IBM

IBM Workload Automation
Driving IBM Z Workload Scheduler
Version 10.2 (with APAR PH68599)

Note

Before using this information and the product it supports, read the information in Notices on page cdxcii.

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..... viii
List of Tables..... ix
About this publication..... xiv
Who should read this publication..... xiv
Conventions..... xiv
How to read syntax diagrams..... xv
Part I. Programming interfaces..... 18
Chapter 1. The Program Interface..... 19
Program interface samples..... 19
Related tools..... 19
Batch command interface tool..... 19
IBM Z Workload Scheduler control language..... 19
Communicating with EQQYCOM..... 20
Required data sets..... 21
Optional data set..... 21
Error messages..... 22
Parameter overview..... 22
Action code..... 23
Resource code..... 23
Data area..... 24
Argument names and values..... 24
Communication block..... 25
Return code..... 25
Sequence of requests..... 26
Data area description and format..... 26
Header format..... 26
Data record format..... 27
Date considerations..... 28
Internal date representation..... 28
Date arguments in PIF applications..... 28
Updating application description run cycles with PIF..... 29
Security considerations..... 29
Running user-written programs compiled for older scheduler versions..... 30
Overview of request types..... 31
DELETE request..... 32
Action code..... 33
Resource code..... 33
Data area..... 35
Arguments..... 35
Communication block address..... 44
Return code..... 44
EXECUTE request..... 45
Action code..... 45
Resource code..... 45
Data area..... 45
Arguments..... 45
Communication block address..... 45
Return code..... 45
INIT request..... 46
Action code..... 46
Resource code..... 46
Data area..... 46
Arguments..... 46
Communication block address..... 49
Return code..... 49
INSERT request..... 49
Action code..... 50
Resource code..... 50
Data area..... 52
Arguments..... 52
Communication block address..... 64
Return code..... 64
LIST request..... 64
Action code..... 64
Resource code..... 65
Data area..... 66
Arguments..... 67
Communication block address..... 80
Return code..... 80
MODIFY request..... 80
Action code..... 80
Resource code..... 80
Data area..... 81
Arguments..... 81
Communication block address..... 92
Return code..... 92
OPTIONS request..... 92
Action code..... 93
Resource code..... 93
Data area..... 93
Arguments..... 94
Communication block address..... 97
Return code..... 97
REPLACE request..... 97
Action code..... 98
Resource code..... 98
Data area..... 99
Arguments..... 99
Communication block address..... 99

Return code..... 100
RESET request..... 100
Action code.....100
Resource code.....100
Data area.....100
Arguments..... 100
Communication block address..... 100
Return code..... 101
SELECT request..... 101
Action code.....101
Resource code.....101
Data area.....104
Arguments..... 105
Communication block address..... 116
Return code..... 116
SETSTAT request..... 117
Action code.....117
Resource code.....117
Data area.....117
Arguments..... 117
Communication block address..... 118
Return code..... 118
TERM request.....118
Action code.....118
Resource code.....118
Data area.....118
Arguments..... 119
Communication block address..... 119
Return code..... 119
JCL preparation using PIF.....119
Substituting variables..... 119
Simulating variable substitution.....121
Chapter 2. Application Programming Interface.....122
Communicating with IBM Z Workload Scheduler..... 122
CPI-C support provided by IBM Z Workload Scheduler..... 123
API buffer layouts.....124
APP - Fixed section.....126
APPOBJ - Object section.....128
APPSEL - Selection section.....131
APPVAL - Selection value section.....132
APPFLD - Field section..... 132
APPDAT - Data section..... 133
Specifying object names.....134
Selecting object instances..... 135
Specifying key types..... 135
Specifying selection criteria..... 136
Broadcasting events..... 137
Selecting object fields to update or retrieve..... 137
Return codes and reason codes generated by IBM Z Workload Scheduler.....137
Return codes and reason codes generated in the fixed section (APP)..... 138
Return codes and reason codes generated in the object section (APPOBJ).....139
Security..... 140
APPC and RACF®..... 140
IBM Z Workload Scheduler and RACF®.....141
Part II. Programming tools.....143
Chapter 3. Batch command interface tool..... 144
Online tools..... 144
The batch command interface..... 144
Input to batch command interface..... 144
BCIT output.....147
Instructions..... 149
COPY..... 151
DELETE.....154
EXPORT..... 169
GROUPDEF support.....170
IMPORT..... 171
INSERT..... 173
LIST..... 190
LISTSTAT.....204
MODIFY..... 207
OPTIONS..... 217
REPLACE..... 219
SELECT.....219
SETSTAT..... 232
Chapter 4. Control Language (OCL).....236
What you can do using OCL..... 236
Advantages of OCL.....236
Summary of OCL instructions.....237
Customizing OCL..... 240
Specifying the initialization parameters..... 243
Example 1..... 243
Example 2..... 243
Example 3..... 244
Example 4..... 244
Obtaining access authorization..... 244
Logging executed instructions..... 245
Specifying OCL instructions..... 246
Specifying input arrival dates and times.....247
Description of OCL instructions..... 251
INIT..... 319
JSUACT..... 320
KILLJOB..... 321

KILLREC....323
LABEL....326
MODCOND....326
MODOP....329
NOP....335
OPSTAT....337
PROMPTN....341
PROMPTY....343
RELEASE....346
RELOP....349
RELSUCC....351
SET....353
Concatenation operators....354
Built-In Functions....354
SETUPD....355
SRSTAT....356
UNNOP....360
UPD....362
WSSTAT....362
WTO....364
Requirements....365
Sample job and procedure....365
EQQYRJCL sample job....365
EQQYRPRC sample procedure....366
Messages....367
Message format....367
Chapter 5. Driving IBM® Z Workload Scheduler with REST APIs....369
Appendix A. Program interface record format....370
TOD fields....370
Application description (resource codes AD, ADCOM)....371
ADAPD - Application dependency segment....372
ADCIV - Interval definition for conditional external predecessor segment....373
ADCOM - Common segment....374
ADDEP - Dependency segment....375
ADCNC - Condition segment....376
ADCNS - Condition dependency segment....376
ADEXT - Extended name segment....377
ADKEY - Key segment....377
ADLAT - Operation user-defined late segment....378
ADOP - Operation segment....378
ADRE - Remote job information segment....381
ADRUN - Run cycle segment....382
ADSAI - Operation system automation information segment....385
ADSR - Special resource segment....386
ADUSF - User field segment....387
ADVDD - Operation variable values....387
ADXIV - Interval definition for external predecessor segment....389
All workstations closed (resource code AWSCL)....390
AWSCL - All workstations closed interval segment....390
Calendar (resource codes CL, CLCOM)....391
CLCOM - Common segment....391
CLSD - Specific date segment....392
CLWD - Weekday segment....392
Current plan condition (resource codes CPCOND, CPCONDCO)....393
CPCOND - Condition segment....393
CPSIMP - Condition dependency segment....394
Current plan occurrence (resource code CPOC, CPOCCOM)....394
CPOC - Current plan occurrence segment....395
CPOCPRE - Occurrence predecessor segment....398
CPOCSUC - Occurrence successor segment....399
Current plan operation (resource codes CPOP, CPOPCOM)....399
CPCPR - Conditional predecessor segment....400
CPCSU - Conditional successor segment....401
CPEXT - Operation extended name segment....401
CPLAT - Operation user-defined late info segment....401
CPOP - Common segment....403
CPOPSRU - Special resource usage segment....410
CPPRE - Predecessor segment....411
CPREND - Distributed remote job info segment....413
CPRENZ - z/OS remote job info segment....413
CPSAI - Operation system automation information segment....415
CPSUC - Successor segment....416
CPSR - Special resource segment....416
CPREC - Operation recovery segment....417
Current plan status (resource code CPST)....418
CPST - Common segment....418
Current plan operation user field (resource codes CPUSRF, CPUSRFELEM)....420
CPUSRF - Operation user field segment....420
Current plan workstation (resource codes CPWS, CPWSCOM)....420
CPWS - Common segment....421
CPIVL - Current plan workstation open interval segment....423
CPOPT - workstation description record segment....424

Current plan virtual workstation destination (resource codes CPWSV, CPWSVCOM)..... 425
CPWSV - Common segment..... 425
CPVIVL - Current plan virtual workstation destination open interval segment..... 427
Operation critical successors (resource code CRITSUCS)..... 428
Current plan special resource (resource codes CSR, CSRCOM)..... 430
CSRCOM - Current plan resource common segment..... 430
CSRIVL - Current plan special resource interval segment..... 433
CSRIWS - Current plan resource interval "connected" workstation..... 433
CSRDWS - Current plan resource default "connected" workstation..... 434
ETT - Event triggered tracking criteria segment..... 434
Dates generated by run cycle rules (resource code GENDAYS)..... 435
JCL setup variables (resource codes JCLPREP, JCLPREPA..... 435
JSVC - Common segment..... 435
JSVV - Variable definition segment..... 436
JCL variable table (resource codes JCLV, JCLVCOM)..... 436
JCLVC - Common segment..... 437
JCLVV - Variable definition segment..... 437
JCLVD - Dependency segment..... 438
Job control language (resource codes JS, JSCOM)..... 439
JS - Job control language segment..... 439
Job log (resource code JLCOM)..... 441
JLCOM - Common segment..... 441
Long-term plan occurrence (resource codes LTOC, LTOCCOM)..... 441
LTOC - Common segment..... 442
LTOP - Operation segment..... 444
LTCPRE- Conditional predecessor segment..... 444
LTCSUC- Conditional successor segment..... 445
LTPRE - Predecessor segment..... 445
LTSUC - Successor segment..... 446
LTEXT - External run cycle group for variable duration and deadline..... 446
Operator instruction (resource codes OI, OICOM)... 447
OI - Operator instruction segment..... 447
Period (resource codes PR, PRCOM)..... 448
PR - Period segment..... 448
Run cycle group (resource codes RG, RGCOM)..... 450
RGCOM - Common segment..... 450
RGRUN - Run cycle segment..... 451
Special resource (resource codes SR, SRCOM)..... 453
Workstation description (resource codes WS, WSCOM)..... 456
WSCOM - Common segment..... 456
WSDEST – Destination segment..... 458
WSIVL - Open interval segment..... 459
WSSD - Specific date segment..... 459
WSWD - Weekday segment..... 460
WSAM - Workstation access method segment..... 460
WSOPT - workstation description record segment..... 461
Virtual workstation destination description (resource codes WSV, WSVCOM)..... 462
WSVCOM - Common segment..... 462
WSVIVL - Open interval segment..... 464
WSVSD - Specific date segment..... 464
WSVWD - Weekday segment..... 465
Appendix B. API object fields..... 466
Current plan status object..... 466
Current plan operation object..... 467
Current plan special resource object..... 477
Current plan workstation object..... 478
Current plan open interval object..... 480
Current plan operation event object..... 481
Current plan OPINFO event object..... 484
Current plan special resource event object..... 485
Current plan backup event object..... 486
Current plan workstation event object..... 487
Appendix C. Sample library (SEQQSAMP)..... 489
IBM Z Workload Scheduler Application Programming Interface..... 489
API buffer examples..... 490
IBM Z Workload Scheduler program interface..... 490
JS data set maintenance..... 490
JCL variable substitution..... 490
Current plan and LTP actions..... 491
Other PIF samples..... 491
Notices..... cdxcii
Index..... 496

List of Figures

Figure 1: Program interface parameters.....23

Figure 2: Program interface arguments in TSO command notation..... 25

Figure 3: Program interface data area example.....27

Figure 4: Example of arguments for processing a list.....105

Figure 5: Example of a send buffer layout for a GET request.....125

Figure 6: Application flow example.....243

List of Tables

Table 1: Comparison of Date Representations....29
Table 2: Access Authority for Program Interface Requests....29
Table 3: Program Interface Resources and the Corresponding IBM Z Workload Scheduler Fixed Resources Used for Checking Authorization....30
Table 4: Records Using a Common Segment....32
Table 5: Delete AD Arguments....35
Table 6: Delete AWSCL Arguments....36
Table 7: Delete CL Arguments....36
Table 8: Delete CPCOND Arguments....36
Table 9: Delete CPOC Arguments....36
Table 10: Delete CPOCPRE Arguments....37
Table 11: Delete CPOCSUC Arguments....37
Table 12: Delete CPOP Arguments....37
Table 13: Delete CPPRE Arguments....37
Table 14: Delete CPSIMP Arguments....38
Table 15: Delete CPSR Arguments....39
Table 16: Delete CPSUC Arguments....39
Table 17: Delete CPUSRF Arguments....40
Table 18: Delete ETT Arguments....40
Table 19: Delete IVL Arguments....40
Table 20: Delete JCLV Arguments....40
Table 21: Delete JL Arguments....41
Table 22: Delete JS, JSCOM Arguments....41
Table 23: Delete LTOC Arguments....41
Table 24: Delete LTCPRE Arguments....41
Table 25: Delete LTPRE Arguments....42
Table 26: Delete OI Arguments....42
Table 27: Delete PR Arguments....43
Table 28: Delete RG Arguments....43
Table 29: Delete SR Arguments....43
Table 30: Delete VIVL Arguments....44
Table 31: Delete WS Arguments....44
Table 32: Delete WSV Arguments....44
Table 33: Insert CPLAT Arguments....53
Table 34: Insert CPOC Arguments....54
Table 35: Insert CPOCPRE Arguments....55
Table 36: Insert CPOCSUC Arguments....55
Table 37: Insert CPCOND Arguments....56
Table 38: Insert CPOP Arguments....56
Table 39: Insert CPPRE Arguments....58
Table 40: Insert CPSAI Arguments....58
Table 41: Insert CPSIMP Arguments....59
Table 42: Insert CPSR Arguments....60
Table 43: Insert CPSUC Arguments....61
Table 44: Insert CPUSRF Arguments....61
Table 45: Insert IVL Arguments....61
Table 46: Insert JCLPREP Arguments....62
Table 47: Insert JCLV Arguments....62
Table 48: Insert LTOC Arguments....62
Table 49: Insert LTPRE Arguments....63
Table 50: Insert VIVL Arguments....63
Table 51: List ADCOM and ADKEY Arguments....69
Table 52: List AWSCL Arguments....70
Table 53: List CLCOM Arguments....70
Table 54: List CPCONDCO Arguments....70
Table 55: List CPOPCOM Arguments....71
Table 56: List CPOPSRU Arguments....73
Table 57: List CPWSCOM Arguments....74
Table 58: List CPWSVCOM Arguments....74
Table 59: List CRITSUCS Arguments....75

Table 60: List CSRCOM Arguments..... 75
Table 61: List ETT Arguments.....76
Table 62: List GENDAYS Arguments..... 76
Table 63: List JCLVCOM Arguments.....77
Table 64: List JLCOM Arguments..... 77
Table 65: List JSCOM Arguments..... 77
Table 66: List LTOCCOM Arguments.....78
Table 67: List OICOM Arguments.....78
Table 68: List PRCOM Arguments.....78
Table 69: List RGCOM, RGKEY Arguments..... 78
Table 70: List SRCOM Arguments.....79
Table 71: List WSCOM Arguments.....79
Table 72: List WSVCOM Arguments..... 79
Table 73: Modify CPCOND Arguments..... 82
Table 74: Modify CPEXT Arguments.....82
Table 75: Modify CPOC Arguments..... 83
Table 76: Modify CPOP Arguments..... 83
Table 77: Modify CPREND Arguments.....86
Table 78: Modify CPRENZ Arguments..... 87
Table 79: Modify CPSAI Arguments.....87
Table 80: Modify CPUSRF Arguments..... 88
Table 81: Modify CPWS Arguments.....88
Table 82: Modify CPWSV Arguments..... 89
Table 83: Modify CSR Arguments..... 90
Table 84: Modify IVL Arguments.....91
Table 85: Modify LTOC Arguments..... 91
Table 86: Modify VIVL Arguments..... 92
Table 87: Replace AD Arguments.....99
Table 88: Select AD, ADCOM Arguments.....106
Table 89: Select AWSCL Arguments.....107
Table 90: Select CL, CLCOM Arguments.....107
Table 91: Select CPCOND, CPCONDCO Arguments..... 108
Table 92: Select CPOC Arguments.....108
Table 93: Select CPOP, CPOPCOM Arguments.....109
Table 94: Select CPUSRF Arguments.....111
Table 95: Select CPWS, CPWSCOM Arguments.....111
Table 96: Select CPWSV, CPWSVCOM Arguments..... 111
Table 97: Select CSR, CSRCOM Arguments..... 112
Table 98: Select ETT Arguments.....112
Table 99: Select JCLPREP Arguments.....112
Table 100: Select JCLPREPA Arguments..... 113
Table 101: Select JCLV, JCLVCOM Arguments.....113
Table 102: Select JLCOM Arguments.....113
Table 103: Select JS, JSCOM Arguments.....114
Table 104: Select LTOC, LTOCCOM Arguments.....114
Table 105: Select OI, OICOM Arguments.....114
Table 106: Select PR, PRCOM Arguments.....114
Table 107: Select RG, RGCOM Arguments.....115
Table 108: Select SR, SRCOM Arguments..... 115
Table 109: Select WS, WSCOM Arguments..... 115
Table 110: Select WSV, WSVCOM Arguments.....116
Table 111: Setstat CPSIMP Argument.....117
Table 112: Contents of a Send Buffer.....125
Table 113: App-Fixed Section.....126
Table 114: APPOBJ-Object Section.....129
Table 115: APPSEL-Selection Section..... 131
Table 116: APPVAL-Selection Value Section.....132
Table 117: APPFLD-Field Section.....133
Table 118: APPDAT-Data Section.....134
Table 119: API Object Names..... 134
Table 120: Operators That You Can Specify in the APPSEL Section.....136
Table 121: Subresource Protection for Requests through the API..... 141

Table 122: Positional parameters that can be passed with the Batch Command Interface.....145

