDTYPE statements
 - JOBREC
 - CMDTYPE keyword 78, 82
- CMDTYPE, keyword of JOBREC 78, 82
- code page
 - view 62
- collecting job interception 307, 308
- command line
 - committing
 - SAP
 - event
 - 345, 371
 - defining extended agent job 184
 - defining extended agent workstation 179
 - getting CCMS alert status 370
 - monitoring
 - SAP
 - event
 - 252
 - commands
 - filemonitor 107
 - filewatch 103
 - common options,
SAP
 - 237
 - composer program
 - defining extended agent job 184
 - defining extended agent workstation 179
 - configuration file
 - example,
R/3
 - 210
 - configuration options
 - PeopleSoft
 - 188
 - SAP
 - , common
 - 237
 - SAP
 - , global
 - 231

- SAP
 - , local
 - 234
 - SAP
 - , usage
 - 249
- configuring
 - job class inheritance for job throttling 335
 - job interception for job throttling 334
 - parent-child for job throttling 315
 - SAP
 - 216
 - SAP
 - access method
 - 229
 - SAP
 - environment
 - 216
 - tracing utility 207
 - configuring proxy 21
 - connecting
 - SAP
 - 250
 - connecting to SAP 256
 - connection SSL
 - Z controller and IBM i agents 140
 - Z controller
 - and
 - IBM Z Workload Scheduler Agents
 - 71
 - connection to
 - SAP
 - troubleshooting 375
 - connections
 - HTTP 42
 - CONNFACORY, keyword of JOBRFC 98, 98
 - considerations about return code
 - mapping 202
 - control file
 - SAP
 - 220
 - correction and transport files
 - SAP
 - 220
 - CPU optimization
 - job plugins 29, 132
 - CPUREC statement
 - creating 181
 - creating
 - action parameters for CCMS event
 - rules 368
 - correlation rules for CCMS events 368
 - CPUREC statement 181
 - criteria profile 302
 - DOMREC statement 181
 - event rule based on CCMS alerts 363
 - event rule based on CCMS alerts, business
 - scenario 361, 362
 - event rule based on IDocs 352
 - event rule based on IDocs, business
 - scenario 351
 - event rule based on
 - SAP
 - event
 - 346
 - internetwork dependency based on
 - SAP
 - event
 - 343, 346
 - job containing InfoPackage 317
 - job containing process chains 317

- PeopleSoft
 - job
 - 196
 - RFC user,
 - SAP
 - 217
 - SAP jobs 254
 - creating and managing variables and
 - passwords locally 52
 - creating and managing variables and
 - passwords locally on the agents 49
 - criteria hierarchy
 - building 303
 - description 300
 - Criteria Manager 300
 - criteria profile
 - activating 304
 - creating 302
 - description 300
 - critical dynamic job promotion 113
 - crucial dynamic jobs 113
 - CUSTOM keyword, to filter
 - SAP
 - events
 - 350
 - customization procedure
 - SAP
 - 216
 - customizing
 - properties file 208, 208, 209
 - CyberArk integration
 - configuration 18

D

- data file
 - SAP
 - 220
- database
 - collecting job metrics 398
 - database data extract 17, 74
 - database data validation 17, 74
 - database executor 76
 - database job 76
 - configuring 139, 159
 - database operations
 - sample JSDL files 17, 74
 - database stored procedure
 - database jobs
 - sample JSDL files 17, 74
 - sample JSDL files 17, 74
- DB2
 - collecting job metrics 398
- DB2 database job type
 - detailed syntax diagram 78
- DB2 for zOS
 - collecting job metrics 398
- DBNAME, keyword of JOBRFC 79, 82
- DBTYPE, keyword of JOBRFC 79, 82
- defining
 - ABAP step attribute 293
 - action parameters for CCMS event
 - rules 368
 - broker workstation 64
 - correlation rules for CCMS events 368
 - destinations 39
 - dynamic pools 66
 - event rule based on CCMS alerts 363
 - event rule based on CCMS alerts, business
 - scenario 361, 362
 - event rule based on IDocs 352

- event rule based on IDocs, business
 - scenario 351
- event rule based on
 - SAP
 - event
 - 346
 - external command step attribute 297
 - external program step attribute 297
 - file dependencies 103
 - global options file 174
 - local options file 174
 - PeopleSoft
 - job
 - 196
 - pools 66
 - SAP
 - event as internetwork dependency
 - 343, 346
 - job
 - 262, 319
 - SAP
 - job dynamically
 - 282
 - SAP jobs 254
 - SAP
 - variant
 - 257
 - supported agents
 - job
 - 183
 - supported agents
 - workstation
 - 178
 - z-centric workstation 55
 - defining job
 - SAP
 - 262
 - SAP
 - , dynamically
 - 282
 - DELETEAFTERDOWNLOAD, keyword of
 - JOBRFC for file transfer job types 87
 - DELETEAFTERUPLOAD, keyword of JOBRFC
 - for file transfer job types 87
 - deleting
 - ITWS for Apps 338
 - SAP
 - job
 - 275
 - SAP
 - variant
 - 257
 - dependency
 - based on
 - SAP
 - event, defining
 - 343, 346
 - based on
 - SAP
 - event, limitation with XBP 2.0
 - 343
 - committing
 - SAP
 - event by external task
 - 345
 - mapping between definition and resolution,
 - SAP
 - 344
 - destinations
 - defining 39