Table 123: Access Authorizations..... 244

Table 124: Input Arrival Date and Time Keywords..... 247

Table 125: How OCL Uses the IADATE, IATIME, and IA Keywords.....248

Table 126: IBM Z Workload Scheduler-supplied Variables.....250

Table 127: Keywords Used in the Add Instruction.....251

Table 128: Keywords used in the Addcond Instruction.....254

Table 129: Keywords used in the Addop Instruction.....258

Table 130: Keywords used in the Addpred Instruction..... 264

Table 131: Keywords used in the Addres Instruction..... 266

Table 132: Keywords used in the Addsimp Instruction.....269

Table 133: Keywords used in the Chgextname Instructions..... 273

Table 134: Keywords used in the Chgjob Instructions.....275

Table 135: Keywords used in the Chgopsai Instructions...277

Table 136: Keywords used in the Chkappl Instruction.....280

Table 137: Chkdate Instruction Variables.....285

Table 138: Keywords used in the Compl Instruction.....294

Table 139: Keywords used in the Del Instruction..... 297

Table 140: Keywords used in the Delcond Instruction.....299

Table 141: Keywords used in the Delpred Instruction..... 301

Table 142: Keywords used in the Delres Instruction..... 304

Table 143: Keywords used in the Delsimp Instruction.....306

Table 144: Keywords used in the Force Instruction..... 310

Table 145: Keywords used in the Hold Instruction.....314

Table 146: Keywords used in the Init Instruction..... 319

Table 147: Keywords used in the JSUACT Instruction.....320

Table 148: Keywords used in the Killjob Instruction..... 321

Table 149: Keywords used in the Killrec Instruction..... 324

Table 150: Keywords used in the Modcond Instruction.... 326

Table 151: Keywords used in the Modop Instructions.....329

Table 152: Operations Details that can be modified..... 333

Table 153: Keywords used in the Nop Instruction.....336

Table 154: Keywords used in the Opstat Instruction.....337

Table 155: Keywords used in the PROMPTN Instruction...341

Table 156: Keywords used in the PROMPTY Instruction... 344

Table 157: Keyword used in the Release Instruction..... 346

Table 158: Keywords used in the Relop Instruction..... 349

Table 159: Keywords used in the Relsucc Instruction..... 351

Table 160: Keywords used in the Srstat Instruction.....356

Table 161: Keywords used in the Unnop Instruction.....360

Table 162: Keywords used in the Wsstat Instruction..... 363

Table 163: Clock value setting at the start of different years..... 370

Table 164: Clock value setting at different time interval... 370

Table 165: ADAPD Control Block..... 372

Table 166: ADCIV Control Block.....373

Table 167: ADCOM Control Block..... 374

Table 168: ADDEP Control Block.....375

Table 169: ADEXT Control Block.....377

Table 170: ADLAT Control Block..... 378

Table 171: ADOP Control Block.....379

Table 172: ADRE Control Block..... 382

Table 173: ADRUN Control Block..... 384

Table 174: Run Cycle Offsets..... 385

Table 175: Rule Definition.....385

Table 176: ADSAI Control Block.....386

Table 177: ADSR Control Block..... 386

Table 178: ADUSF Control Block.....387

Table 179: ADVDD Control Block..... 388

Table 180: ADXIV Control Block.....389

Table 181: AWSCL Control Block..... 390

Table 182: CLCOM Control Block....391
Table 183: CLSD Control Block.... 392
Table 184: CLWD Control Block.... 393
Table 185: CPOC Control Block....395
Table 186: CPOCPRE Control Block....398
Table 187: CPOCSUC Control Block....399
Table 188: CPEXT Control Block....401
Table 189: CPLAT Control Block.... 402
Table 190: CPOP Control Block....403
Table 191: CPOPSRU Control Block....410
Table 192: CPPRE Control Block....412
Table 193: CPREND Control Block.... 413
Table 194: CPRENZ Control Block.... 414
Table 195: CPSAI Control Block....415
Table 196: CPSUC Control Block....416
Table 197: CPSR Control Block.... 416
Table 198: CPREC Control Block....417
Table 199: CPST Control Block.... 419
Table 200: CPUSRF Control Block....420
Table 201: CPUSRFELEM Control Block.... 420
Table 202: CPWS Control Block.... 421
Table 203: CPIVL Control Block.... 423
Table 204: CPOPT Control Block.... 424
Table 205: CPWSV Control Block....425
Table 206: CPVIVL Control Block....427
Table 207: CRITSUCS Control Block.... 428
Table 208: CSRCOM Control Block.... 431
Table 209: CSRIVL Control Block....433
Table 210: CSRIWS Control Block....433
Table 211: CSRDWS Control Block.... 434
Table 212: ETT Control Block....434
Table 213: GNDAY Control Block.... 435
Table 214: JSVC Control Block.... 436
Table 215: JSVV Control Block.... 436
Table 216: JCLVC Control Block.... 437
Table 217: JCLVV Control Block.... 438
Table 218: JCLVD Control Block.... 439
Table 219: JS Control Block.... 440
Table 220: JLCOM Control Block.... 441
Table 221: LTOC Control Block....442
Table 222: LTOP Control Block....444
Table 223: LTPRE Control Block....445
Table 224: LTSUC Control Block....446
Table 225: LTEXT Control Block....446
Table 226: OI Control Block....447
Table 227: PR Control Block....449
Table 228: Period Origin Dates....449
Table 229: Period Interval End Dates....450
Table 230: RGCOM Control Block.... 450
Table 231: RGRUN Control Block.... 452
Table 232: Rule Definition....453
Table 233: SRCOM Control Block....454
Table 234: SRIVL Segment.... 455
Table 235: SRIWS Segment.... 455
Table 236: SRDWS Segment....455
Table 237: WSCOM Control Block....457
Table 238: WSDEST Control Block....459
Table 239: WSIVL Control Block.... 459
Table 240: WSSD Control Block.... 459
Table 241: WSWD Control Block.... 460
Table 242: WSAM Control Block.... 461
Table 243: WSOPT Control Block....461
Table 244: WSVCOM Control Block.... 463
Table 245: WSVIVL Control Block.... 464

Table 246: WSVSD Control Block.....465

Table 247: WSWD Control Block.....465

Table 248: CP_STATUS Object Fields..... 467

Table 249: CP_OPERATION Object Fields.....468

Table 250: CP_RESOURCE Object Fields..... 477

Table 251: CP_WORK_STATION Object Fields.....478

Table 252: CP_OPEN_INTERVAL Object Fields..... 481

Table 253: CP_OPER_EVENT Object Fields..... 481

Table 254: CP_OPINFO_EVENT Object Fields..... 484

Table 255: CP_SR_EVENT Object Fields.....485

Table 256: BACKUP_EVENT Object Fields.....487

Table 257: CP_WS_EVENT Object Fields.....487

Table 258: SEQQSAMP Library Members for Programming Interfaces and the API.....489

About this publication

Developer's Guide: Driving IBM Z Workload Scheduler shows you how to use the programming interfaces to IBM Z Workload Scheduler to help you plan, schedule, and monitor work in the production department of your computer installation.

Your workload can run on various platforms, but you control it from a central z/OS® system that runs the IBM Z Workload Scheduler controller.

This guide is part of a set of guides that allows you to program many aspects of working with the products in the IBM Workload Automation family. These guides comprise:

- *IBM Workload Automation: Developer's Guide: Driving IBM Z Workload Scheduler*
- *IBM Workload Automation: Developer's Guide: Extending IBM Workload Automation*
- *Workload Automation Programming Language for Z User's Guide and Reference*

Note: If you control your Z controller using Dynamic Workload Console, information about the programming interfaces you can use with the Dynamic Workload Console are available in both of the other Developer's Guides in the set.

The term *scheduler*, when used in this publication, refers to IBM Z Workload Scheduler. The term DB2®, when used in this publication, refers to DATABASE 2 and DB2 Universal Database™.

The term *z/OS*® is used in this publication to mean z/OS® and OS/390® operating systems. Where the term *OS/390*® appears, the related information applies only to OS/390® operating systems.

Who should read this publication

This publication is for users who write application programs that request services from IBM Workload Automation.

This publication documents the programming interface (PIF) and the application programming interface (API). To use PIF you must know job control language (JCL) and have a good working knowledge of a programming language, for example, assembler or PL/I. You can use programming languages that support z/OS® and OS/390® linkage conventions and that can load and delete an assembler program.

To use the API, you require a knowledge of Advanced Program-to-Program Communication (APPC). You must be able to write application transaction programs (ATPs) that use the services of APPC. Because the API is implemented using a subset of CPI-C (Common Programming Interface for Communications) verbs, you must be able to write ATPs that use CPI-C.

Conventions used in this publication

Conventions used in this publication.

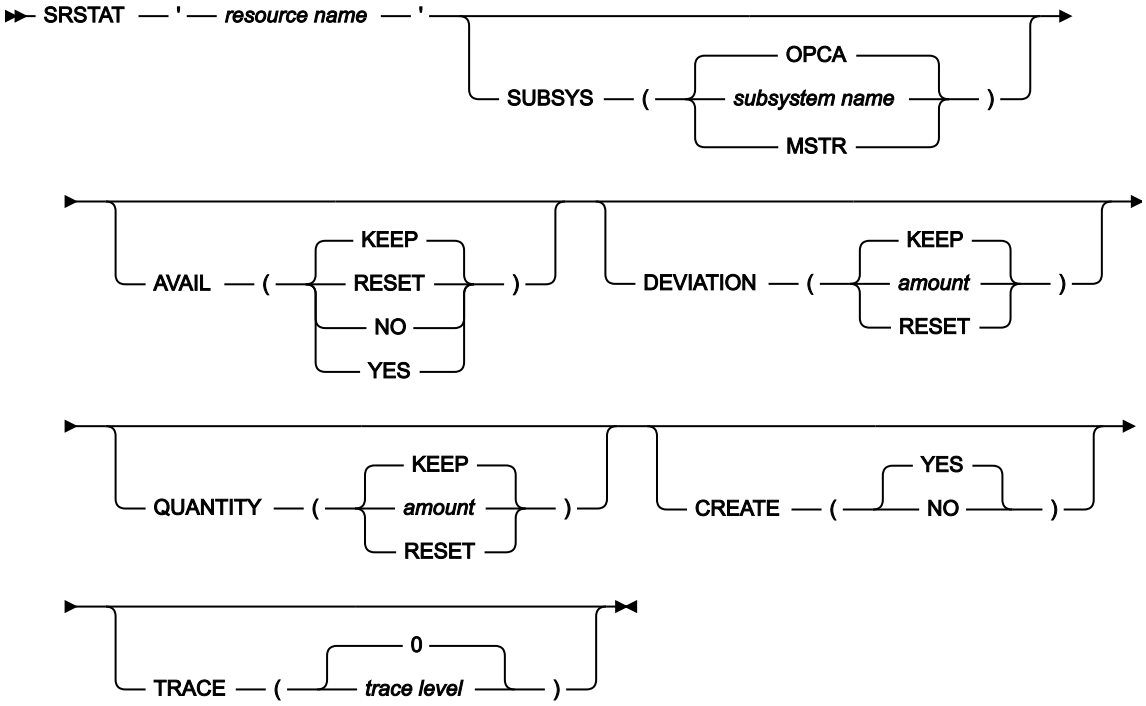
The publication uses several typeface conventions for special terms and actions. Technical changes to the text are indicated by a vertical line to the left of the change. These conventions have the following meanings:

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	*Application*

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.

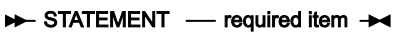
The statement is continued from a previous line.

The statement ends here.

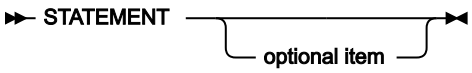
Read the syntax diagrams from left to right and from top to bottom, following the path of the line.

These are the conventions used in the diagrams:

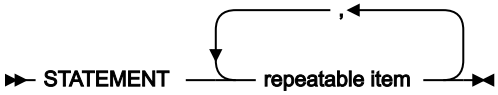
- Required items appear on the horizontal line (main path):



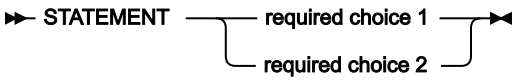
- Optional items appear below the main path:



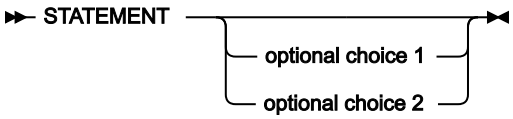
- An arrow returning to the left above the item indicates an item that you can repeat. If a separator is required between items, it is shown on the repeat arrow.



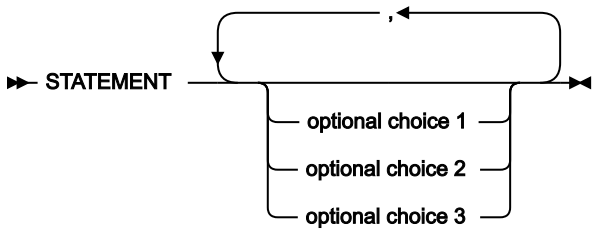
- If you can choose from two or more items, they appear vertically in a stack.
  - If you must choose one of the items, one item of the stack appears on the main path:

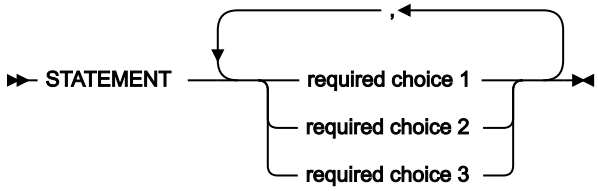


  - If choosing one of the items is optional, the entire stack appears below the main path:

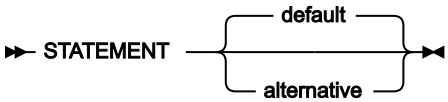


  - A repeat arrow above a stack indicates that you can make more than one choice from the stacked items:

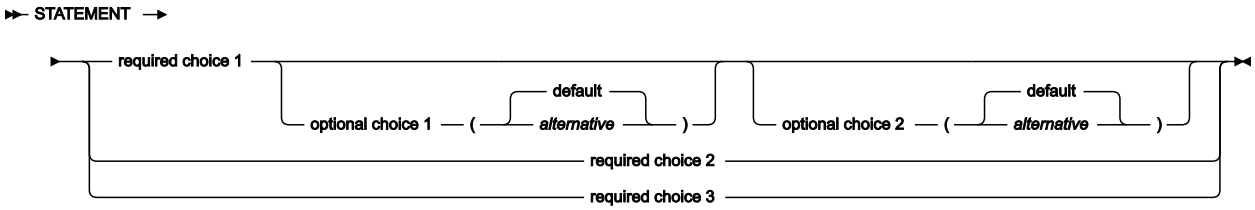




- Parameters that are above the main line are default parameters:



- 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:



Part I. Programming interfaces

Chapter 1. The program interface (PIF)

About this task

This chapter describes the IBM Z Workload Scheduler program interface (PIF), which lets a user-written program issue various requests to the IBM Z Workload Scheduler subsystem. For example, you can automate actions that you perform through the IBM Z Workload Scheduler dialogs.

The program interface supports these basic requests:

Database requests

- Read and update information from the application description and operator instruction databases.
- Read information from the workstation description and calendar databases.

LTP requests

Read and update occurrences in the LTP data set.

Current plan requests

Read and update this information in the current plan data set:

- Occurrences
- Operations
- Workstations

The program interface is supported using an IBM Z Workload Scheduler-supplied communication subroutine, EQQYCOM.

Program interface samples

The IBM Z Workload Scheduler sample library shipped with the IBM Z Workload Scheduler programs contains many sample programs that use the program interface function. These programs will execute successfully with a few minor changes to suit your installation. You can continue to run them as they are, or use them as a model to create your own programs. For a description of the PIF sample members provided in the SEQQSAMP library, see Sample library (SEQQSAMP) on page 489.

Related tools

IBM Z Workload Scheduler is delivered with some tools that take advantage of the PIF. These tools are provided as additional aids in the process of automating workload management with IBM Z Workload Scheduler.

Batch command interface tool

A Batch Command Interface tool is supplied to perform some of the actions supplied by the PIF interface by means of a batch command interface. For running this tool, refer to the EQQYCBAT sample.

IBM Z Workload Scheduler control language

About this task

The IBM Z Workload Scheduler Control Language (OCL) is a REXX program, EQQOCL, delivered in the IBM Z Workload Scheduler SEQQMISC library, that allows some of the IBM Z Workload Scheduler data to be handled from a simple REXX-like interface. You use the tool by running EQQOCL as a compiled REXX program in a batch TSO environment. You can choose to run it in a separate job or as a step in your production JCL.

The tool is described in Control Language (OCL) on page 236. The following members of IBM Z Workload Scheduler sample libraries are also supplied to help you use the tool:

EQQYRJCL

A sample JCL to run the tool

EQQYRPRC

A sample procedure to invoke EQQOCL

EQQYRMSG

Contains the messages issued by the EQQOCL tool

EQQYRPRM

A sample input parameter member

The OCL tool requires the IBM® Compiler Libraries for REXX/370 Version 1.3.0, Program Number 5695-014. The Compiler Libraries must be installed on the system, and the TSO Compiler Programming Table, IRXCMPTM, must be customized and the Compiler Libraries modules must be made available to TSO. Refer to the Compiler Libraries for REXX/370 manuals for more information.

The OCL tool calls the EQQSTOR program, available in the IBM Z Workload Scheduler sample library member EQQRXSTG. If the WTO function is used, OCL calls the IPOWTO program, available in the scheduler sample library member EQQOCWTO. If the UPD function is used, OCL calls the EQQPIFT program available in IBM Z Workload Scheduler sample library member EQQPIFJV. This sample program requires the PL/I compiler and runtime libraries. The EQQSTOR program, and optionally IPOWTO and EQQPIFT programs, must be link-edited and made available to OCL at runtime.