- displaying in MLOG 39
- listing in MLOG 39
- modifying 39
- refreshing 39
- view 62
- detailed syntax diagram
 - DB2 database job type 78
 - download job type 85
 - executable job type 99
 - file transfer job type 85
 - J2EE jms job type 97
 - Java job type 93
 - MSSQL database job type 81
 - native job type 99
 - Oracle database job type 78
 - script job type 99
 - upload job type 85
 - web service job type 91
 - xajob job type 95
- displaying details
 - Business Warehouse InfoPackage 323
 - Business Warehouse InfoPackage job 323
 - process chain job 324
 - SAP
 - job
 - 274
- DOMREC statement
 - creating 181
- download job type
 - detailed syntax diagram 85
- dynamic agent
 - access method
 - option file 169
 - overview 165
- dynamic domain manager
 - destinations 39
 - modifying destinations 39
- dynamic domain managers
 - defining destinations 39
 - heartbeat checking interval 39
- dynamic job definition parameter description
 - SAP
 - 285
- dynamic job definition syntax
 - ABAP step definition attribute 293
 - external command step definition
 - attribute 297
 - external program step definition
 - attribute 297
 - SAP
 - 282
- dynamic job promotion 113
- dynamic pools
 - defining 66
- dynamic scheduling
 - job types with advanced options
 - 17
- Dynamic Workload Console
 - defining
 - supported agents
 - job
 - 183
 - defining workstation for agent with access
 - method 178
- Dynamic Workload Console
 - global settings
 - 42
- dynamically
 - defining
 - SAP
 - jobs

- 282
- dynamically, defining
 - SAP
 - job
 - 282
- E**
 - encrypting passwords 191, 225, 249
 - encrypting user password
 - PeopleSoft
 - 191
 - SAP
 - 249
 - end-to-end scheduling
 - defining extended agent job 185
 - defining supported agent workstation 180
 - end-to-end with z-centric capabilities
 - overview 13
 - steps to set up environment 13
 - ENDTIME keyword of AROPTS 147
 - enigma program
 - encrypting user password 249
 - environment variables
 - job promotion 113
 - ENVVAR, keyword of JOBREC 95, 100
 - EQQMOPRL panel 114
 - EQQWBGEP panel 56
 - EQQWCEOD panel 65
 - EQQWCGEP panel 55, 65
 - EQQWDGEP panel 57
 - EQQWMGEP panel 56
 - EQQWWSEP panel 57
 - event rule
 - action parameters for CCMS event
 - rules 368
 - based on CCMS alerts 362
 - based on CCMS alerts, business
 - scenario 361, 362
 - based on IDocs 352
 - based on IDocs, business scenario 351
 - based on IDocs, matching criteria 353
 - correlation rules for CCMS alerts 368
 - definition 346
 - monitoring
 - SAP
 - event
 - 252
 - SAP
 - , defining
 - 346
 - SAP
 - , filtering events
 - 350
 - SAP
 - , prerequisite to define
 - 252
 - events, logging 302
 - events, raising SAP 278
 - example
 - dynamic
 - SAP
 - job definition
 - 298
 - return code mapping 202
 - executable job type
 - detailed syntax diagram 99
 - executable jobs 17, 74
 - executable or script executor 76
 - executors 76
 - exporting SAP calendars
 - business scenario 339

- exporting SAP R/3 calendars
 - r3batch export function 340
- extended agent
 - access method
 - option file 169
 - response messages 168
 - running 172
 - overview 165
- extended agent job
 - SAP
 - 253
- extended agent job defining with
 - command line 184
- extended agent workstation, defining with
 - command line 179
 - ISPF 182
- extended and dynamic agent workstation
 - defining with
 - end-to-end scheduling 180
- EXTENSION, keyword of JOBREC 101
- external command step definition
 - attribute 297
- external program step definition
 - attribute 297
- EXTRACOMMANDS, keyword of JOBREC 87
- F**
 - feature
 - job interception, setting
 - SAP
 - 309, 309
 - features
 - job interception and parent-child,
 - SAP
 - 305
 - job interception, activating
 - SAP
 - 315
 - job interception, collecting
 - SAP
 - 307, 308
 - job interception, implementing
 - SAP
 - 306
 - job interception, SAP 306
 - job interception, setting
 - SAP
 - 309
 - job throttling,
 - SAP
 - 333
 - parent-child R/3 315
 - PeopleSoft
 - 186
 - return code mapping 200
 - SAP
 - 211
 - file 208
 - configuration for
 - R/3
 - 210
 - psagent.properties 208
 - r3batch.properties 208
 - return code mapping 201
 - file changes, checking 103, 107
 - file dependencies, defining 103
 - file name
 - return code mapping 205
 - file transfer executor 76
 - file transfer job 76
 - file transfer job type

- detailed syntax diagram 85
- file transfer jobs
 - sample JSDL files 17, 74
- file transfer operations
 - sample JSDL files 17, 74
- filemonitor command 107
- files
 - JobManagerGW.ini 34, 136
- filewatch
 - filewatchdb utility 106
 - installing 103
 - maintaining log database 106
 - variable substitution 103
- filewatchdb, syntax 106
- filtering
- SAP
 - events in security file 350

G

- gateway
 - configure 34, 136
- generic Java job
 - template 17, 74
- generic web service call
 - template 17, 74
- global options file
 - defining 174
 - mvsjes.opts 175, 176
 - mvsopc.opts 175, 176
 - name 170, 175, 176
 - psagent.opts 175, 176
 - r3batch.opts 175, 176
- global options, SAP 231

H

- heartbeat checking
 - z-centric agents and dynamic domain managers 43
- HTTP 42, 42
- HTTP subtasks
 - start, stop 45
- HTTPOPTS 42

I

- IBM i jobs 74
 - AS400 jobs 74
- IBM Workload Scheduler agent
 - log configuration 22, 124
 - trace configuration 23, 126
- IBM Z Workload Scheduler Agent
 - access method
 - option file 169
 - browsing 56
- IBM Z Workload Scheduler Agents
 - defining destinations 39
 - heartbeat checking interval 39
 - SAF SSL connection with z/OS
 - Z controller
 - 68
 - SSL connection with z/OS
 - Z controller
 - 71
- IBM Z Workload Scheduler Agents on
 - SAF SSL connection with Z controller 68

- SSL connection with Z controller 71
- IDoc
 - defining event rule 352
 - defining event rule, business scenario 351
 - defining event rule, matching criteria 353
- implementing
 - job interception 306
- important dynamic jobs 113
- InfoPackage
 - managing 317
 - user authorization 317
- InfoPackage schedule options
 - SAP 317
- inheritance r3batch
 - definition 177
- initialization statements
 - USRREC
 - definition 46, 47
 - USRCPU keyword 47
 - USRNAM keyword 47
 - USRPSW keyword 47
- installing
 - ABAP modules, SAP 217
- intercepted job
 - return code mapping 207
- interception criteria
 - setting, SAP 309
- interface 166
- internetwork dependency
 - based on SAP
 - event, defining 343, 346
 - based on SAP
 - event, limitation with XBP 2.0 343
 - based on SAP
 - event, limitation with XBP 3.0 343
 - based on SAP R/3 event, prerequisites 342
 - committing
 - SAP
 - event by external task 345
 - mapping between definition and resolution, SAP 344
 - placeholder
 - SAP
 - job 343
- INTRACTV, keyword of JOBREC 101
- introduction
 - IBM Workload Scheduler
 - for SAP 210
 - PeopleSoft 186
 - SAP 213

- INVOKEYTYPE, keyword of JOBREC 98
- ISPF
 - defining extended agent workstation 182
- ITWS_PSPA project
 - PeopleSoft 192

J

- j2ee jms executor 76
- j2ee jms job 76
- J2EE jms job type
 - detailed syntax diagram 97
- J2EE job
 - configuring 139, 159
- J2EE jobs 17, 74
- JARPATH, keyword of JOBREC 94
- java executor 76
- java job 76
- Java job
 - configuring 139, 159
- Java job configuration
 - JobManager.ini file 18
- Java job type
 - detailed syntax diagram 93
- Java jobs
 - sample JSDL files 17, 74
- Java operations
 - sample JSDL files 17, 74
- Java Virtual Machine options 33, 136
- JAVAPARM, keyword of JOBREC 94
- JCL
 - RECOVER statement 143
- job
 - assigning a server group 276
 - PeopleSoft 196
 - SAP 262
 - job state 276
 - submitting for supported agent 186
- job definition
 - creating 74
 - PeopleSoft 196
 - SAP 262
 - SAP 282
 - , dynamically 282
- job definition parameter description
 - SAP 285
- job environment on AS/400 121
- job environment on i5/OS 121
- job environment on IBM i 121
- job executors 63, 65
 - configuring 139, 159
 - Java options 33, 136
- job
 - IBM Workload Scheduler
 - creating job containing InfoPackage 317
 - creating job containing process chain 317
- job interception
 - activating, SAP 315
 - collecting, SAP 315
 - feature 315