Communicating with EQQYCOM

Requests to IBM Z Workload Scheduler to perform particular actions are calls to EQQYCOM, using normal z/OS® linkage conventions.

You must create a program that calls EQQYCOM and provide it with the necessary instructions, such as a parameter list, to enable IBM Z Workload Scheduler to perform the required action. With each call to EQQYCOM, you can make one IBM Z Workload Scheduler request.

EQQYCOM can be linked with the modules from which it is called, or it can be created as a separate load module and control passed to it using the link macro. If you create EQQYCOM as a separate load module and frequent calls are required, you should, for performance reasons, consider placing EQQYCOM in the link-pack area. All modules in the same job-step must be in an APF-authorized library. The first module loaded at the start of the job-step must also be link-edited with the APF-authorized attribute. In the TSO or TSO-batch environment, you need not have the PIF program authorized.

Details of your request to IBM Z Workload Scheduler are a parameter list that you pass to EQQYCOM. Before passing control to EQQYCOM, you must load the address of your parameter list into general purpose register 1.

Note: If you want to run a PIF program from an IBM Z Workload Scheduler dialog, ensure that your PIF program is invoked as a separate task. Otherwise, your dialog session will end when the PIF program has completed. For example, you can run a REXX exec that runs your PIF program using the ATTACH command.

Calling EQQYCOM from exits that are taken by the controller address space is not supported and will cause unpredictable results if attempted.

Required data sets

When you use the program interface, allocate the data sets identified by these ddnames to each address space where your program runs:

EQQMLIB

IBM Z Workload Scheduler message library.

EQQMLOG

Data Set for messages from the program interface.

An extra message log is required for each additional INIT request made before a TERM request, or when PIF is invoked by a program or started task that already uses the EQQMLOG data set allocated to the caller address space. For detailed information, see INIT request on page 46.

Optional data set

EQQYPARM

Parameter file for specifying the INIT initialization statement. EQQYPARM must reference a sequential data set or a member of a partitioned data set whose logical record length is 80 bytes.

The //EQQYPARM DD * notation followed by INIT statements is not allowed and might cause a system X'0C4' abend.

Note:

1. It is important that you also allocate the IBM Z Workload Scheduler diagnostic data set, EQQDUMP. Debugging information is written to this data set, for example, IBM Z Workload Scheduler control blocks and traces.
2. If you plan to run PIF applications many times per day from a long-running non-TSO address space (for example, NetView®), to prevent a storage shortage do not specify the EQQYPARM ddname. Instead, specify the parameters either in the PIF application or in the controller INTFOPTS initialization statement. When you

run a PIF application by specifying the EQQYPARM ddname, a TSO environment must be established each time and some of the resources remain allocated until the task ends. This might lead to a storage shortage, if the commands are issued many times.

Error messages

When an error occurs in a request, messages are always written to the message log data set allocated to the caller address space. The data set is either EQQMLOG or that specified in the MLOGDDN argument of the INIT request. In certain cases, messages are also written to the EQQMLOG data set allocated to the IBM Z Workload Scheduler subsystem to which your requests are directed.

Errors related to the request itself (for example, a spelling error in a parameter argument) result in a message written only to the message log allocated to the caller address space.

Errors related to the IBM Z Workload Scheduler subsystem (for example, an error detected by IBM Z Workload Scheduler data validation) result in a brief message to the caller message log. A more detailed message about the error is written to the EQQMLOG allocated to the IBM Z Workload Scheduler subsystem.

Parameter overview

The parameter list contains the necessary information for one request. Figure 1: Program interface parameters on page 23 illustrates the basic structure of the parameter list and the addressing linkage to it.

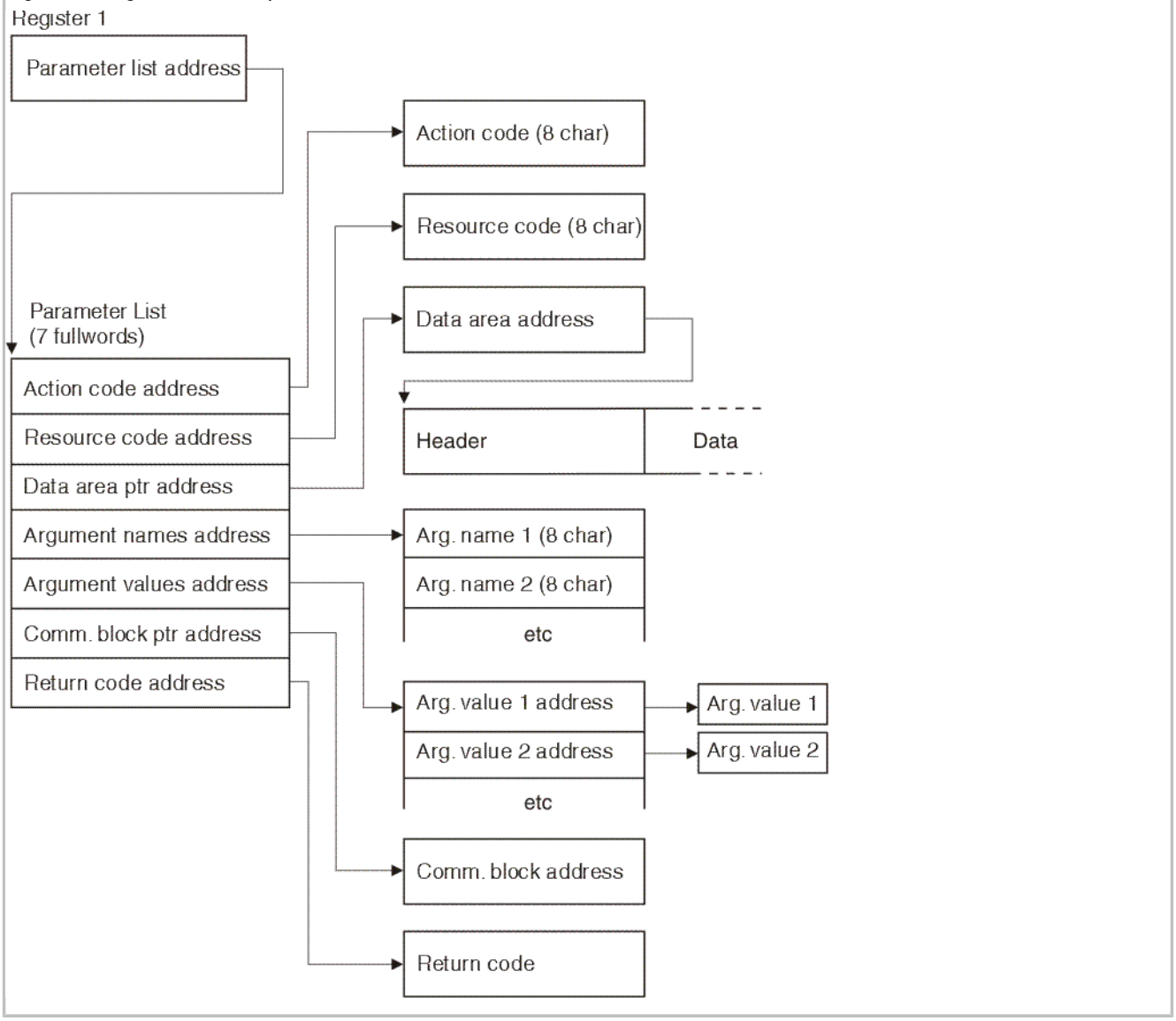
The parameter list must always consist of seven fullwords, representing the seven parameter types outlined here. Not all parameters are required for some requests, in which case you must set the parameter value to hexadecimal zeros. A character-type parameter value that contains blanks also indicates that the parameter is omitted. The parameter list itself must not contain zeros.

Figure 1: Program interface parameters on page 23 describes the parameter values that are referenced by the parameter address list.

An overview of the parameters follows. More detailed descriptions of the required parameters are given with the description of each request type.

Example

Figure 1. Program interface parameters



Action code

The first fullword in the parameter list is the address of the action code.

The action code describes the action to be performed. For example, to update a record in one of the IBM Z Workload Scheduler databases, you use the REPLACE action code.

Resource code

The second fullword in the parameter list is the address of the resource code.

The resource code describes the IBM Z Workload Scheduler resource that the request is directed to. For example, to replace an application description in the AD database, you use the AD resource code.

Data area

The third fullword in the parameter list is the address of a fullword that contains the address of a data area.

A data area consists of the actual data involved in the request. If you are retrieving information from a database, EQQYCOM places the record in this area and provides its address in the fullword whose address is in the parameter list.

Note: EQQYCOM might use the same piece of data area storage for successive data retrieval requests, overwriting the storage area used for the previous request each time. Therefore, your program must copy the information to its own storage area if it must be kept during later retrieval requests.

If you are writing information to a database, your program must build its own data area and provide its address in the fullword whose address is in the parameter list.

Attention: When the data area is not used, the data area address in the parameter list must be set to hexadecimal zero; failure to do so might cause unpredictable results. Some programming languages might require special coding to achieve this task; for example, in PL/I programs, use the SYSNULL built-in function.

The data area consists of a header, which describes the structure of the data record, and the data itself. For a detailed description, see Data area description and format on page 26.

Argument names and values

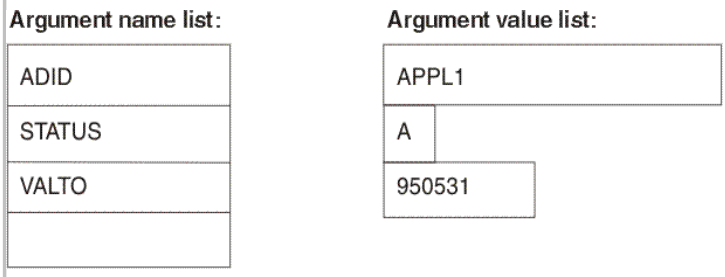
The fourth and fifth fullwords in the parameter list are the addresses of the argument name list and argument value address list.

The arguments provide specific information about your request. An argument can consist of an *argument name* alone, or an argument name and a matching *argument value*. Some requests require only one or more argument names, and some require argument names and values. If argument values are required, they are always associated one-for-one with the argument names.

Arguments can be compared to operands of a TSO command, where the argument name corresponds to the parameter keyword, and the argument value corresponds to the parameter value. For example:

Example

Figure 2. Program interface arguments in TSO command notation



Equivalent TSO command notation:

```
ADID(APPL1) STATUS(A) VALTO(950531)
```

The parameter list contains two addresses for the arguments, one pointing to the argument name list and one pointing to the argument value address list.

The argument name list is an array of 8-byte character fields. Each field contains an argument name, is left justified. Blanks must appear to the right of the argument name if it is shorter than 8 characters. The list is terminated by an all-blank field.

The argument value address list contains a list of addresses that point to the argument values. For a character argument value, the length of the field should be the same as that shown in the argument table. But, when a character argument is used as a selection argument, only the characters up to the first blank or comparison operator are used. Date and time data types are processed in the same way as character argument values. A numeric argument value must always be a fullword.

The retrieval of a record from the application description database is an example of how arguments are used. Here, the arguments identify the particular record required. The argument names identify the names of fields in the record, and the argument values identify the values of those fields for the record you want to retrieve (for details, see Figure 2: Program interface arguments in TSO command notation on page 25).

Sometimes, there might be a reason to specify the same argument more than once. For example, to get a list of active operations, you can specify argument name STATUS and C ≠ for the value, plus argument name STATUS and D ≠ for the value. You can specify an argument multiple times; up to 32 arguments can be defined in the argument name list.

Communication block

The sixth fullword in the parameter list is the address of a fullword containing the address of the communication block.

The first request to EQQYCOM must be an INIT request, which establishes a *communication session* between EQQYCOM and your program. During INIT request processing, EQQYCOM builds a *communication block* representing the session and returns its address in the fullword whose address is in the parameter list. The communication block address provided must remain unmodified during each subsequent call to EQQYCOM until the end of the session, so that EQQYCOM can identify the session that requests are coming from.

Return code

After each request, EQQYCOM provides a return code indicating if the request was successful or not.

The seventh fullword in the parameter list is the address of a fullword containing the return code. This return code is also placed in register 15.

Sequence of requests

About this task

Each communication session must always start with an INIT request and end with a TERM request. There can be several requests between them.

When modifying the current plan, requests must be made as follows:

1. With a series of requests, an *MCP block* is built containing all the necessary information required for one modification of the current plan.
2. With an EXECUTE request, information in the MCP block is used to actually update the current plan data set.

Also, when modifying the current plan, you can make a series of requests that refer to the same occurrence. The first request identifies the occurrence, and following requests can modify data related to that occurrence without needing to specifically identify it each time. The program interface remembers what the *current occurrence* is. Similarly, the program interface remembers the *current operation* and, once identified, a series of requests can be made that refer to it.

Other requests can be made in any sequence except where specifically noted. For example, you can produce a *list* of records with one request, which you can follow with one or more requests that *select* records from the list.

Data area description and format

About this task

Requests to EQQYCOM often involve either reading one or more records from an IBM Z Workload Scheduler database or data set, or writing them. In both cases, the record is placed in a data area and its address provided in a fullword whose address is in the parameter list. When you are retrieving information, EQQYCOM places the required record in a data area and provides the address of this area. When you are writing information to an IBM Z Workload Scheduler database or data set, your program must build its own data area and provide its address. Note that EQQYCOM might use the same piece of storage for data areas in successive data retrieval requests, overwriting the data area used for the previous request each time.

The data area consists of two parts:

- The header
- The data record

Header format

The header describes the segments in the record and their actual location within the record. The length and format of each segment type is fixed. For a description of the segments, see Program interface record format on page 370.

Note: For records retrieved with the SELECT request, the header always has a length that is a multiple of 32, with any unused header entries set to 00x. For records created for the INSERT and REPLACE requests, it is not necessary to set the header length to a multiple of 32, but if you do, you can use direct byte for byte comparison of input and output records.

The header consists of one or more header entries, each entry describing one segment in the data record. Each header entry is 16 bytes and consists of:

Segment name (8 characters)

A character field containing the name of a segment. If this field is blank, this is the last header entry in the header.

Offset to segment (1 fullword)

Offset to the start of this segment within the record from the start of the header. If this data area is from a LIST or SELECT request and it is the last header entry (segment name is blanks), this field contains more information about the request. This is further described under the detailed descriptions of the requests later in this chapter.

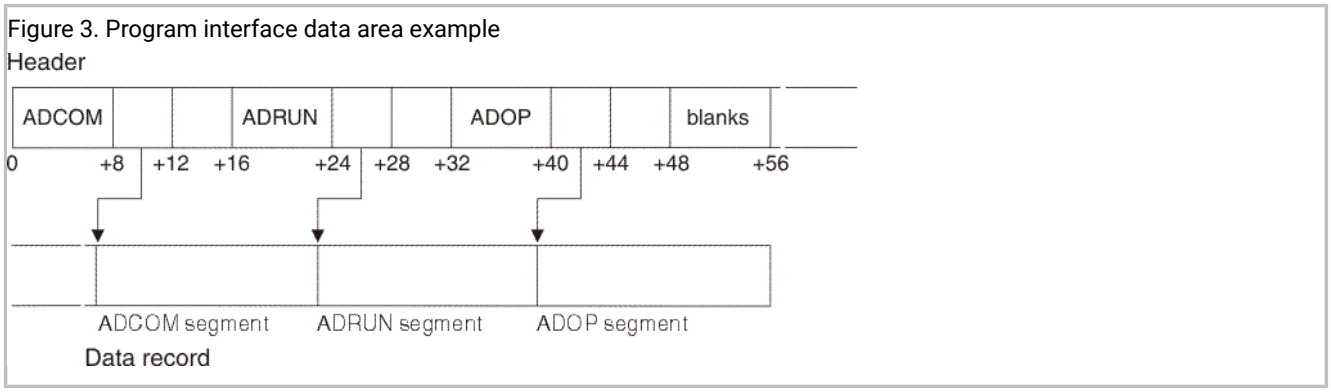
Reserved (4 bytes)

Reserved for use by IBM Z Workload Scheduler.

The header is terminated by a header entry with a blank segment name. Figure 3: Program interface data area example on page 27 shows an example of a data area using an application description.

Example

Figure 3. Program interface data area example



Data record format

Each data record handled by the program interface function consists of a *subset* of the complete IBM Z Workload Scheduler record. Each record consists of the same fields that are available in the ISPF dialogs, in the same format. Yes/No fields are single character fields, which contain either Y or N. Integer values are fullword fields.

The amount of information in an IBM Z Workload Scheduler record can vary enormously. For example, an application description can contain one run cycle and one operation, or it can contain many run cycles and many operations. The size of

each record and its format can vary greatly. Because of this, the program interface function uses a *header* for each record. The header contains information about the record.

Each record consists of one or more *segments* representing different information in that record. For example, an application description consisting of one run cycle and three operations is described by a record consisting of one run cycle segment and three operation segments. Also, one *common* segment always exists, which contains basic information, such as the application name, owner, and validity date. The common segment is always the first segment of the data record. Other segments can appear in any order except that segments that are logically related appear together. For example, in an application description record, the operation segments (ADOP) can appear in any order, but the dependency (ADDEP) and special resource segments (ADSR) always follow immediately after the ADOP to which they belong.

Date considerations

IBM Z Workload Scheduler can handle dates up to 31 December 2071. This high date is the default for application description valid-to and run cycle out-of-effect dates when you use the IBM Z Workload Scheduler dialogs.

Internal date representation

Internally, IBM Z Workload Scheduler works with a two-digit year format, so dates are represented as 00 to 99. In order to handle dates before and after 2000, IBM Z Workload Scheduler has chosen 72 as the base year. This means that, internally, 1972 is represented as 00, 1995 as 23, and 2071 as 99.

This internal date does not affect IBM Z Workload Scheduler dialog and report users. They always see the real date. However, PIF requests often involve reading or writing records in an IBM Z Workload Scheduler database. These records contain dates in the internal two-digit format with base year 72. You use the PIFCWB and PIFHD parameters of the INTFOPTS statement or the CWBASE and HIGHDATE parameters of the INIT statement to define how you want these dates to be presented to PIF applications.

PIFCWB and CWBASE values determine what base year IBM Z Workload Scheduler uses when presenting dates to PIF applications. If you specify 00, dates are presented as the last two digits of the real date. For example, 1995 is presented as 95 and 2001 as 01. Note, however, that the PIFCWB and CWBASE parameters affect all dates *except* the default out-of-effect and valid-to dates. These dates are presented to PIF application as the value specified in the PIFHD and HIGHDATE parameters.

For details about these statements, see *IBM Z Workload Scheduler: Customization and Tuning*.

Date arguments in PIF applications

You might have PIF applications developed before year 2000 that use 991231 as the value of the VALTO argument to indicate the default valid-to date of the last version of the AD. The real default date is 31 December 2071. However, by using the PIFHD parameter of the INTFOPTS statement or the HIGHDATE parameter of the INIT statement to define the high date as 991231, you can use these existing PIF applications without updating them.

A good way to avoid specifying a specific date for default valid-to dates is to define the PIFHD keyword of the INTFOPTS statement or the HIGHDATE keyword of the INIT statement as a six-character string. IBM Z Workload Scheduler will always interpret this as representing the default valid-to date.

Table 1: Comparison of Date Representations on page 29 gives an overview of the different date representations.

Table 1. Comparison of Date Representations

Real date	Internal date	PIF date (base 00, high date 711231)	PIF date (base 72, high date 991231)
1994/06/15	220615	940615	220615
2004/06/15	320615	040615	320615
2071/12/31	991231	711231	991231

Updating application description run cycles with PIF

When you use the ISPF dialogs to update or create application descriptions, you specify a run cycle out-of-effect date. Then IBM Z Workload Scheduler calculates the run cycle valid-to date by subtracting one day from the out-of-effect date. However, when you use PIF to update an AD you do not specify the out-of-effect date, you specify the valid-to date. Then IBM Z Workload Scheduler calculates the out-of-effect date by adding one day. If you specify the valid-to date as the default high date, adding one day would make the date higher than the highest allowed date. Therefore, when you specify the valid-to date in a PIF application as the default high date, IBM Z Workload Scheduler takes the IBM Z Workload Scheduler high date as the out-of-effect date.

Security considerations

You need authorization to use many of the program interface requests. If you do not have authority for the request you need, give the relevant access type and RACF® resource code to your IBM Z Workload Scheduler administrator. Table 2: Access Authority for Program Interface Requests on page 29 describes the access authority you need:

Table 2. Access Authority for Program Interface Requests

Program interface request	Access type required
INIT OPTIONS RESET TERM	None
LIST SELECT	Read
DELETE EXECUTE	Update

Table 2. Access Authority for Program Interface Requests (continued)

Program interface request	Access type required
INSERT MODIFY REPLACE SETSTAT	

You need authorization to access these IBM Z Workload Scheduler fixed resources:

Table 3. Program Interface Resources and the Corresponding IBM Z Workload Scheduler Fixed Resources Used for Checking Authorization

Program interface resource	IBM Z Workload Scheduler fixed resource
ADCOM, AD, ADEXT, ADKEY, ADRE	AD
AWSCL	WSCL
CL, CLCOM	CL
CPEXT, CPST, CPOC, CPOCCOM, CPOP, CPOPCOM, CPWS, CPWSV, CPWSCOM, CPWSVCOM, IVL, VIVL, MCPBLK	CP
CSR, CSRCOM, CPOPSRU	SR
ETT	ETT
JCLV, JCLVCOM	JV
JS, JSCOM, JCLPREP, JCLPREPA, JL, JLCOM	JS
LTOC, LTOCCOM	LT
OI, OICOM	OI
PR, PRCOM	PR
RG, RGCOM	RG
SR, SRCOM	RD
WS, WSCOM, WSV, WSVCOM	WS

For example, to list the intervals during which all workstations are closed, resource AWSCL, you need READ access to the WSCL fixed resource.

Running user-written programs compiled for older scheduler versions

Before you try to run a program compiled for a previous version of IBM Z Workload Scheduler, the program OBJ must be compiled, or at least link-edited, for the current IBM Z Workload Scheduler version.

Overview of request types

The requests that you can make to the program interface are summarized here. The requests are described in detail in the following sections and are arranged in alphabetical order.

DELETE

Deletes data items.

EXECUTE

Performs an actual update of the current plan.

INIT

Initializes the communication session between your program and the IBM Z Workload Scheduler subsystem.

INSERT

Inserts new data items or additional information into existing data items.

LIST

Retrieves a list of data items of a specified type using generic search arguments.

MODIFY

Modifies data fields in the LTP or current plan, or identifies CP or LTP data items for further modification.

OPTIONS

Specifies options to be used when performing PIF requests. You can use these options to automatically resolve external dependencies when adding LTP or CP occurrences, improve the time taken to retrieve information about operations, request the address of the area where the message ID is returned, and to prevent messages being written to the message log.

REPLACE

Replaces an existing application description or operator instruction.

RESET

Cancels a series of modify current plan requests if performed before the EXECUTE request.

SELECT

Retrieves a single data item in detail.

SETSTAT

Modifies the status of a condition dependency. You can use it to change the condition status from undecided to true or false, if the original status is undecided because of missing step-end information.

TERM

Terminates the communication session between your program and the IBM Z Workload Scheduler subsystem.

Table 4. Records Using a Common Segment

Arg names	Length
ADCOM	192
AWSCL	80
CLCOM	96
CPOCCOM	428
CPOPCOM (*)	380
CPOPSRU (*)	96
CSRCOM (*)	240
CPWSCOM (*)	128
CPWSVCOM(*)	129
ETT	128
JCLVCOM	96
JLCOM	64
JSCOM	96
LTOCCOM	157
OICOM	96
PRCOM	96
RGCOM	160
SRCOM	204
WSCOM	128
WSVCOM	128

(*): You cannot specify this argument name to delete the entire record.

DELETE request

The DELETE request deletes a record or record segment. If you delete a record the arguments identify the particular record to be deleted. If you want to delete only some information in an occurrence (for example, one of the operations in an occurrence), you must first use a MODIFY request to identify the occurrence before you use the DELETE request for the operation. Similarly, if you want to delete a special resource specification or a current plan condition for an operation, you must use a MODIFY request to identify the occurrence and then use a MODIFY request to identify the operation, before using a DELETE for the special resource.

To delete an interval of a current plan workstation you must precede the DELETE IVL with a MODIFY CPWS to identify the workstation.

To delete the extended name of an operation you must use the MODIFY request. For details, see MODIFY CPEXT on page 209.

The DELETE request can be used to modify information in the current plan. All requests that cause a modification of the current plan require a later EXECUTE request for the modification to actually take effect.

Action code

DELETE

Resource code

The resource code specifies which record type or record segment you want to delete. You can specify these values:

AD

Application description record

AWSCL

All workstations closed record

CL

Calendar record

CPCOND

Current plan condition

CPLAT

Operation user-defined late information

CPOC

Current plan occurrence record

CPOP

Current plan operation record

CPPRE

Current plan predecessor segment

CPSIMP

Current plan condition dependency

CPSR

Current plan special resource segment

CPSUC

Current plan successor segment

CPUSRF

Current plan user field segment

ETT

Event triggered tracking criteria record

IVL

Current plan workstation interval segment

JCLV

JCL variable table record

JL

JS file JOBLOG record

JS

Job control language record

LTOC

LTP occurrence record

LTCPRE

LTP conditional predecessor segment

LTPRE

LTP predecessor segment

OI

Operator instruction record.

PR

Period record

RG

Run cycle group record

SR

Special resource record

VIVL

Current plan virtual workstation destination interval segment

WS

Workstation description record

WSV

Virtual workstation destination record

Data area

Not used.

Arguments

The arguments identify the particular record you want to delete. Two ways you can do this are:

- Specify field names of the record as argument names and specify the addresses of field values, to identify the particular record you want to delete. The values can be:
  - Character values. A blank character terminates the field.
  - Numeric values, which must occupy a fullword.

  You must specify sufficient arguments to *uniquely* identify a record. You can use a comparison operator after the argument values. The default, *equals* (=), is assumed if you do not.
- Specify the record type as an argument name and the address of the previously retrieved common segment as the argument value address, if you have already retrieved the common segment of a record but you then want to delete the entire record. For a description of the record types that you can specify as argument names, see Table 4: Records Using a Common Segment on page 32.

Note: The values of PIF arguments as dates depend on the PIF base year, which is defined by the PIFCWB keyword on the INTFOPTS statement, or the CWBASE keyword of the INIT statement. The value of the VALTO argument for default high date depends on the PIFHD keyword of the INTFOPTS statement or the HIGHDATE keyword of the INIT statement. For details about these statements, see *Customization and Tuning*.

Delete AD arguments

Table 5. Delete AD Arguments

Arg names	Length	Data type	Description
ADID	16	Char	Application description ID
GROUP	8	Char	Authority group name
GROUPDEF	16	Char	Group definition ID
OWNER	16	Char	Owner ID
PRIORITY	4	Integer	Priority
STATUS	1	Char	Status: P=Pending A=Active
TYPE	1	Char	Application type: A=Application G=Group Default is A
VALFROM	6	Char YYMMDD	Valid-from date

Table 5. Delete AD Arguments (continued)

Arg names	Length	Data type	Description
VALTO	6	Char YYMMDD	Valid-to date

Note: IBM Z Workload Scheduler assumes application type A, if you do not specify the TYPE argument name.

Delete AWSCL arguments

Table 6. Delete AWSCL Arguments

Arg names	Length	Data type	Description
DATE	6	Char YYMMDD	Date

Delete CL arguments

Table 7. Delete CL Arguments

Arg names	Length	Data type	Description
CALENDAR	16	Char	Calendar ID

Delete CPCOND arguments

Note: Always identify an operation with a MODIFY CPOP request before a DELETE CPCOND request.

Table 8. Delete CPCOND Arguments

Arg names	Length	Data type	Description
CONDID	4	Integer	Condition ID. Valid values are from 1 to 999.

Delete CPLAT arguments

There are no arguments for the Delete CPLAT request.

Delete CPOC arguments

Table 9. Delete CPOC Arguments

Arg names	Length	Data type	Description
ADID	16	Char	Application description ID
IA	10	Char YYMMDDHHMM	Input arrival date and time

Delete CPOCPRE arguments

Table 10. Delete CPOCPRE Arguments

Arg names	Length	Data type	Description
PREADID	16	Char	Predecessor application ID
PREIA	10	Char YYMMDDHHMM	Predecessor input arrival date and time
*	1	Char	Reserved
PREOPNO	4	Integer	Predecessor operation number

Delete CPOCSUC arguments

Table 11. Delete CPOCSUC Arguments

Arg names	Length	Data type	Description
SUCADID	16	Char	Successor application ID
SUCIA	10	Char YYMMDDHHMM	Successor input arrival date and time
SUCOPNO	4	Integer	Successor operation number

Delete CPOP arguments

Table 12. Delete CPOP Arguments

Arg names	Length	Data type	Description
OPNO	4	Integer	Operation number

Delete CPPRE arguments

Table 13. Delete CPPRE Arguments

Arg names	Length	Data type	Description
PREADID	16	Char	Predecessor application ID
PREIA	10	Char YYMMDDHHMM	Predecessor input arrival date and time
PREMAND	1	Char	The predecessor is mandatory. The value can be Y or N (default). Specify Y if the predecessor is mandatory.
PREOPNO	4	Integer	Predecessor operation number

Note: When deleting an internal predecessor, only specify PREOPNO. Specify all arguments to delete an external mandatory predecessor. Omit PREMAND if the predecessor is not mandatory.

Delete CPSIMP arguments

Note: Always identify an occurrence, an operation and a condition with:

- An INSERT or MODIFY CPOC request
- An INSERT or MODIFY CPOP request
- An INSERT or MODIFY CPCOND request

before a DELETE CPSIMP request.

Table 14. Delete CPSIMP Arguments

Arg names	Length	Data type	Description
PREADID	16	Char	Predecessor application name
PREIA	10	Char	Predecessor application input arrival date and time
PREOPNO	4	Integer	Predecessor operation number
PROCSTEP	8	Char	Use it to define a step level dependency. If the step is not in a procedure, this parameter identifies the job step name, otherwise it identifies the step name in the JCL procedure. It must correspond to the name of an `EXEC PGM=` statement.
STEPNAME	8	Char	Use it in conjunction with PROCSTEP when defining a step level dependency, only if the step is in a procedure, to identify the name of a step that invokes an in-stream or cataloged procedure. It must correspond to the name of an `EXEC PROC=` statement.
TYPE	2	Char	Condition type: RC = To check the predecessor return code ST = To check the predecessor status
LOG	2	Char	Logical operator:

Table 14. Delete CPSIMP Arguments (continued)

Arg names	Length	Data type	Description
			GE = Greater than or equal to. Valid only for RC condition type. GT = Greater than. Valid only for RC condition type. LE = Less than or equal to. Valid only for RC condition type. LT = Less than. Valid only for RC condition type. EQ = Equal to. NE = Not equal to. Use it to specify conditions on final statuses only. RG = Range.
VALRC	4	Char	Return code value.
VALRC2	4	Char	Return code value, as second boundary in a range expressed by the RG logical operator.
VALST	1	Char	Status, valid only for ST type

Delete CPSR arguments

Table 15. Delete CPSR Arguments

Arg names	Length	Data type	Description
RESNAME	44	Char	Special resource name

Delete CPSUC arguments

Table 16. Delete CPSUC Arguments

Arg names	Length	Data type	Description
SUCADID	16	Char	Successor application ID
SUCIA	10	Char YYMMDDHHMM	Successor input arrival date and time
SUCOPNO	4	Integer	Successor operation number

Note: When deleting an internal successor, only specify SUCOPNO. All three arguments must be specified to delete an external successor.

Delete CPUSRF arguments

Table 17. Delete CPUSRF Arguments

Arg names	Length	Data type	Description
UFNAME	16	Char	User field name

Note: When deleting a user field, only specify UFNAME. The corresponding user field value is also deleted.

Delete ETT arguments

Table 18. Delete ETT Arguments

Arg names	Length	Data type	Description
ADID	16	Char	Associated application ID
ETTNAME	44	Char	Name of trigger
ETTTYPE	1	Char	Type of trigger, 2 -> job 3 -> special resource

Delete IVL arguments

An interval can have information originating from the workstation description, indicator CPIVLDP in segment CPIVL is set to Y, or else to N. If an interval is changed or created via the dialog or the program interface, the indicator CPIVLMOD in CPIVL is set to Y, or else to N.

DELETE IVL only affects modifications. Intervals with CPIVLDP=Y remain after a DELETE, the interval is reset to the daily planning values and CPIVLMOD is set to N. Intervals with CPIVLDP=N are fully deleted.

Table 19. Delete IVL Arguments

Arg names	Length	Data type	Description
FROM	10	Char YYMMDDHHMM	Interval start date and time

Delete JCLV arguments

Table 20. Delete JCLV Arguments

Arg names	Length	Data type	Description
JCLVTAB	16	Char	JCL variable table ID

Delete JL arguments

Table 21. Delete JL Arguments

Arg names	Length	Data type	Description
ADID	16	Char	Application ID
IA	10	Char YYMMDDHHMM	Input arrival date and time
JOBNAME	8	Char	z/OS® Job name
OPNO	4	Integer	Operation number
WSNAME	4	Char	Workstation name

Delete JS arguments

Table 22. Delete JS, JSCOM Arguments

Arg names	Length	Data type	Description
ADID	16	Char	Application ID
IA	10	Char YYMMDDHHMM	Input arrival date and time
JOBNAME	8	Char	z/OS® Job name
OPNO	4	Integer	Operation number
WSNAME	4	Char	Workstation name

Delete LTOC arguments

Table 23. Delete LTOC Arguments

Arg names	Length	Data type	Description
ADID	16	Char	Application description ID
IAD	6	Char YYMMDD	Input arrival date
IAT	4	Char HHMM	Input arrival time

Delete LTCPRE arguments

Table 24. Delete LTCPRE Arguments

Arg names	Length	Data type	Description
ADID	16	Char	Application description ID
IAD	6	Char YYMMDD	Input arrival date
IAT	4	Char HHMM	Input arrival time

Table 24. Delete LTCPRE Arguments (continued)

Arg names	Length	Data type	Description
PREADID	16	Char	Conditional predecessor application ID
PREIAD	6	Char YYMMDD	Conditional predecessor input arrival date
PREIAT	4	Char HHMM	Conditional predecessor input arrival time

Delete LTPRE arguments

Table 25. Delete LTPRE Arguments

Arg names	Length	Data type	Description
ADID	16	Char	Application description ID
IAD	6	Char YYMMDD	Input arrival date
IAT	4	Char HHMM	Input arrival time
PREADID	16	Char	Predecessor application ID
PREIAD	6	Char YYMMDD	Predecessor input arrival date
PREIAT	4	Char HHMM	Predecessor input arrival time

Note: DELETE LTPRE is used only to delete external predecessors. No support is provided in the long-term plan for internal dependencies.

Delete OI arguments

Table 26. Delete OI Arguments

Arg names	Length	Data type	Description
ADID	16	Char	Application ID
OPNO	4	Integer	Operation number

Note: To delete both the operator instruction and any associated temporary instructions, issue a LIST OICOM request followed by this loop:

1. A request with the OICOM segment as the argument
2. A SELECT OICOM with argument NEXT.

Continue the loop until SELECT OICOM NEXT gives a return code greater than 0.