- 307, 308
- enabling and configuring for job throttling 334
- implementing, SAP
 - feature
 - 306
 - SAP
 - feature
 - 305
 - setting placeholders in template file 314
 - setting, SAP
 - feature
 - 309, 309
- job log collection
 - 153
- job log header
 - customization 151
 - template 151
- job metrics
 - collecting 398
 - SQL queries 398
- job plug-ins 63, 65
 - configuring 139, 159
 - Java options 33, 136
- job promotion
 - environment variables 113
- job promotion on dynamic pools 113
- job state
 - SAP
 - 276
- job status mapping
 - PeopleSoft
 - 198
- job throttling
 - business scenario 333
 - configuring logging properties 335
 - deleting ITWS for Apps 338
 - enabling and configuring job interception 334
 - enabling job class inheritance 335
 - options in options file 334
 - SAP
 - feature
 - 333
 - sending data to CCMS 337
 - starting 336
 - stopping 337
 - throttling_send_ccms_data 337
 - throttling_send_ccms_rate 337
 - job tracking
 - PeopleSoft
 - 187
 - job types
 - template 17, 74
 - job types with advanced options
 - configuration files 139, 159
 - configuring 139, 159
 - customizing 139, 159
 - Java options 33, 136
 - log configuration 159
 - sample JSDL files 17
 - template 17
 - job types with advanced options configuration files
 - location 139, 159
 - JOBCMD, keyword of JOBREC 101
 - JOBLIB statements
 - JOBREC

- APPENDTEXT keyword for file transfer job types 87
- AUTHALIAS keyword 97
- CLASSNAME keyword 94
- CONNFACTORY keyword 98, 98
- DBNAME keyword 79, 82
- DBTYPE keyword 79, 82
- definition 75
- DELETEAFTERDOWNLOAD keyword for file transfer job types 87
- DELETEAFTERUPLOAD keyword for file transfer job types 87
- ENVVAR keyword 95, 100
- EXTENSION keyword 101
- EXTRACOMMANDS keyword 87
- INTRACTV keyword 101
- INVOKEYTYPE keyword 98
- JARPATH keyword 94
- JAVAPARM keyword 94
- JOBCMD keyword 101
- JOBPWD keyword 79
- JOBPWD keyword for file transfer job types 87
- JOBPWD keyword for J2EE jms job type 98
- JOBPWD keyword for MSSQL database job type 83
- JOBPWD keyword for script and executable job type 101
- JOBPWD keyword for web service job types 92
- JOBPWD keyword for xajob job type 96
- JOBTYP keyword 79
- JOBTYP keyword for file transfer job types 87
- JOBTYP keyword for J2EE jms job type 98
- JOBTYP keyword for Java job type 94
- JOBTYP keyword for MSSQL database job type 83
- JOBTYP keyword for web service job types 92
- JOBTYP keyword for xajob job type 96
- JOBUSR keyword 79
- JOBUSR keyword for file transfer job types 87
- JOBUSR keyword for J2EE jms job type 98
- JOBUSR keyword for MSSQL database job type 83
- JOBUSR keyword for script and executable job type 101
- JOBUSR keyword for web service job types 92
- JOBUSR keyword for xajob job type 96
- LOCALCODEPAGE keyword 88
- LOCALFILE keyword 88
- LOCALPWD keyword 88, 88
- MAXPORT keyword 88
- MESSAGE keyword 99
- MINPORT keyword 88
- OPNAME keyword 93
- PARM keyword 93, 102
- PASSIVEMODE keyword for file transfer job types 89
- PORT keyword 80, 84
- PROTOCOL keyword 89
- REMOECODEPAGE keyword 89
- REMOTEFILE keyword 90
- SERVER keyword 80, 84, 90
- STATEMENT keyword 80, 84

- STDERROR keyword 102
- STDINPUT keyword 102
- STDOUTOUTPUT keyword 102
- TIMEOUT keyword 90
- TRANSFERMODE keyword 90
- TRANSFERTYPE keyword 90
- URL keyword 93
- WRKDIR keyword 102
- JOBPWD, keyword of JOBREC 79
- JOBPWD, keyword of JOBREC for file transfer job types 87
- JOBPWD, keyword of JOBREC for J2EE jms job type 98
- JOBPWD, keyword of JOBREC for MSSQL database job type 83
- JOBPWD, keyword of JOBREC for script and executable job type 101
- JOBPWD, keyword of JOBREC for web service job types 92
- JOBPWD, keyword of JOBREC for xajob job type 96
- JOBREC 52
- JOBREC statement of JOBLIB
 - APPENDTEXT keyword for file transfer job types 87
 - AUTHALIAS keyword 97
 - CLASSNAME keyword 94
 - CMDTYPE keyword 78, 82
 - CONNFACTORY keyword 98, 98
 - DBNAME keyword 79, 82
 - DBTYPE keyword 79, 82
 - definition 75
 - DELETEAFTERDOWNLOAD keyword for file transfer job types 87
 - DELETEAFTERUPLOAD keyword for file transfer job types 87
 - ENVVAR keyword 95, 100
 - EXTENSION keyword 101
 - EXTRACOMMANDS keyword 87
 - INTRACTV keyword 101
 - INVOKEYTYPE keyword 98
 - JARPATH keyword 94
 - JAVAPARM keyword 94
 - JOBCMD keyword 101
 - JOBPWD keyword 79
 - JOBPWD keyword for file transfer job types 87
 - JOBPWD keyword for J2EE jms job type 98
 - JOBPWD keyword for MSSQL database job type 83
 - JOBPWD keyword for script and executable job type 101
 - JOBPWD keyword for web service job types 92
 - JOBPWD keyword for xajob job type 96
 - JOBTYP keyword 79
 - JOBTYP keyword for file transfer job types 87
 - JOBTYP keyword for J2EE jms job type 98
 - JOBTYP keyword for Java job type 94
 - JOBTYP keyword for MSSQL database job type 83
 - JOBTYP keyword for web service job types 92
 - JOBTYP keyword for xajob job type 96
 - JOBUSR keyword 79
 - JOBUSR keyword for file transfer job types 87
 - JOBUSR keyword for J2EE jms job type 98
 - JOBUSR keyword for MSSQL database job type 83