Delete PR arguments

Table 27. Delete PR Arguments

Arg names	Length	Data type	Description
PERIOD	8	Char	Period name
PRTYPE	1	Char	Period type

Delete RG arguments

Table 28. Delete RG Arguments

Arg names	Length	Data type	Description
RGID	8	Char	Run cycle group ID
RGOWNER	16	Char	Run cycle group owner
RGCALEND	16	Char	Run cycle group calendar
RGVARTAB	16	Char	Run cycle group variable table
RUNNAME	8	Char	Run cycle name
RUNCAL	16	Char	Run cycle calendar
RUNVTAB	16	Char	Run cycle variable table
RUNSETID	8	Char	Run cycle subset ID

Delete SR arguments

Table 29. Delete SR Arguments

Arg names	Length	Data type	Description
RESGROUP	8	Char	Special resource group ID
RESHIPER	1	Char	DLF resource indicator
RESNAME	44	Char	Special resource name

Delete VIVL arguments

If an interval contains information originating from the Virtual Workstation Destination description, the indicator CPVIVLDP in segment CPVIVL is set to Y, otherwise it is set to N. If an interval is changed or created using the dialog or the program interface, the indicator CPVIVLMOD in segment CPVIVL is set to Y, otherwise it is set to N.

DELETE VIVL only affects modifications. Intervals with CPVIVLDP=Y remain after a DELETE, the interval is reset to the daily planning values and CPVIVLMOD is set to N. Intervals with CPVIVLDP=N are fully deleted.

Table 30. Delete VIVL Arguments

Arg names	Length	Data type	Description
FROM	10	Char YYMMDDHHMM	Interval start date and time

Delete WS arguments

Table 31. Delete WS Arguments

Arg names	Length	Data type	Description
WSNAME	4	Char	Workstation name
WSREP	1	Char	Workstation reporting attribute
WSRETYPE	1	Char	Remote engine type: D = distributed, Z = z/OS® or blank
WSTYPE	1	Char	Workstation type
WSWAIT	1	Char	WAIT workstation, Y or N

Delete WSV arguments

Table 32. Delete WSV Arguments

Arg names	Length	Data type	Description
WSNAME	4	Char	Virtual workstation name
WSDEST	8	Char	Virtual workstation destination

Communication block address

This is the address returned by INIT request processing, which must remain unmodified for all following requests.

Return code

When EQQYCOM returns control, this fullword shows the outcome of the request:

0

The request was successful.

4

The record; AD, AWSCL, CL, ETT, JCLV, JS, OI, PR, SR, WS, or WSV is currently being updated by another user. The record is not deleted.

8

The request was unsuccessful. An error message has been written to the message log data set.

EXECUTE request

The EXECUTE request causes an update of the current plan after one or more modify, insert, or delete current plan requests are completed.

If you are changing more than one current plan occurrence or current plan workstation before an EXECUTE request, you must complete all changes to one occurrence or workstation before changing another. If you do not complete all changes to one occurrence or workstation a message is issued and all modifications since the last EXECUTE request are reset.

For changes to current plan resources, CSR, no EXECUTE is required.

Action code

EXECUTE

Resource code

MCPBLK

Data area

Not used.

Arguments

Not used.

Communication block address

This is the address returned by INIT request processing, which should remain unmodified for all following requests.

Return code

When EQQYCOM returns control, this fullword shows the outcome of the request:

0

The request was successful.

8

The request was unsuccessful. An error message has been written to the message log data set.

INIT request

The INIT request identifies the IBM Z Workload Scheduler subsystem required and initializes the communication session between this subsystem and your program. It must always be the first request. The INIT request builds a communication block. EQQYCOM returns its address to your program.

Through the INIT statement in the parameter file EQQYPARM, the user might override the parameters specified in the INIT request.

The parameter file can be a sequential file, or a PDS allocated as:

```
//EQQYPARM DD DISP=SHR,DSN=OPCESA.SYS1.CNTL(YPARM)
```

Action code

INIT

Resource code

The name of an active IBM Z Workload Scheduler subsystem to which all following requests are directed.

Data area

Not used.

Arguments

You can specify arguments to:

- Determine if a recovery environment is established. The recovery environment consists of a SPIE exit routine and an ESTAE recovery routine, which, in case of error, dumps certain storage areas and terminates execution. You can specify argument name ESTAEI, ESTAER, LUNAME, or NOESTAE. Argument values are not required.
- Identify the message log to which that messages are written.

Argument name=ACCOID

The parameter that determines if the OI database is to be accessed when a LIST or SELECT request on CP operations is issued.

Argument value=*accoid*

A 1-byte character field for the accoid: valid values are Y or N. Y means that the OI database is read (this is the default). N means that the OI database is not read.

Argument name=DUBPROC

The parameter with which the BPX1SDD routine is invoked for the program interface TCP/IP session.

Argument value=*dubproc*

A 1-byte character field for the dubproc: valid values are Y or N. Y means that BPX1SDD is invoked by using DUBPROCESS parameter. N means that BPX1SDD is invoked by using DUBPROCESSDEFER parameter. The default is N.

Argument name=ESTAEI

The recovery environment is established at the INIT request. It remains in effect until the TERM request. This is the default.

Argument name=ESTAER

The recovery environment is established and terminated for each individual request. This might be needed if, for example, your program has a recovery environment dependent on the setting of a certain register, as in PLI.

Argument name=LUNAME

This argument allows the user to specify a server or controller LU name for the program interface session.

Argument value=*luname*

A 17-byte field for the LU name address, ending by a blank if shorter than 17 bytes.

Argument name=MLOGDDN

This argument identifies a message log that messages are written to, rather than the default message log, EQQMLOG.

Each INIT request requires its own message log. If you make more than one INIT request before a TERM request, or if PIF is invoked by a program or started task that is already using EQQMLOG, specify MLOGDDN for each additional INIT request. If MLOGDDN is not specified, and EQQMLOG is already in use, message EQQZ038E is written to the SYSLOG and the INIT request fails.

Argument value=*ddname*

An 8-character field, left justified, which identifies the ddname of the data set that messages are written to.

Argument name=NOESTAE

No recovery environment is established.

Argument name=PIFDLCHECK

For occurrences that are dynamically added to the current plan through OCL, PIF, or BCIT without setting a deadline, PIFDLCHECK enables IBM Z Workload Scheduler to flexibly set a deadline.

Argument value=Y

When the occurrence deadline to be taken from the first run cycle is not available or is earlier than the Input Arrival time, IBM Z Workload Scheduler sets the deadline as the IA time + 8 hours.

Argument value=N

Default. IBM Z Workload Scheduler sets the occurrence deadline as the IA time plus 8 hours *only* if the deadline to be taken from the first run cycle is not available. If it is available and is earlier than the Input Arrival time, an error condition occurs.

Argument name=REMHOST

The server host name for the program interface TCP/IP session. REMHOST and LUNAME are mutually exclusive.

Argument value=*server hostname*

A 52-byte field for the host name address, ending by a blank if shorter than 52 bytes.

Argument name=REMPORT

The server port number for the program interface TCP/IP session. REMHOST and LUNAME are mutually exclusive.

Argument value=*server port number*

A 4-byte integer field for the port number: valid values are from 0 to 65535.

Argument name=USRLEV

This argument communicates to EQQYCOM the level of the user program. If not specified, the programming interface assumes that the user-written program is invoking the PIF program at its latest version, and you need to recompile to see the changes in the segment layouts. In this situation, PIF uses new layouts to communicate with old user program.

According to the functions that you are using (earlier or later than IBM Z Workload Scheduler V9.5), specify one of the following values:

Argument value=*n*

A 4-byte integer field that identifies the level of the user program. The valid values are:

11

Identifies IBM Z Workload Scheduler V9.5. If you are using this level, you need to recompile the user-written applications to see the changes in the segment layouts.

12

Identifies the enablement of changes done for IBM Z Workload Scheduler V9.5 through the APAR PH12689.

13

Identifies the enablement of changes done for IBM Z Workload Scheduler V9.5 through the APAR PH22448.

14

Identifies the enablement of changes done for IBM Z Workload Scheduler V9.5 through the APAR PH30466.

17

Identifies the enablement of changes done for IBM Z Workload Scheduler V10.2 through the APAR PH68409.

Communication block address

When EQQYCOM returns control to your program, this contains the address of the communication block representing this program interface session. Ensure that this address remains unmodified during all following calls to EQQYCOM. The initial value of this field is not important, because it will be overlaid with the communication block address by EQQYCOM.

Return code

When EQQYCOM returns control, this fullword indicates the outcome of the request:

0

The request was successful. A program interface session has been successfully started. The address of the communication block has been placed in the parameter list.

8

The request was unsuccessful. Check the message log, SYSLOG, and EQQDUMP data sets for error information.

INSERT request

The INSERT request writes a new record or record segment to an IBM Z Workload Scheduler database or data set. This can be done in several ways:

- To insert new application descriptions, operator instructions, JCL records, new all workstations closed, calendar, ETT, period, special resource or workstation record, your program must provide the complete record to be inserted in the data area. Arguments are not used.
- To insert a new application occurrence into the current plan, you can:
  - Provide a complete *application description* record in the data area. This is then converted by IBM Z Workload Scheduler into a current plan occurrence. Here, the arguments can be used to provide the input arrival and deadline date and time.

    Or

  - Select an existing application description from the database to be added as an occurrence into the current plan. Here, the arguments are used to identify the existing application description from which the occurrence will be created. The arguments can also specify occurrence-related information such as input arrival time and deadline time. The data area is not used.

  When inserting an application occurrence into the long-term plan, the name of the application description must be supplied through the argument parameters. You cannot supply an application description through the data area. The data area pointer address must be set to zero before your program call.

When inserting a new occurrence using either of the previous methods, the input arrival date and time and deadline date and time can be provided in the arguments. If the input arrival is not provided when inserting a current plan occurrence, the current date and time is used (that is, the date and time at which the occurrence is inserted). However, if an occurrence already exists with this application ID and input arrival date and time, the next available minute in which no occurrence of this application exists will be used. You must supply an input arrival date and time if you are inserting an occurrence in the LTP.

If arguments are not provided for the deadline, these defaults are observed by IBM Z Workload Scheduler:

- If the occurrence is being added to the current plan and the input arrival is provided, the deadline from the first run cycle is used if a run cycle exists. If there are no run cycles or the input arrival is not provided, the deadline is set to the input arrival time plus 8 hours.
- When the occurrence is being added to the long-term plan, the deadline is set to the input arrival plus 8 hours.

By default, external dependencies of the occurrence are not resolved when it is added to the LTP or current plan. If resolution of external dependencies is required, the OPTIONS request must be used to specify this.

- To insert the extended name of an operation you must use the MODIFY request. For details, see MODIFY CPEXT on page 209.
- To insert new information into an existing LTP or current plan occurrence, you use the arguments to provide all the necessary information. For example, you can insert a new operation into an existing current plan occurrence. But the actual occurrence to which the information is to be added must have been identified by a previous MODIFY or INSERT request. Similarly, you can insert new information for an existing current plan operation, provided that the operation has been identified. This means you must first use a MODIFY request to identify the occurrence and then use a MODIFY request to identify the operation, before inserting a predecessor (CPPRE), successor (CPSUC), or special resource (CPSR).

  When identified, the program interface maintains a *current occurrence* and *current operation*.

  If you want to insert a new interval into a current plan workstation you must first identify the workstation with a MODIFY CPWS request.

  The arguments are used to specify all required information. The data area is not used.

The INSERT request can be used to modify information in the current plan. All requests that cause a modification of the current plan require a later EXECUTE request for the modification to take effect.

Action code

INSERT

Resource code

The resource code specifies which record type or record segment you want to insert. You can specify these values:

AD

Application description record

AWSCL

All workstations closed record

CL

Calendar record

CPCOND

Current plan condition

CPLAT

Operation user-defined late information

CPOC

Current plan occurrence record

CPOP

Current plan operation record

CPPRE

Current plan predecessor segment

CPSAI

System automation information for the current plan operation

CPSIMP

Current plan condition dependency

CPSR

Current plan special resource segment

CPSUC

Current plan successor segment

CPUSRF

Current plan user field segment

ETT

Event triggered tracking criteria record

IVL

Current plan workstation interval

JCLPREP

Promptable setup variables for the current operation

JCLV

JCL variable table record

JS

Job control language record

LTOC

LTP occurrence record

LTPRE

LTP predecessor segment

OI

Operator instruction record

PR

Period record

RG

Run cycle group record

SR

Special resource record

VIVL

Current plan virtual workstation destination interval segment

WS

Workstation record

WSV

Virtual workstation destination record

Data area

The data area is used in these situations:

- If you are inserting new application descriptions, operator instructions, JCL records, new all workstations closed, calendar, ETT, period, special resource, or workstation records, provide the complete record in the data area.
- If you are inserting new current-plan occurrences, specify the application ID as an argument and specify that no data area is available. If it is necessary to supply the application description via the data area, omit the application ID argument and give the application description via the data area.

Arguments

The arguments are used in these situations:

- If you are inserting a new current-plan occurrence of an existing application description, use the arguments to identify the application rather than having to provide the complete record yourself. The arguments tell IBM Z Workload Scheduler which application is required, and it handles the insertion of this application as an occurrence record in the LTP or CP. The arguments can also provide additional information, such as input arrival time, deadline, and priority. If you use the arguments, set the data area pointer address to zero before you issue your call.

  If you are inserting a new LTP or current plan occurrence, use the arguments to identify the application.
- If you are inserting a new current-plan occurrence and providing the application description information in the data area, the arguments can specify occurrence information, such as input arrival time, deadline, and priority. These arguments override the values in the application description.
- If you are inserting information for an existing LTP or current plan occurrence or operation, use the arguments to provide all the information to be inserted.

Note:

1. No arguments can be provided for AD, OI, JS, AWSCL, CL, ETT, PR, SR, WS, and WSV resource codes, because the complete record must be in the data area.
2. When inserting calendar records (CL) the standard day segment (that is, ='STANDARD') must appear as the second segment in the input field right after the CLCOM segment. Its corresponding interval segment must be immediately after.
3. When inserting special resource (SR), DAY=8 represents the *STANDARD* day.
4. The format of the duration used in the data area in Insert AD/WS will be defined by the DURSEC option, described in the paragraph OPTIONS request on page 92.

Note: The values of PIF arguments as dates depend on the PIF base year, which is defined by the PIFCWB keyword on the INTFOPTS statement, or the CWBASE keyword of the INIT statement. The value of the VALTO argument for default high date depends on the PIFHD keyword of the INTFOPTS statement or the HIGHDATE keyword of the INIT statement. For details about these statements, see Customization and Tuning.

Note: If the argument DURATION is used with the argument EDUR, an error message occurs.

Insert CPLAT

Table 33. Insert CPLAT Arguments

Arg names	Length	Data type	Description
LATACT	1	Char	The action taken if the operation has not yet started when the specified day and time is reached:

Table 33. Insert CPLAT Arguments (continued)

Arg names	Length	Data type	Description
			A = Only an alert message is issued. C = The operation is set to Complete, if its status allows it. Otherwise, it is NOPed. E = The operation is set to Error with OLAT, if its status allows it. Otherwise, this setting is postponed at the time when the status allows it. N = The operation and all its internal successors are NOPed, if their status allows NOPing. Otherwise, it is ignored.
LATACTDT	10	Char YYMMDDHHMM	Date and time by which the operation must start. If not, an action is issued.
LATALEDT	10	Char YYMMDDHHMM	Date and time by which the operation must start. If not, an alert is taken.

Note:

1. Always identify an operation with an INSERT or MODIFY CPOP request before an INSERT CPLAT request.
2. To modify a value already set in LATACTDT or LATALEDT, you must re-issue an INSERT CPLAT request with the desired values.

Insert CPOC arguments

When you are inserting a current plan occurrence, the ADID argument is required unless you are providing the entire application description in the data area. The ADID argument identifies an existing application description, an occurrence of which is to be inserted into the current plan. All remaining arguments are optional and provide more information about the occurrence.

Table 34. Insert CPOC Arguments

Arg names	Length	Data type	Description
ADID	16	Char	Application description ID
DEADLINE	10	Char YYMMDDHHMM	Deadline date and time
DESC	24	Char	Descriptive text
ERRCODE	4	Char	Error code
GROUP	8	Char	Authority group
GROUPDEF	16	Char	Group definition ID

Table 34. Insert CPOC Arguments (continued)

Arg names	Length	Data type	Description
IA	10	Char YYMMDDHHMM	Input arrival date and time
JCLVTAB	16	Char	JCL variable table
ODESC	24	Char	Descriptive text of owner
OWNER	16	Char	Owner ID
PRIORITY	4	Integer	Priority

Notes:

1. A DEADLINE argument is accepted also when no IA argument is specified. If the IBM Workload Scheduler selected IA is later than the DEADLINE argument value, the argument value is ignored. The default, IA plus 8 hours, is used instead.
2. If you specify 24.00 as the IA time, it is converted to 00.00 of the following day. In fact, the valid input arrival times are 00.00 through 23.59.
3. If you specify as deadline 00.00, it is converted to 24.00 of the previous day. In fact, the valid deadline times are 00.01 through 24.00.

Insert CPOCPRE arguments

Table 35. Insert CPOCPRE Arguments

Arg names	Length	Data type	Description
PREADID	16	Char	Application ID
PREIA	10	Char YYMMDDHHMM	Input arrival date and time
PREOPNO	4	Integer	Operation number
TRPTTIME	4	Integer HHMM	Tansport time

Insert CPOCSUC arguments

Table 36. Insert CPOCSUC Arguments

Arg names	Length	Data type	Description
SUCADID	16	Char	Application ID
SUCIA	10	Char YYMMDDHHMM	Input arrival date and time
SUCOPNO	4	Integer	Operation number

Insert CPCOND arguments

Note: Always identify an operation with an INSERT or MODIFY CPOP request before an INSERT CPCOND request.

Table 37. Insert CPCOND Arguments

Arg names	Length	Data type	Description
CONDID	4	Integer	Condition ID. Valid values are from 1 to 999.
COUNT	4	Integer	Condition counter. Use it to define the rule type: 0 = All the condition dependencies, in the corresponding INSERT CPSIMP list, must be true *n>0* = At least *n* out of the specified condition dependencies must be true The default is 0.
DESC	16	Char	Descriptive text

Insert CPOP arguments

Table 38. Insert CPOP Arguments

Arg names	Length	Data type	Description
AEC	1	Char	Automatic error completion
AJR	1	Char	Automatic job hold/release
ASUB	1	Char	Automatic job submission
CLATE	1	Char	Cancel if late
CLNTYPE	1	Char	Data Set cleanup type
CONDRJOB	1	Char	Conditional recovery job
DEADWTO	1	Char	Issue deadline WTO
DESC	24	Char	Descriptive text
DURATION	4	Integer	Estimated duration in 100th of a second
EDUR	4	Char HHMM	Estimated duration
EXPJCL	1	Char	Expanded JCL option
FORM	8	Char	Form number or blanks
HRC	4	Integer	Highest successful return code

Table 38. Insert CPOP Arguments (continued)

Arg names	Length	Data type	Description
JCLASS	1	Char	Job class
JOBCRT	1	Char	Critical job. P=Critical path target W=Eligible for WLM assistance N=Not eligible for WLM assistance
JOBNAME	8	Char	Job name
JOBPOL	1	Char	Workload monitor late job policy. ‘ ’ (blank) = default L = Long duration D = Deadline S = Latest start time C = Conditional mode
MONITOR	1	Char	Y=Operation monitored by an external product N=Operation not monitored by an external product
OPDL	10	Char YYMMDDHHMM	Operation deadline date and time or blank
OPDLACT	1	Char	The action taken if the operation does not complete at its deadline: ‘ ’ (blank) = Default. No action is taken. A = Only an alert message is issued. C = The operation is set to Complete, if its status allows it. Otherwise, it is NOPed. E = The operation is set to Error with ODEA, if its status allows it. Otherwise, this setting is postponed at the time when the status allows it. N = The operation and all its internal successors are NOPed, if their status allows NOPing. Otherwise, it is ignored.
OPIA	10	Char YYMMDDHHMM	Operation input arrival date and time or blank
OPNO	4	Integer	Operation number
PSUSE	4	Integer	Parallel servers required

Table 38. Insert CPOP Arguments (continued)

Arg names	Length	Data type	Description
R1USE	4	Integer	Resource 1 required
R2USE	4	Integer	Resource 2 required
RERUT	1	Char	Reroutable operation
RESTA	1	Char	Restartable operation
STATUS	1	Char	Operation status
TIMEDEP	1	Char	Time-dependent job
USERDATA	16	Char	Information stored in operation user data
USRSYS	1	Char	User sysout support
WSNAME	4	Char	Workstation name
WLMSCLS	8	Char	WLM service class

Insert CPPRE arguments

Table 39. Insert CPPRE Arguments

Arg names	Length	Data type	Description
PREADID	16	Char	Application ID
PREIA	10	Char YYMMDDHHMM	Input arrival date and time
PREOPNO	4	Integer	Operation number
TRPTTIME	4	Integer HHMM	Transport time

Note: When CPPRE is needed to insert an internal dependency, only PREOPNO and TRPTTIME arguments are valid.

Insert CPSAI arguments

Table 40. Insert CPSAI Arguments

Arg names	Length	Data type	Description
AUTFUNC	8	Char	System Automation automated function (for operation). It must be an alphanumeric value, uppercase format. The first character cannot be numeric.
COMMETXT	255	Char	System Automation command text. It must be set and cannot be blank.

Table 40. Insert CPSAI Arguments (continued)

Arg names	Length	Data type	Description
COMPINFO	64	Integer	System Automation completion information
SECELEM	8	Char	System Automation security element

Note:

1. The occurrence and operation to which the system automation information refers are identified, respectively, by the INSERT CPOC and INSERT CPOP sequences
2. You can use the insert CPSAI only if the operation runs on an automation workstation.

Insert CPSIMP arguments

Note: Always identify an occurrence, an operation and a condition with:

- An INSERT or MODIFY CPOC request
- An INSERT or MODIFY CPOP request
- An INSERT or MODIFY CPCOND request

before an INSERT CPSIMP request.

Table 41. Insert CPSIMP Arguments

Arg names	Length	Data type	Description
PREADID	16	Char	Predecessor application name
PREIA	10	Char	Predecessor application input arrival date and time
PREOPNO	4	Integer	Predecessor operation number
PROCSTEP	8	Char	Use it to define a step level dependency. If the step is not in a procedure, this parameter identifies the job step name, otherwise it identifies the step name in the JCL procedure. It must correspond to a step specifying the `EXEC PGM=` statement.
STEPNAME	8	Char	Use it in conjunction with PROCSTEP when defining a step level dependency, only if the step is in a procedure, to identify the procedure invocation step name.

Table 41. Insert CPSIMP Arguments (continued)

Arg names	Length	Data type	Description
TYPE	2	Char	Condition type: RC = To check the predecessor return code ST = To check the predecessor status
LOG	2	Char	Logical operator: GE = Greater than or equal to. Valid only for RC condition type. GT = Greater than. Valid only for RC condition type. LE = Less than or equal to. Valid only for RC condition type. LT = Less than. Valid only for RC condition type. EQ = Equal to. NE = Not equal to. Use it to specify conditions on final statuses only. RG = Range.
VALRC	4	Char	Return code value.
VALRC2	4	Char	Return code value, as second boundary in a range expressed by the RG logical operator.
VALST	1	Char	Status, valid only for ST type

Note: To create an internal dependency, do not specify either PREADID or PREIA.

Insert CPSR arguments

Table 42. Insert CPSR Arguments

Arg names	Length	Data type	Description
ONCOMPL	1	Char	Action on complete Y|N|R
ONERROR	1	Char	Keep on error Y|N
QUANTITY	4	Integer	Quantity required. Specify 0 to allocate the total quantity of the special resource. The value 0 is the same as blank in the dialogs.
RESNAME	44	Char	Special resource name

Table 42. Insert CPSR Arguments (continued)

Arg names	Length	Data type	Description
RESUSAGE	1	Char	Special resource usage S|X

Insert CPSUC arguments

Table 43. Insert CPSUC Arguments

Arg names	Length	Data type	Description
SUCADID	16	Char	Application ID
SUCIA	10	Char YYMMDDHHMM	Input arrival date and time
SUCOPNO	4	Integer	Operation number

Note: When CPSUC is needed to insert an internal dependency, only the SUCOPNO argument is valid.

Insert CPUSRF arguments

Note: Always identify an operation with an INSERT or MODIFY CPOP request before an INSERT CPUSRF request.

Table 44. Insert CPUSRF Arguments

Arg names	Length	Data type	Description
UFNAME	16	Char	User field name
UFVALUE	54	Char	User field value

Insert IVL arguments

An interval can have information originating from the workstation description, indicator CPIVLDP in segment CPIVL is set to Y, or otherwise to N. If an interval is changed via the dialog or the program interface then the indicator CPIVLMOD is Y, or otherwise N

INSERT IVL can insert an interval spanning existing intervals with CPIVLMOD=N. The inserted interval will be converted to several intervals as required by daily planning. Other requests following the INSERT must take this possible split into account; each request is handled fully before the next request.

Table 45. Insert IVL Arguments

Arg names	Len	Data type	Description
FROM	10	Char YYMMDDHHMM	Interval start date/time
PSCAP	4	Integer	Parallel server capacity

Table 45. Insert IVL Arguments (continued)

Arg names	Len	Data type	Description
R1CAP	4	Integer	Resource 1 capacity
R2CAP	4	Integer	Resource 2 capacity
TO	10	Char YYMMDDHHMM	Interval end date and time

Insert JCLPREP arguments

Table 46. Insert JCLPREP Arguments

Arg names	Length	Data type	Description
ADID	16	Char	Application ID
IA	10	Char YYMMDDHHMM	Input arrival date and time
OPNO	4	Integer	Operation number

Note: For a description about how to perform a JCL preparation using the program interface, see JCL preparation using PIF on page 119.

Insert JCLV arguments

Table 47. Insert JCLV Arguments

Arg names	Length	Data type	Description
JCLVTAB	16	Char	JCL variable table ID

Insert LTOC arguments

Table 48. Insert LTOC Arguments

Arg names	Length	Data type	Description
ADID	16	Char	Application ID
DEADLINE	10	Char YYMMDDHHMM	Deadline date and time
ERRCODE	4	Char	Error code
GROUPDEF	16	Char	Group definition ID
IAD	6	Char YYMMDD	Run date
IAT	4	Char HHMM	Input arrival time
JCLVTAB	16	Char	JCL variable table

Table 48. Insert LTOC Arguments (continued)

Arg names	Length	Data type	Description
PRIORITY	4	Integer	Priority

Insert LTPRE arguments

Table 49. Insert LTPRE Arguments

Arg names	Length	Data type	Description
ADID	16	Char	Application ID
IAD	6	Char YYMMDD	Run date
IAT	4	Char HHMM	Input arrival time
PREADID	16	Char	Application ID
PREIAD	6	Char YYMMDD	Run date
PREIAT	4	Char HHMM	Input arrival time

Note: INSERT LTPRE is used only to insert external predecessors. No support is provided in the long-term plan for internal dependencies.

Insert VIVL arguments

If an interval contains information originating from the workstation description, the indicator CPVIVLDP in segment CPVIVL is set to Y, otherwise it is set to N. If an interval is changed using the dialog or the program interface then the indicator CPVIVLMOD in segment CPVIVL is set to Y, otherwise it is set to N.

INSERT VIVL can insert an interval spanning existing intervals with CPVIVLMOD=N. The inserted interval will be converted to several intervals as required by daily planning. Other requests following the INSERT must take this possible split into account; each request is completed before the next request.

Table 50. Insert VIVL Arguments

Arg names	Len	Data type	Description
FROM	10	Char YYMMDDHHMM	Interval start date and time
PSCAP	4	Integer	Parallel server capacity
R1CAP	4	Integer	Resource 1 capacity
R2CAP	4	Integer	Resource 2 capacity
TO	10	Char YYMMDDHHMM	Interval end date and time

Communication block address

About this task

This is the address returned by INIT request processing, which should remain unmodified for all following requests.

Return code

When EQQYCOM returns control, this fullword shows the outcome of the request:

0

The request was successful.

4

One or more of the dependencies, specified by the application description of the INSERT LTOC request, was not set up because no applicable predecessor occurrence exists. This return code could also result from an INSERT request for any of LTPRE, CPOP, CPOC, CPPRE, and CPSR, if the dependency was not set up.

8

The request was unsuccessful. An error message has been written to the message log data set.

LIST request

The LIST request retrieves a list of records from the specified database or data set. The first entry in the list is made available for processing. Other records in the list can be retrieved using the SELECT request.

When you use the LIST request, the resulting list consists only of the common segments of the records. For a description of the data fields that make up the common segment of each record, see Program interface record format on page 370. To retrieve a complete record, you must use the SELECT request.

After a successful LIST request for a particular resource code, the list remains available until you build a new list for the same resource code, or until a TERM request. This means that you can have several active lists if required, but only one at a time for each resource code.

When retrieving current plan occurrences and operations, the default is to retrieve all matching objects except those in deleted status. When STATUS is provided as an argument, the specified selection overrides the default processing.

In case of large amount of data, the use of queries without filter argument might exceed any available storage and needs to be limited. For program interface applications, invoked by a clist and IKJEFT01, a test allocation is done finding out how much storage is available (between a minimum of 32KB and a maximum of 64MB) and thereafter a fraction (1/4) of it is allocated to receive the unknown amount of data from the IBM Workload Scheduler subsystem.

Action code

LIST

Resource code

The resource code identifies the record type the list will comprise. You can specify these values:

ADCOM

Application description common segment

ADKEY

Application description key segment

AWSCL

All workstations closed

CLCOM

Calendar common segment

CPEXT

Current plan operation extended name segment

CPCONDCO

Current plan condition common segment

CPOC, CPOCCOM

Current plan occurrence

CPOPCOM

Current plan operation common segment

CPOPSRU

Current plan operation segment with information about the operation in relation to a special resource

CPWSCOM

Current plan workstation common segment

CPWSVCOM

Current plan virtual workstation common segment

CRITSUCS

Current plan critical successors segment

CSRCOM

Current plan special resource common segment

ETT

Event triggered tracking criteria

GENDAYS

Run dates generated by run cycle rule segment

JCLVCOM

JCL variable table common segment

JLCOM

Job log common segment

JSCOM

JCL common segment

LTOCCOM

LTP occurrence common segment

OICOM

Operator instruction common segment

PRCOM

Period common segment

RGCOM

Run cycle group common segment

RGKEY

Run cycle group key segment

SRCOM

Special resource common segment

WSCOM

Workstation description common segment

WSVCOM

Virtual workstation destination common segment

Data area

When EQQYCOM returns control to your program after a successful LIST request, this fullword contains the address of a data area containing the first record from the requested list. Only the common segment of the requested record is provided when you use the LIST request. Appendix A. Program Interface Record Format describes the fields in each record common segment.

The header section for this record contains, besides the normal header information, a field containing a count of the number of elements in the list. This count field is in the final header entry, that is, the entry that has a blank segment-name field. The count is stored in the field that normally contains the segment offset. For a complete description of headers, see Header format on page 26.

Note: The resource code JSCOM retrieves JCL records from the JCL repository data set and not from a JCL library. But a SELECT request tries to get JCL records from a JCL library if they are not found in the JCL repository data set.

Arguments

Argument names specify field names of the record to be tested to determine if the record should be included in the list.

For each argument name specified, a corresponding argument value must be specified. The argument value you specify is compared with values in the actual database records to determine if the record should be included in the list. Argument values can be:

- Character values. Any number of characters terminated by a blank or comparison operator. Character values can be specified generically, using asterisks and percent signs as masking characters. An asterisk (*) can be used in place of any number of characters or a null string. A percent sign (%) can be used in place of exactly one character.

  Note:

  1. Because the first blank or comparison-operator symbol ends the argument value, you cannot search for fields that contain imbedded blanks or comparison-operator symbols.
  2. Generic search arguments, * and %, cannot be used in the year part (YY) of date arguments.

- Numeric values, which must occupy a fullword.

A comparison operator can follow the argument value, either with or without an intervening blank. The record is included in the list if:

=

The argument value is equal to the record value.

≠

The argument value is not equal to the record value.

>

The argument value is greater than the record value.

>=

The argument value is greater than or equal to the record value.

<

The argument value is less than the record value.

<=

The argument value is less than or equal to the record value.

If no comparison operator is supplied, equals (=) is assumed.

Note:

1. When you want to use a comparison operator (such as <, >, or ≠) in an argument, and the argument contains an IA value that includes a date and time, supply the full value as the argument value. The comparison operator can follow this value.
2. To prevent unpredictable results when the system assigns an area that was just freed from a previous request, remember to do the following:
    a. GETMAIN an area size of one additional byte to the length of the specific argument's request.
    b. Insert a blank character at the end of the argument value.

    To clarify what unpredictable results could take place, consider the following sequence in a PIF request:
    a. GETMAIN 27 bytes (to store ADID, IA, and the > (greater than) operator
    b. LIST request
    c. FREEMAIN
    d. GETMAIN 26 bytes (to store ADID,IA)
    e. SELECT request

    As shown in this sequence, if the GETMAIN assigned to the SELECT request is the same as the one of the LIST request, the > operator is still present in the SELECT storage and this can originate unwanted results.

For example, if the current plan contains such occurrences as:

```
AAAAAA                98/01/21 08.00  C
AAAAAA                98/01/22 07.00  C
BBBBBB                98/01/22 08.00  C
BBBBBB                98/01/23 08.00  C
```

and you want to list all occurrences whose IA value is greater than the first IA value, you must supply '9801210800<' as the argument value. Alternatively, if you want to list all the occurrences whose application name is greater than the first application name, for example, you can supply a string of any number of characters terminated by a comparison operator, such as 'AAA<='. You can thus use the comparison operators in different ways, depending on the type of data you use as the argument.

The comparison operators do not work with the generic search arguments.

Argument name: MATCHTYP

This argument can have the following values:

- EXA
- PFX
- SFX

With argument MATCHTYP specified, characters * and % are treated as normal characters instead of as generic matching characters, and blank as a normal character instead of ending the selection value. MATCHTYP EXA, PFX, and SFX affect:

- The STATUS and UFVALUE arguments of the CPOPCOM segment
- The ETTNAME argument of the ETT segment
- The RESNAME argument of the SRCOM and CSRCOM segments

If the MATCHTYP argument is specified, characters *, %, blank and comparison operators in a STATUS/ETTNAME/RESNAME argument, values are treated as normal characters.

When MATCHTYP is specified together with RESNAME or ETTNAME, the selection value must be padded with blanks up to the full resource name length of 44 characters. When RESNAME or ETTNAME are specified without MATCHTYP then the selection value is treated in the same way as any other selection value: it will be truncated at the first blank.

Note:

1. If MATCHTYP has the EXA value specified, then a record is selected only if the value in the record is exactly the same as the argument value.
2. If MATCHTYP has the PFX value specified, then a record is selected only if the start value in the record is the same as the argument value.
3. If MATCHTYP has the SFX value specified, then a record is selected only if the end value in the record is the same as the argument value.

The argument names and values that you can use to select records with a LIST request, are now described for each resource code.

Note: The values of PIF arguments as dates depend on the PIF base year, which is defined by the PIFCWB keyword on the INTFOPTS statement, or the CWBASE keyword of the INIT statement. The value of the VALTO argument for default high date depends on the PIFHD keyword of the INTFOPTS statement or the HIGHDATE keyword of the INIT statement. For more details about these statements, see *Customization and Tuning*.