- JOBUSR keyword for script and executable job type 101
- JOBUSR keyword for web service job types 92
- JOBUSR keyword for xajob job type 96
- LOCALCODEPAGE keyword 88
- LOCALFILE keyword 88
- LOCALPWD keyword 88, 88
- MAXPORT keyword 88
- MESSAGE keyword 93, 99
- MINPORT keyword 88
- PARM keyword 93, 102
- PASSIVEMODE keyword for file transfer job types 89
- PORT keyword 80, 84
- PROTOCOL keyword 89
- REMOTECODEPAGE keyword 89
- REMOTEFILE keyword 90
- SERVER keyword 80, 84, 90
- STATEMENT keyword 80, 84
- STDERROR keyword 102
- STDINPUT keyword 102
- STDOUTPUT keyword 102
- TIMEOUT keyword 90
- TRANSFERMODE keyword 90
- TRANSFERTYPE keyword 90
- URL keyword 93
- WRKDIR keyword 102

jobs

- new SAP 254
- jobs by reference
 - submitting 67
- jobs dynamic definition
 - SAP 282

- jobthrottling.bat, usage parameter 336
- jobthrottling.sh, usage parameter 336
- JOBTYPE, keyword of JOBREC 79
- JOBTYPE, keyword of JOBREC for file transfer job types 87
- JOBTYPE, keyword of JOBREC for J2EE jms job type 98
- JOBTYPE, keyword of JOBREC for Java job type 94
- JOBTYPE, keyword of JOBREC for MSSQL database job type 83
- JOBTYPE, keyword of JOBREC for web service job types 92
- JOBTYPE, keyword of JOBREC for xajob job type 96
- JOBUSR, keyword of JOBREC 79
- JOBUSR, keyword of JOBREC for file transfer job types 87
- JOBUSR, keyword of JOBREC for J2EE jms job type 98
- JOBUSR, keyword of JOBREC for MSSQL database job type 83
- JOBUSR, keyword of JOBREC for script and executable job type 101
- JOBUSR, keyword of JOBREC for web service job types 92
- JOBUSR, keyword of JOBREC for xajob job type 96
- JVM options 33, 136

K

- killing
 - SAP job 277

L

- listing destinations 39
- LJUSER, PeopleSoft
 - option 189
- local options file
 - defining 174
 - name 170, 175, 176
- local options, SAP
 - 234
 - LOCALCODEPAGE, keyword of JOBREC 88
 - LOCALFILE, keyword of JOBREC 88
 - LOCALPWD, keyword of JOBREC 88, 88
- log configuration
 - IBM Workload Scheduler agent 22, 124
- log database, filewatch 106
- log files
 - maintenance 38
- log level on AS/400 116, 117
- log level on i5/OS 116, 117
- log level on IBM i 116, 117
- log settings on AS/400 116, 117
- log settings on i5/OS 116, 117
- log settings on IBM i 116, 117
- logging
 - job types with advanced options 159
- logging raised events 302
- logon group
 - SAP 251
- logs
 - agent encoding 22, 124

M

- maintenance
 - agent configuration 38
- managing
 - Business Warehouse InfoPackage and process chain 317
 - SAP extended agent job running 253
- mapping
 - IBM Workload Scheduler and SAP job states 276
- mapping job status
 - PeopleSoft 198
- max_jobs_to_release_for_user 231
- MAXPORT, keyword of JOBREC 88
- MESSAGE, keyword of JOBREC 99
- MINPORT, keyword of JOBREC 88
- MODIFY command
 - dspdest 39
 - rfrdest 39
 - rfruser 46
- modifying
 - agent traces 25, 127
- modifying operations in the current plan panel 114
- monitoring
 - MTEs 362
 - SAP event defined as event rule 252

- SAP
 - event defined as internetwork dependency, XBP 2.0 343
- MSSQL database job type
 - detailed syntax diagram 81
- MSSQL jobs 17, 74
- MTE alert
 - action parameters for CCMS event rules 368
 - committing by external task 371
 - correlation rules 368
 - defining event rule 363
 - defining event rule, business scenario 361, 362
 - getting CCMS alert status 370
 - mapping
 - attribute MTE name and IBM Workload Scheduler fields 367
 - context MTE name and IBM Workload Scheduler fields 366
 - object MTE name and IBM Workload Scheduler fields 367
 - multiple application servers, PeopleSoft 191
- mvsjes.opts file
 - definition 175, 176
- mvsopc.opts file
 - definition 175, 176

N

- name
 - global options file 170, 175, 176
 - local options file 170, 175, 176
- National Language support
 - SAP 373
- native job configuration
 - JobManager.ini file 18
- native job type
 - detailed syntax diagram 99
- new copy
 - re-running jobs SAP 281
- new executors 76
 - template 17, 74
- new plug-ins 17, 74
 - template 17, 74

O

- old copy
 - re-running jobs SAP 281
- operating system
 - view 62
- operator password
 - encrypting on PeopleSoft 191
- OPNAME, keyword of JOBREC 93
- option file
 - dynamic agent
 - access method 169
 - extended agent
 - access method 169
 - IBM Z Workload Scheduler Agent

- RFC 225
- REMOTECODEPAGE, keyword of JOBREC 89
- REMOTEFILE, keyword of JOBREC 90
- rerunning job
 - process chain 326, 330
 - SAP
 - 280, 326
- Resource advisor agent
 - JobManager.ini file 18
- restarting
 - process chain 330
- return code mapping 203
 - considerations 202
 - example 202
 - feature 200
 - file name 205
 - file, creating 201
 - intercepted job 207
 - parameter 201
 - PeopleSoft
 - 203
 - R/3 204
 - syntax 202
- return code on AS/400 121
- return code on i5/OS 121
- return code on IBM i 121
- return codes
 - NOERROR parameter 161
- RFC
 - Remote Function Call 225
- RFC profile
 - SAP
 - 217
- RFC user
 - SAP
 - 375
- RFC user ID password
 - changing 225
 - encrypting 225
- RFC user password
 - SAP
 - 225
- roles and tasks
 - PeopleSoft
 - 187
 - SAP
 - 214
- ROUTOPTS 39
- ROUTOPTS initialization statement
 - making changes immediately effective 39
- RUNLOCATION,
 - PeopleSoft
 - option
 - 189

S

- SAF connection SSL
 - Z controller
 - and
 - IBM Z Workload Scheduler Agents
 - 68
- SAF SSL connection with
 - IBM Z Workload Scheduler Agents
 - 68
- SAP
 - ABAP step definition attribute 293
 - ABAP/4 modules, importing 221
 - access method 213
 - application server 250
 - authorization profile 217
 - Business Information Warehouse
 - support 316
 - Business Warehouse components 316
 - CCMS alerts used in event rule 362
 - changing password, RFC user 225
 - committing MTE alert by external task 371
 - committing SAP event by external task 345
 - common options 237
 - configuration 216
 - configuration options 231
 - configuration options usage 249
 - connecting 250
 - connection 375
 - control file 220
 - correction and transport files 220
 - creating job containing InfoPackage 317
 - creating job containing process chain 317
 - creating jobs 254
 - creating RFC user 217
 - customization procedure 216
 - data file 220
 - defining event as event rule 346
 - defining event as internetwork
 - dependency 343, 346
 - defining event rule based on CCMS
 - alerts 363
 - defining event rule based on IDocs 352
 - defining job 262
 - defining job dynamically 282
 - defining jobs 254
 - defining variant 257
 - deleting variant 257
 - dynamic job definition example 298
 - dynamic job definition parameter
 - description 285
 - dynamic job definition syntax 282
 - encrypting user password 249
 - event defined as event rule 346
 - event defined as internetwork
 - dependency 343, 346
 - event rule based on CCMS alerts, business
 - scenario 361, 362
 - event rule based on IDocs, business
 - scenario 351
 - event rule prerequisite 252
 - exporting calendars, business scenario 339
 - extended agent job 253
 - external command step definition
 - attribute 297
 - external program step definition
 - attribute 297
 - features 211
 - filtering events in security file 350
 - getting CCMS alert status by external
 - task 370
 - global options 231
 - IDoc record used in event rule 352
 - InfoPackage schedule options 317
 - installing ABAP modules 217
 - introduction 210, 213
 - job interception 306, 307, 308, 315
 - job interception and parent-child 305
 - job throttling 333
 - jobs dynamic definition 282
 - local options 234
 - logon group 251
 - mapping between definition and resolution
 - of internetwork dependency 344
 - monitoring event 252
 - National Language support 373
 - options file 213, 229, 248