List ADCOM, ADKEY arguments

Table 51. List ADCOM and ADKEY Arguments

Arg names	Length	Data type	Description
ADID	16	Char	Application description ID
ADRULEP	8	Char	Name of period or run cycle group
GROUP	8	Char	Authority group name
GROUPDEF	16	Char	Group definition ID
MONITOR	1	Char	Y=application with at least one operation monitored by an external product

Table 51. List ADCOM and ADKEY Arguments (continued)

Arg names	Length	Data type	Description
			N=application with no operation monitored by an external product
OWNER	16	Char	Owner ID
PRIORITY	4	Integer	Priority
STATUS	1	Char	Status: P=Pending A=Active
TYPE	1	Char	Application type: A=Application G=Group Default is A
VALFROM	6	Char YYMMDD	Valid-from date
VALTO	6	Char YYMMDD	Valid-to date

Note:

1. The VALTO argument value depends on the PIFHD keyword of the INTFOPTS statement, or the HIGHDATE keyword of the INIT statement. For details, see *Customization and Tuning*.
2. IBM Z Workload Scheduler assumes application type A, if you do not specify the TYPE argument name.
3. The ADSAI segment is retrieved only if the system automation information is defined for the selected application.

List AWSCL arguments

Table 52. List AWSCL Arguments

Arg names	Length	Data type	Description
DATE	6	Char YYMMDD	Date

List CLCOM arguments

Table 53. List CLCOM Arguments

Arg names	Length	Data type	Description
CALENDAR	16	Char	Calendar ID

List CPCONDCO arguments

Table 54. List CPCONDCO Arguments

Arg names	Length	Data type	Description
ADID	16	Char	Application description ID

Table 54. List CPCONDCO Arguments (continued)

Arg names	Length	Data type	Description
IA	10	Char	Input arrival date and time
OPNO	4	Integer	Operation number
CONDID	4	Integer	Condition ID. Valid values are from 1 to 999.
CONDVAL	1	Char	Final condition status: U = Undefined T = True F = False

List CPOC, CPOCCOM arguments

Note: By default, occurrences in deleted status are not retrieved when the STATUS argument is not supplied. If you do not provide the STATUS argument, the request is processed as STATUS ≠ DELETED. When the STATUS argument is specified, its value can be W, S, C, E, U, D.

List CPOPCOM arguments

Table 55. List CPOPCOM Arguments

Arg names	Length	Data type	Description
ADID	16	Char	Application description ID.
AUTFUNC	8	Char	System Automation automated function (for operation).
CLNSTAT	1	Char	Data Set cleanup status.
CLNTYPE	1	Char	Data Set cleanup type.
COMMTEXT	255	Char	System Automation command text.
COMPINFO	8	Char	System Automation completion information.
CONDRJOB	1	Char	Conditional recovery job.
DPREM	1	Char	Removable by DP.
ERRCODE	4	Char	Error code.
EXECDEST	8	Char	Execution destination. To indicate a local destination, specify ********
EXPJCL	1	Char	Expanded JCL option.

Table 55. List CPOPCOM Arguments (continued)

Arg names	Length	Data type	Description
EXTNAME	54	Char	Operation extended name.
EXTSE	16	Char	Scheduling Environment name.
GROUP	8	Char	Authority group.
IA	10	Char YYMMDDHHMM	Input arrival date and time of the occurrence.
JOBCRT	1	Char	Critical job: P=Critical path target W=Eligible for WLM assistance N=Not eligible for WLM assistance
JOBNAME	8	Char	Job name
JOBPOL	1	Char	Workload monitor late job policy. ‘ ’ (blank) = default L = Long duration D = Deadline S = Latest start time C = Conditional mode
LATEE	1	Char	Operations that are either late on their latest start time, or late on the settings for Not Started Alert or Not Started Action. Y or N.
LATEL	1	Char	Operations that are late on their latest start time. Y or N.
LATEN	1	Char	Operations that are late on the settings for Not Started Alert or Not Started Action settings. Y or N.
MONITOR	1	Char	Y = Operation monitored by an external product N = Operation not monitored by an external product
OPNO	4	Integer	Operation number.
OWNER	16	Char	Owner ID.
PRIORITY	4	Integer	Priority.
SECELEM	8	Char	System Automation security element.

Table 55. List CPOPCOM Arguments (continued)

Arg names	Length	Data type	Description
SHADOWJ	1	Char	Shadow job, Y or N.
STATUS	1	Char	Operation status.
UFNAME	16	Char	User field name.
UFVALUE	54	Char	User field value.
UNEXPRC	1	Char	Y=Unexpected RC is ON N=Unexpected RC is OFF
USRSYS	1	Char	User sysout support.
VIRTDEST	8	Char	Submission destination. To indicate a local destination, specify ********
WAITFORW	1	Char	Started on WAIT workstation, Y or N.
WAITSE	1	Char	Waiting for Scheduling Environment, N or Y.
WLMSCLS	8	Char	WLM service class.
WMPRED	1	Char	Waiting for mandatory pending predecessors, Y or N.
WPMPRED	1	Char	Waiting for either mandatory pending or pending predecessors, Y or N.
WPPRED	1	Char	Waiting for pending predecessors, Y or N.
WSNAME	4	Char	Workstation name.

Note:

1. By default, operations in deleted status are not retrieved when the STATUS argument is not supplied. If you do not provide the STATUS argument, the request is processed as STATUS ≠ DELETED.
2. The ADSAI segment is retrieved only if the system automation information is defined for the selected application.

List CPOPSRU arguments

Table 56. List CPOPSRU Arguments

Arg names	Length	Data type	Description
LISTTYPE	5	Char	Type of list, INUSE or WAITQ

Table 56. List CPOPSRU Arguments (continued)

Arg names	Length	Data type	Description
RESNAME	44	Char	Special resource name

Note: Both arguments are required. The argument value specified for RESNAME is the name of the special resource for which the In-Use list or Wait Queue is to be retrieved. Generic characters are not supported. It is processed as if MATCHTYP EXA was specified; exact match is required for record selection. The argument MATCHTYP is NOT supported.

List CPWSCOM arguments

Table 57. List CPWSCOM Arguments

Arg names	Length	Data type	Description
WSAUTO	1	Char	Automation workstation, Y or N
WSNAME	4	Char	Workstation name
WSREP	1	Char	Workstation reporting attribute
WSRETYPE	1	Char	Remote engine type (Z, D, or blank)
WSTYPE	1	Char	Workstation type
WSVIRT	1	Char	Virtual workstation, Y or N
WSWAIT	1	Char	WAIT workstation, Y or N
WSZCENTR	1	Char	z-centric workstation, Y or N

List CPWSVCOM arguments

Table 58. List CPWSVCOM Arguments

Arg names	Length	Data type	Description
WSNAME	4	Char	Virtual workstation name
WSDEST	8	Char	Virtual workstation destination. To indicate a local destination, specify ********

List CRITSUCS arguments

Table 59. List CRITSUCS Arguments

Argument name	Length	Data type	Description
ADID	16	Char	Application description ID of the job whose critical successors you want to list
IA	10	Char	Occurrence input arrival of the job whose critical successors you want to list
OPNO	4	Integer	Operation number of the job whose critical successors you want to list

Note: The Workload Service Assurance process requires at least 1 operation in the current plan that is marked as CRITICAL=P to have been processed by the latest daily planning job. If there are no such operations, any operation dynamically added to the CP is not considered critical and is not returned by ISPF option 6.7 nor by the LIST CRITSUCS request. To prevent this issue, after you dynamically add a critical operation it is required that you run a REPLAN daily planning job.

This problem can also be avoided by ensuring that there is at least 1 critical operation in the current plan. The simplest way to do this is to mark the daily planning jobs EXTEND and REPLAN as CRITICAL=P, because one of these jobs is always included in the current plan and they are critical to the operations of IBM® Z Workload Scheduler.

List CSRCOM arguments

Table 60. List CSRCOM Arguments

Arg names	Length	Data type	Description
RESALCS	1	Char	If any operation is currently allocating the resource shared, Y or N
RESAVAIL	1	Char	Whether or not the resource is available, Y or N
RESGROUP	8	Char	Resource group name
RESHIPER	1	Char	Whether or not it is a DLF control resource, Y or N
RESNAME	44	Char	Resource name
RESWAIT	1	Char	Whether or not any operation is waiting for the resource.

Note: All the arguments are optional. The argument MATCHTYP is supported.

List ETT arguments

Table 61. List ETT Arguments

Arg names	Length	Data type	Description
ADID	16	Char	Associated application ID
ETTNAME	44	Char	Name of trigger
ETTTYPE	1	Char	Type of trigger

List GENDAYS arguments

The LIST GENDAYS PIF call generates run dates for a run cycle that is provided in input by using a particular structure. The request is not linked to a job stream. It only uses calendar and periods definitions.

Table 62. List GENDAYS Arguments

Arg names	Length	Data type	Description
CALENDAR	16	Char	Calendar ID
FDAYRULE	1	Char	Free day rule
FROM	6	Char YYMMDD	One day before the `Out of effect` date
IAT	4	Char HHMM	Input arrival time
RULEDEF	*	Structure	Rule definition
TO	6	Char YYMMDD	`In effect` date

Note:

- The earliest value for FROM is the first day of the current month in a year four years previous to the current year.
- The latest value for FROM is the first day of January in a year seven years after the current year.
- The latest value for TO is the 31st of December in a year seven years after the current year.

For example, if the current date is 13/09/23, then: 090901 < FROM < 200101 and FROM < TO < 201231.

- RULEDEF is made up by a structure similar to the one used for a rule definition in ADRUN. The first four bytes declare the length of the rule, while the remaining bytes are the rule text, which is preceded by the ADRULE keyword. For example:

```
Dcl 1 ruledef,
 2 rulelen bin(31),
 2 ruletxt char(30);

rulelen = 30;
ruletxt = 'ADRULE EVERY DAY(FRIDAY) YEAR ';
```

- Your PIF program need to first run the LIST request, followed by a loop of SELECT (NEXT) on the GENDAYS resource (no SEQn support).

List JCLVCOM arguments

Table 63. List JCLVCOM Arguments

Arg names	Length	Data type	Description
JCLVTAB	16	Char	JCL variable table ID

List JLCOM arguments

Table 64. List JLCOM Arguments

Arg names	Length	Data type	Description
ADID	16	Char	Application ID
IA	10	Char YYMMDDHHMM	Input arrival date and time
JOBNAME	8	Char	z/OS® job name
OPNO	4	Integ	Operation number
WSNAME	4	Char	Workstation name

List JSCOM arguments

Table 65. List JSCOM Arguments

Arg names	Length	Data type	Description
ADID	16	Char	Application ID
IA	10	Char YYMMDDHHMM	Input arrival date and time
JOBNAME	8	Char	z/OS® job name
OPNO	4	Integer	Operation number

Table 65. List JSCOM Arguments (continued)

Arg names	Length	Data type	Description
WSNAME	4	Char	Workstation name

List LTOCCOM arguments

Table 66. List LTOCCOM Arguments

Arg names	Length	Data type	Description
ADID	16	Char	Application ID
GROUP	8	Char	Authority group
GROUPDEF	16	Char	Group definition ID
IAD	6	Char YYMMDD	Run date
IAT	4	Char HHMM	Input arrival time
OWNER	16	Char	Owner ID

List OICOM arguments

Table 67. List OICOM Arguments

Arg names	Length	Data type	Description
ADID	16	Char	Application ID
OPNO	4	Integer	Operation number

List PRCOM arguments

Table 68. List PRCOM Arguments

Arg names	Length	Data type	Description
PERIOD	8	Char	Period name
PRTYPE	1	Char	Period type

List RGCOM, RGKEY arguments

Table 69. List RGCOM, RGKEY Arguments

Arg names	Length	Data type	Description
RGID	8	Char	Run cycle group ID

Note: To list all the records of the run cycle group, run:

1. `LIST RGKEY` to obtain the first record and the total number of records in the run cycle group.
2. A loop of `SELECT RGKEY next` to list all the other records.

List SRCOM arguments

Table 70. List SRCOM Arguments

Arg names	Length	Data type	Description
RESGROUP	8	Char	Special resource group ID
RESHIPER	1	Char	DLF resource indicator
RESNAME	44	Char	Special resource name

List WSCOM arguments

Table 71. List WSCOM Arguments

Arg names	Length	Data type	Description
WSAUTO	1	Char	Automation workstation, Y or N
WSNAME	4	Char	Workstation name
WSREP	1	Char	Workstation reporting attribute
WSRETYPE	1	Char	Remote engine type: D = distributed, Z = z/OS® or blank
WSTYPE	1	Char	Workstation type
WSVIRT	1	Char	Virtual workstation, Y or N
WSWAIT	1	Char	WAIT Workstation, Y or N
WSZCENTR	1	Char	z-centric workstation, Y or N

List WSVCOM arguments

Table 72. List WSVCOM Arguments

Arg names	Length	Data type	Description
WSNAME	4	Char	Virtual workstation name
WSDEST	8	Char	Virtual workstation destination. To indicate a local destination, specify ********

Communication block address

This is the address returned by INIT request processing, which must remain unmodified for all following requests.

Return code

When EQQYCOM returns control, this fullword shows the outcome of the request:

0

The request was successful. Either all the data are returned or an incomplete list if the message EQQG009W is issued.

4

The request was unsuccessful, for one of these reasons:

- The requestor is not authorized to read the records.
- No records meet the criteria specified by the arguments.

8

The request was unsuccessful. An error message has been written to the message log data set.

MODIFY request

The MODIFY request modifies one or more fields in an LTP or current plan record. The arguments can be used both to identify the record to be modified, and to provide new values for this record. Or, the arguments can be used just to identify a record, and later requests can be used to perform particular actions. For example, with a MODIFY request, you can identify a particular current plan occurrence record. Then, with later INSERT requests, you can insert new operation records for that occurrence.

The MODIFY request can be used to modify information in the current plan. Requests that cause a modification of the current plan, except CSR requests, require a later EXECUTE request for the modification to actually take effect.

Action code

MODIFY

Resource code

CPCOND

Current plan condition segment

CPEXT

Current plan operation extended name

CPOC

Current plan occurrence

CPOP

Current plan operation

CPREND

Distributed remote job info

CPRENZ

z/OS® remote job info

CPSAI

System automation information for the current plan operation

CPUSRF

User field information for the current plan operation

CPWS

Current plan workstation

CPWSV

Current plan virtual workstation destination

CSR

Current plan special resource

IVL

Current plan workstation open interval

LTOC

LTP occurrence

VIVL

Current plan virtual workstation destination open interval

Data area

Not used.

Arguments

With the arguments described here, you specify the names and values of fields, either to identify a particular record, or provide updated information for a record.

Note: The values of PIF arguments as dates depend on the PIF base year, which is defined by the PIFCWB keyword on the INTFOPTS statement, or the CWBASE keyword of the INIT statement. The value of the VALTO argument for

default high date depends on the PIFHD keyword of the INTFOPTS statement or the HIGHDATE keyword of the INIT statement. For details about these statements, see Customization and Tuning.

Modify CPCOND arguments

When you are modifying an existing current plan condition, the CONDID argument is required to identify the condition to be modified. All remaining arguments are optional and provide the information used to modify the condition.

Note: Always identify an operation with an INSERT or MODIFY CPOP request before a MODIFY CPCOND request.

Table 73. Modify CPCOND Arguments

Arg names	Length	Data type	Description
CONDID	4	Integer	Condition ID. Valid values are from 1 to 999.
COUNT	4	Integer	Condition counter. Use it to define the rule type: 0 = All the condition dependencies, in the corresponding INSERT CPSIMP list, must be true *n>0* = At least *n* out of the specified condition dependencies must be true The default is the current value.
DESC	16	Char	Descriptive text

Modify CPEXT arguments

Create or modify the extended name of an operation in the current plan

Table 74. Modify CPEXT Arguments

Arg names	Length	Data type	Description
EXTNAME	54	Char	Operation extended name. To delete the operation extended name, enter blanks between single quotation marks or `EXTNAME=`.
EXTSE	16	Char	Scheduling Environment name. Special characters are allowed. To delete the SE name, enter blanks between single quotation marks or EXTSE=.

Modify CPOC arguments

When you are modifying an existing current plan occurrence, the ADID and IA arguments identify the occurrence to be modified. All remaining arguments provide the information used to modify the occurrence. The only valid values for the STATUS argument are W (Waiting) and C (Complete).

Table 75. Modify CPOC Arguments

Arg names	Length	Data type	Description
ADID	16	Char	Application description ID
ALLMON	1	Char	Y=all operations of occurrence monitored by an external product N=all operations of occurrence not monitored by an external product
DEADLINE	10	Char YYMMDDHHMM	Deadline date and time
ERRCODE	4	Char	Error code
GROUPDEF	16	Char	Group definition ID
IA	10	Char YYMMDDHHMM	Input arrival date and time
IANEW	10	Char YYMMDDHHMM	New input arrival date and time
JCLVTAB	16	Char	JCL variable table
PRIORITY	4	Integer	Priority
STATUS	1	Char	Occurrence status

Modify CPOP arguments

When you are modifying an existing current plan operation, the OPNO argument is required to identify the operation to be modified. All remaining arguments are optional and provide the information used to modify the operation. If you are inserting, modifying, or deleting a predecessor connection or special resource specification for the operation, the MODIFY CPOP request is required only to identify the operation that will be referred to in the following INSERT, MODIFY, or DELETE request. Then, only the OPNO argument is required.

Note: Always identify an occurrence with a MODIFY CPOC request before a MODIFY CPOP request.