- options National Language support 374
- parameters to define job 262
- placeholder job for internetwork
 - dependencies 343
- prerequisite to define event rule 252
- process chain schedule options 317
- r3evman command 252
- r3evmon configuration file 378
- re-running jobs 281
- refreshing variant 257
- rerunning job 280, 326
- RFC user password 225
- roles and responsibilities 214
- security file, filtering events 350
- setting a filter in security file 350
- setting interception criteria 309, 309
- setting variant 257
- supported code pages 375
- task string 262
- task string parameter 319
- task string parameters 262
- transaction PFCG 218
- transaction se38 334
- transaction su02 217
- updating variant 257
- variable substitution 298
- viewing variant 257
- SAP data connection 256
- SAP
 - event
 - committing by external task 345
 - defining as event rule 346
 - defining as internetwork dependency 343, 346
 - filtering in security file 350
 - monitoring 252
 - placeholder job for internetwork
 - dependencies 343
 - prerequisite to define a rule 252
 - prerequisites for defining as internetwork
 - dependency 342
 - r3evman command 252
 - r3evmon configuration file 378
 - raising 278
- SAP
 - job
 - defining dynamically 282
 - deleting 275
 - displaying details 274
 - editing 261
 - example of defining dynamically 298
 - killing 277
 - placeholder 343
 - task string 262
 - variable substitution 298
 - verifying status 274
- SAP R/3
 - calendars, exporting 340
 - exporting factory calendars, command 340
 - r3batch export function 340
- SAP
 - table criteria
 - setting template file 313
 - setting using the
 - Dynamic Workload Console
 - , BC-XBP 2.0
 - 310
 - setting using the
 - Dynamic Workload Console
 - , BC-XBP 3.0
 - 311

- SAP
 - user password
 - changing 249
 - encrypting 249
 - script job type
 - detailed syntax diagram 99
 - secure network communications 226
 - security
 - PeopleSoft
 - 188
 - WSSTAT command 58
 - security file, filtering
 - SAP
 - events
 - 350
 - security SAP
 - SNC 226
 - server group
 - balancing
 - SAP
 - workload
 - 276
 - SERVER_NAME_LIST, PeopleSoft
 - option
 - 189
 - SERVER, keyword of JOBREC 80, 84, 90
 - setting
 - job interception 309
 - job interception using interception criteria and template files 309
 - job throttling 334
 - SAP
 - table criteria on the workstation
 - 310
 - SAP
 - table criteria using the Dynamic Workload Console
 - 311
 - SAP
 - variant
 - 257
 - template file 313
 - setting trace levels for application server 394
 - settings
 - agent traces 25, 127
 - SNC 226
 - Solution Manager
 - job scheduling 387, 387, 391
 - direct 392
 - from job documentation 393
 - monitoring jobs 393
 - registering 387
 - SMSE adapter 387, 387, 391, 392, 393, 393
 - specific job types
 - sample JSDL files 17, 74
 - spool data
 - browsing 277
 - introduction 277
 - SQL queries for job metrics 398
 - DB2 398
 - DB2 for zOS 398
 - SSL connection
 - Z controller and IBM i agents 140
 - Z controller
 - and IBM Z Workload Scheduler Agents
 - 71
 - SSL connection (SAF)
 - Z controller
 - and

- IBM Z Workload Scheduler Agents
 - 68
- SSL connection with IBM i agents 140
- SSL connection with IBM Z Workload Scheduler Agents
 - 71
- STARTTIME keyword of AROPTS 147
- state
 - SAP
 - job
 - 276
- STATEMENT, keyword of JOBREC 80, 84
- status mapping
 - PeopleSoft
 - job
 - 198
- STDERROR, keyword of JOBREC 102
- STDINPUT, keyword of JOBREC 102
- STDOUTPUT, keyword of JOBREC 102
- submitting
 - supported agent job 186
- supported
 - code pages, SAP
 - 375
 - supported agent job
 - submitting 186
 - supported agent job defining with end-to-end scheduling 185
 - supported agents
 - job defining with Dynamic Workload Console
 - 183
- syntax
 - agent
 - access method 166
 - defining
 - SAP
 - jobs dynamically
 - 282
 - return code mapping 202
 - syntax diagrams, how to read ix
 - System scanner
 - JobManager.ini file 18

T

- table criteria
 - SAP
 - setting template file 313
 - setting using the Dynamic Workload Console
 - 310, 311
- task executors
 - configuring 139, 159
- task options
 - access method 166
- task string parameters
 - PeopleSoft
 - job
 - 197
 - SAP
 - job
 - 262, 262, 319
- template file
 - creating 313
 - description 313
 - setting placeholders for job interception 314
- temporary variants, examples 298
- throttling, job 333
- TIMEOUT, keyword of JOBREC 90

- trace and log files agent
 - agent twstrace syntax 25, 127
- trace configuration
 - IBM Workload Scheduler agent
 - 23, 126
- trace file
 - trace-psagent.log 209
 - trace-r3batch.log 209
- trace levels
 - application server
 - setting 394
- tracing utility
 - configuring 207
- tracking
 - PeopleSoft
 - job
 - 187
- transaction PFCG
 - SAP
 - 218
- transactions, SAP
 - PFCG 218
 - sa38 382
 - se16 309, 335
 - se37 343
 - se38 334, 382
 - sm69 297, 297
 - su02 217
- TRANSFERMODE, keyword of JOBREC 90
- TRANSFERTYPE, keyword of JOBREC 90
- troubleshooting 158
 - EE00778E 382
 - extended agent
 - job log 386
 - PeopleSoft, job submission fails 386
 - PeopleSoft, local options file rights 386
 - extend agent
 - Oracle, job submission fails 386
 - job throttling
 - alerts are not generated according to threshold 385
 - does not start 384, 384
 - does not stop 385
 - saving MTE properties generates message 385
 - longlink file 384
 - permission denied
 - job log 386
- r3batch 377
 - does not start 383
 - environment variables 383
 - modifying job step error 383, 383
 - monitoring IDoc events 380, 381, 381
 - monitoring
 - SAP
 - events
 - 376, 378, 378, 378
 - output with unreadable characters 376
 - scheduling
 - SAP
 - jobs
 - 382
 - system cannot intercept jobs 385
 - r3event
 - output with unreadable characters 376
 - r3evmon
 - monitoring events 378
 - monitoring events increases memory consumption 381
 - restarting process of subchain 377

- SAP
 - connection 375
 - error defining internetwork dependency 382
- TWS_MAX_WAIT_TIME,
- PeopleSoft
- option
 - 190
- TWS_MIN_WAIT_TIME,
- PeopleSoft
- option
 - 190
- TWS_PROMOTED_JOB 173
- TWS_RETRY,
- PeopleSoft
- option
 - 190
- twstrace syntax
 - agent log and trace files 25, 127
- TWSXA_INLINE_CI,
- PeopleSoft
- option
 - 190
- TWSXA_SCHED_METH,
- PeopleSoft
- option
 - 190

U

- u
 - jobthrottling parameter 336
- Unicode support
 - R/3
 - 227
- updating
 - SAP
 - variant
 - 257
- upload job type
 - detailed syntax diagram 85
- URL, keyword of JOBREC 93
- user authorizations
 - Business Warehouse InfoPackage 317
 - Business Warehouse process chain 317
- user password
 - encrypting on
 - PeopleSoft
 - 191
 - encrypting on
 - SAP
 - 249
- user return code on AS/400 121, 122
- user return code on i5/OS 121, 122
- user return code on IBM i 121, 122
- user security
 - setting 215
- USRCPU, keyword of USRREC 47
- USRNAM, keyword of USRREC 47
- USRPSW, keyword of USRREC 47
- USRREC initialization statement
 - definition 46, 47
 - making changes immediately effective 46
 - USRCPU keyword 47
 - USRNAM keyword 47
 - USRPSW keyword 47
- utility commands
 - filemonitor 107
 - filewatch 103

V

- variable
 - defining on agent 49

- job types with advanced options 49
- managing locally on agents 49, 52
- resolving on agent 49
- variable substitution 103
 - filewatch 103
 - jobs 103
 - SAP
 - 298
- variant
 - SAP
 - defining 257
 - deleting 257
 - refreshing 257
 - setting 257
 - updating 257
 - viewing 257
- variants
 - temporary 298
- verifying status
 - SAP
 - job
 - 274
- viewing
 - agent traces 25, 127
 - SAP
 - variant
 - 257
 - spool data 277

W

- web service executor 76
- web service job 76
- web service job type
 - detailed syntax diagram 91
- web service jobs
 - sample JSDL files 17, 74
- workload, balancing SAP R/3 276
- workstation for extended agent or dynamic agent, defining with
 - Dynamic Workload Console
 - 178
- workstation for extended agent, defining with
 - ISPF 182
- workstation for supported agent, defining with
 - command line 179
- workstation status 57
- workstation, defining with
 - Dynamic Workload Console
 - 178
- WRKDIR, keyword of JOBREC 102
- WSSTAT command 57

X

- XA jobs 17
- xajob executor 76
- xajob job 76
- xajob job type
 - detailed syntax diagram 95

Z

- Z controller
 - SAF SSL connection with
 - IBM Z Workload Scheduler Agents
 - 68
 - SSL connection with IBM i agents 140
 - SSL connection with
 - IBM Z Workload Scheduler Agents
 - 71
- z-centric workstation
 - browsing 56
 - defining 55
 - deleting 57

- modifying definition 56
- specifying list selection criteria 57
- zosHttpTimeout 42