Table 76. Modify CPOP Arguments

Arg names	Length	Data type	Description
AEC	1	Char	Automatic error completion

Table 76. Modify CPOP Arguments (continued)

Arg names	Length	Data type	Description
AJR	1	Char	Automatic job hold/release
ASUB	1	Char	Automatic job submission
CLATE	1	Char	Cancel if late
CLNTYPE	1	Char	Data Set cleanup type
CONDRJOB	1	Char	Conditional recovery job
DEADWTO	1	Char	Issue deadline WTO
DESC	24	Char	Operation descriptive text
DURATION	4	Integer	Estimated duration in 100th of second
EDUR	4	Char HHMM	Estimated duration
ERRCODE	4	Char	Error code
EXPJCL	1	Char	Expanded JCL option
FORM	8	Char	Form number or blanks
HRC	4	Integer	Highest successful return code
JCLASS	1	Char	Job class
JOBCRT	1	Char	Critical job: P=Critical path target W=Eligible for WLM assistance N=Not eligible for WLM assistance
JOBNAME	8	Char	Job name
JOBPOL	1	Char	Workload monitor late job policy: ‘ ’ (blank) = default L = Long duration D = Deadline S = Latest start time C = Conditional mode
MONITOR	1	Char	Y=Operation monitored by an external product N=Operation not monitored by an external product
OPCMD	2	Char	Operation command:

Table 76. Modify CPOP Arguments (continued)

Arg names	Length	Data type	Description
			BD = Bind shadow job EX = Execute operation KJ = Kill operation[1] MH = Hold operation MR = Release operation NP = NOP operation PN = Prompt reply no PY = Prompt reply yes UN = Un-NOP operation **Note:** 1. Applies only to operations running on IBM Z Workload Scheduler Agents (z-centric agents).
OPDL	10	Char YYMMDDHHMM	Operation deadline date and time or blank
OPDLACT	1	Char	The action taken if the operation does not complete at its deadline: ‘ ’ (blank) = Default. No action is taken. A = Only an alert message is issued. C = The operation is set to Complete, if its status allows it. Otherwise, it is NOPed. E = The operation is set to Error with ODEA, if its status allows it. Otherwise, this setting is postponed at the time when the status allows it. N = The operation and all its internal successors are NOPed, if their status allows NOPing. Otherwise, it is ignored.
OPIA	10	Char YYMMDDHHMM	Operation input arrival date and time or blank
OPNO	4	Integer	Operation number
PSUSE	4	Integer	Parallel servers required
R1USE	4	Integer	Resource 1 required
R2USE	4	Integer	Resource 2 required

Table 76. Modify CPOP Arguments (continued)

Arg names	Length	Data type	Description
RERUT	1	Char	Reroutable operation
RESTA	1	Char	Restartable operation
STATUS	1	Char	Operation status
TIMEDEP	1	Char	Time-dependent job
USERDATA	16	Char	Information stored in operation user data
USRSYS	1	Char	User sysout support
WLMSCLS	8	Char	WLM service class
WSNAME	4	Char	Workstation name

Note: If the argument DURATION is used with the argument EDUR, an error message occurs.

Modify CPREND arguments

Table 77. Modify CPREND Arguments

Arg names	Length	Data type	Description
COMPBNDF	1	Char	Complete if bind fails option (Y|N)
REJOBNM	40	Char	Remote job name
REJSNM	16	Char	Remote job stream name
REJSWS	16	Char	Remote job stream workstation

Note:

1. The occurrence and operation to which the remote job information refers are identified, respectively, by the INSERT and/or MODIFY CPOC (ADID, IA) and INSERT and/or MODIFY CPOP (OPNO) sequences.
2. You can use modify CPREND only if the operation runs on an remote engine workstation.
3. When you run MODIFY CPOP to modify the workstation type from remote engine to any other type, the remote job info related to the operation are automatically deleted.

Modify CPRENZ arguments

Table 78. Modify CPRENZ Arguments

Arg names	Length	Data type	Description
COMPBNDF	1	Char	Complete if bind fails option (Y|N)
READID	16	Char	Remote application name
REOPNO	4	Integer	Remote operation number

Note:

1. The occurrence and operation to which the remote job information refers are identified, respectively, by the INSERT and/or MODIFY CPOC (ADID, IA) and INSERT and/or MODIFY CPOP (OPNO) sequences.
2. You can use modify CPRENZ only if the operation runs on an remote engine workstation.
3. When you run MODIFY CPOP to modify the workstation type from remote engine to any other type, the remote job info related to the operation are automatically deleted.

Modify CPSAI arguments

Table 79. Modify CPSAI Arguments

Arg names	Length	Data type	Description
AUTFUNC	8	Char	System Automation automated function (for operation). It must be an alphanumeric value, uppercase format. The first character cannot be numeric.
COMMETXT	255	Char	System Automation command text. It must be set and cannot be blank.
COMPINFO	64	Integer	System Automation completion information.
SECELEM	8	Char	System Automation security element. It must be set and cannot be blank.

Note:

1. The occurrence and operation to which the system automation information refers are identified, respectively, by the MODIFY CPOC (ADID, IA) and MODIFY CPOP (OPNO) sequences.
2. You can use modify CPSAI only if the operation runs on an automation workstation.

Modify CPUSRF arguments

When you are modifying an existing current plan user field, the UFNAME argument is required to identify the user field to be modified. The UFVALUE argument provides the information used to modify the user field.

Note: Always identify an operation with an INSERT or MODIFY CPOP request before a MODIFY CPUSRF request.

Table 80. Modify CPUSRF Arguments

Arg names	Length	Data type	Description
UFNAME	16	Char	User field name.
UFVALUE	54	Char	User field value.

Modify CPWS arguments

When you are modifying a current plan workstation, the WSNAME argument is required; it identifies the workstation. The remaining arguments contain the modified information.

Table 81. Modify CPWS Arguments

Arg names	Length	Data type	Description
ALTWS	4	Char	When the workstation is set to failed or offline then another workstation can be specified for rerouting. Specify ALTWS if operations should be rerouted; if ALTWS is not supplied then no rerouting takes place.
PSC	1	Char	Control on parallel server.
R1C	1	Char	Control on resource 1.
R2C	1	Char	Control on resource 2.
STARTACT	1	Char	Action to be taken on current plan operations that have a status of started when the workstation status is set to failed or offline. Values are restart (R), set to error (E), or leave operation as is (L). **Note:** If the STARTACT argument is omitted when a workstation is set to failed or offline

Table 81. Modify CPWS Arguments (continued)

Arg names	Length	Data type	Description
			then no action is performed on the operations, as though STARTACT L was specified.
STATUS	1	Char	New status of active (A), failed (F), or offline (O).
WSNAME	4	Char	Workstation name
WSREP	1	Char	Workstation reporting attribute. For virtual workstations, the argument is ignored.

Modify CPWSV arguments

When you are modifying a current plan virtual workstation, the WSNAME and WSDEST arguments are required; they identify the virtual workstation destination. The remaining arguments contain the modified information.

Table 82. Modify CPWSV Arguments

Arg names	Length	Data type	Description
WSNAME	4	Char	Virtual workstation name
WSDEST	8	Char	Virtual Workstation destination. To indicate a local destination, specify ********
PSC	1	Char	Control on parallel server
R1C	1	Char	Control on resource 1
R2C	1	Char	Control on resource 2
STARTACT	1	Char	Action to be taken on current plan operations that have a status of started when the workstation status is set to failed or offline. Values are restart (R), set to error (E), or leave operation as is (L). **Note:** If the STARTACT argument is omitted when a workstation is set to failed or offline then no action is performed on the operations, as though STARTACT L was specified.
STATUS	1	Char	New status of active (A), failed (F), or offline (O)

Modify CSR arguments

MODIFY CSR takes as selection argument the resource name in the RESNAME argument. This argument is required. The resource name must be padded to the full length of 44 characters. The special resource name cannot start with a quote since it will be removed from the first position, if present, during argument parsing. It is processed as if MATCHTYP EXA was specified and an exact match is required for record selection. Alternatively, the common segment CSRCOM can be given as the selection argument. Remaining arguments are optional and contain modifications.

Table 83. Modify CSR Arguments

Arg names	Length	Data type	Description
DEFAVAIL	1	Char	Default availability, N or Y
DEFQTY	4	Integer	Default quantity, 1 to 999999
MAXLIMIT	4	Integer	Maximum usage limit. From 0 (no limit) to 999999.
MAXTYPE	1	Char	Type of action when maximum usage limit is reached: Y|N|R
ONCOMPL	1	Char	Action on complete Y|N|R
ONERROR	2	Char	Action on error, F, FX, FS, K, or blank
QUANTITY	4	Integer	Override quantity, numeric 1 to 999999, or 0 to indicate that there is no overriding quantity.
RESAVAIL	1	Char	Override availability, N, Y, or blank to indicate there is no overriding availability
RESDEVIA	4	Integer	Deviation, -999999 to 999999.
RESNAME	44	Char	Resource name
USEDFOR	1	Char	Used for C, P, B, or N

Note: MATCHTYP is NOT supported.

Modify IVL arguments

When you are modifying a workstation open interval, the FROM argument is required to identify the interval to be modified. All remaining arguments are optional and provide the information used to modify the open interval.

Note: Always identify a workstation with a MODIFY CPWS request before a MODIFY IVL request.

Table 84. Modify IVL Arguments

Arg names	Length	Data type	Description
ALTWS	4	Char	Workstation to take over if this one fails or is set offline
FROM	10	Char YYMMDDHHMM	Interval start date and time
PSCAP	4	Integer	Parallel server capacity
R1CAP	4	Integer	Resource 1 capacity
R2CAP	4	Integer	Resource 2 capacity

Modify LTOC arguments

When you are modifying an existing LTP occurrence, the ADID, IAD, and IAT arguments identify the occurrence to be modified. All remaining arguments provide the information used to modify the occurrence.

Table 85. Modify LTOC Arguments

Arg names	Length	Data type	Description
ADID	16	Char	Application description ID
DEADLINE	10	Char YYMMDDHHMM	Deadline date and time
ERRCODE	4	Char	Error code
GROUPDEF	16	Char	Group definition ID
IAD	6	Char YYMMDD	Input arrival date
IAT	4	Char HHMM	Input arrival time
JCLVTAB	16	Char	JCL variable table
PRIORITY	4	Integer	Priority

Modify VIVL arguments

When you are modifying a virtual workstation destination open interval, the FROM argument is required to identify the interval to be modified. All remaining arguments are optional and provide the information used to modify the open interval.

Note: Always identify a workstation with a MODIFY CPWSV request before a MODIFY VIVL request.

Table 86. Modify VIVL Arguments

Arg names	Length	Data type	Description
FROM	10	Char YYMMDDHHMM	Interval start date and time
PSCAP	4	Integer	Parallel server capacity
R1CAP	4	Integer	Resource 1 capacity
R2CAP	4	Integer	Resource 2 capacity

Communication block address

This is the address returned by INIT request processing, which should remain unmodified for all following requests.

Return code

When EQQYCOM returns control, this fullword shows the outcome of the request.

0

The request was successful.

4

The MODIFY CPOP request might end with return code 4 if the operation input arrival value specified in the request is earlier than the occurrence. If this happens, run the execute request for the modification to be enforced.

8

The request was unsuccessful. An error message has been written to the message log data set.

OPTIONS request

The OPTIONS request lets you specify options to be used when performing PIF requests. You can use these options to automatically:

- Resolve external dependencies when adding LTP or CP occurrences
- Improve the time taken to retrieve information about operations
- Request the address of the area where the message ID is returned
- Prevent messages being written to the message log.
- Handles different versions of the same application. If you delete, insert or replace an application, this operation might cause the change of the valid-to date of all versions involved. By default, different versions of the same application are not supported.

Automatic resolution of external dependencies involves:

- The external predecessors of the occurrence you are inserting are in the LTP or current plan. In the LTP, if more than one occurrence of a specified predecessor application occurs, IBM Z Workload Scheduler selects as predecessor the one with the nearest earlier input arrival time. In the current plan, if more than one occurrence of a specified predecessor application occurs, IBM Z Workload Scheduler selects as predecessor the one with the nearest earlier input arrival time and containing a candidate predecessor.
- All predecessor occurrences selected by the preceding rule are updated so that they specify the new occurrence as a successor.

If automatic resolution is not required, external dependencies that exist in the application descriptions you are inserting are removed before the LTP or current plan is updated.

By default, automatic resolution is *not* performed. When you use the OPTIONS request, the option you choose remains in effect until the end of the current program interface session or until altered by another OPTIONS request. An OPTIONS request can be made any time after the INIT request.

Action code

OPTIONS

Resource code

Not used.

Data area

The data area is used only if the RETMSG or RETMSGID argument name is specified.

The data area address is set to locate an area for a message ID. This address is available on return from the OPTIONS request. At each subsequent program interface request (excluding the TERM request), the ID of an issued message is returned in this area.

The first 3 characters of the returned message ID are **MSG**. The last character is either:

I

Information

W

Warning

E

Error

Blank

If the message is suppressed by the SUPMSG OPTIONS request.

The message ID area is blank if no message is issued for a request. If a program interface request causes more than one message to be written to the message log, the message returned is the one considered to be the highest severity. The

severity levels are E, blank, W, and I. The highest severity is E (error), and the lowest severity is I (information). If more than one message has the same severity level, the first message issued takes precedence.

When RETMSG is specified, the message ID is part of a larger area. The data available at the address returned is:

Returned message area

Offset	Length	Type	Description
-4	2	Int	Text line length
-2	2	Int	Number of text lines
0	8	Char	Message ID
+8	*	Char	Text lines, the length is the number of text lines multiplied by the text line length

Arguments

Argument name=LTDEPR

Automatic resolution of external dependencies when inserting new LTP occurrences.

Argument value=Y

Yes.

Argument value=N

No (default).

Argument name=CPDEPR

Automatic resolution of external dependencies when inserting new current plan occurrences.

Argument value=Y

Add successor and predecessor dependencies.

Argument value=N

Do not add any dependencies (default).

Argument value=P

Add predecessor dependencies.

Argument value=S

Add successor dependencies.

Argument name=FASTPATH

FASTPATH can make the search for operations considerably faster when you want only to retrieve computer and printer operations.

Argument value=Y

If you specify Y (YES), IBM Z Workload Scheduler searches the current plan for computer or printer operations matching the job name search argument. It then selects *all* operations in the

occurrences that contain these computer or printer operations (that is, even operations at general workstations), and retrieves these operations based on the remaining search arguments that you have specified.

Argument value=N

If you specify N (NO), which is the default value, all operations are retrieved that match the search argument criteria that you have specified.

Argument name=RETMSG

This argument lets you request the address of the area where the message ID and message text is returned. The address points to the message ID, the layout of the area is described in the paragraph Data area on page 93. There is no argument value for this argument name.

Argument name=RETMSGID

This argument lets you request the address of the area where the message ID is returned. There is no argument value for this argument name.

Argument name=SUPMSG

SUPMSG lets you prevent a message from being written to the message log. You can prevent more than one message from being written to message log by issuing multiple OPTIONS requests with the SUPMSG argument specified.

Argument value=MSG*msgid*

Specify MSG followed by the message identifier. To obtain the message identifier, remove the IBM Z Workload Scheduler prefix (EQQ) from the beginning of the message and the severity indicator from the end of the message.

For example, to prevent message EQQW002E from being written to the message log, specify an argument value of MSGW002.

Argument name=ADVERS

Application description versions support when delete, insert or replace an AD record.

Argument value=Y

Yes. When inserting or replacing an AD record, and another record with the same ADID exists, the VALTO and VALFROM values will be set so that the different versions of the application have consecutive validity intervals, with the same logic used by the ISPF dialogs.

Argument value=N

No (default). The AD record is stored as provided by the user.

Argument name=ADOICHK

Use this option to specify whether or not you want AD/OI consistency checks to be made every time an application is deleted or modified.

Consistency checks involve looking in the application description data base for matches for all the operator instructions in the application. Any operator instructions without a match are deleted.

The checks are made immediately after the application description PIF action has completed with a zero return code.

Argument value=Y

Yes. Consistency checks are performed whenever an application description record is deleted or replaced using the PIF.

Argument value=N

No (default). Consistency checks are not performed.

Argument name=VERADGRD

Application descriptions that are members of an application group have the name of the group definition in field ADGROUPID of segment ADCOM. VERADGRD controls the verification of this field when a new application description is created or an existing one is replaced. The verification is done for active application descriptions.

Argument value=F

The group definition is verified to check that it exists, is active and valid for at least a part of the validity period of the application description being created or updated.

Argument value=Y

Same as for value=F, except that the application group id is accepted if the application description already has this application group id. It could be an update without any change to the application group id or an insert of a new version when there already are active versions with the same application group id.

Argument value=N (default)

No check is made to verify that the application group exists.

Argument name=VERSRWSN

The special resource description, SR, has fields representing workstations, the full workstation names or generic names; field SRDWSNAME of segment SRDWS for default connected workstations, field SRIWSNAME of segment SRIWS for workstations connected to an interval. VERSRWSN controls the verification of these fields when a new special resource is created or an existing one is replaced.

Argument value=F

The workstation fields are verified against the workstation description file. Each workstation field in the resource description must match at least one of the workstation descriptions.

Argument value=Y

Same as for value=F except that the workstation value is accepted if the resource description already has this workstation name. It could be an update without any change to the workstation names.

Argument value=N (default)

No check is made to verify that the workstation description exists.

Argument name=DURSEC

This argument lets you decide the duration format of Insert and Replace Action of AD/WS record. ADOPDUR and WSOPDUR fields contain duration value in minutes. APOPDURI and WSOPDURI fields contain duration value in hundredths of a second. If DURSEC is not specified, Adopur/Wsopdur value will be used.

Argument Value=Y

Adopduri/Wsopduri will be always used.

Argument Value=N

The field Adopduri/Wsopduri will be checked to have the same value of the field Adopdur/Wsopdur when the field Adopduri/Wsopduri is rounded up to a number of minutes. If this happens, it means that no change occurred in the duration value and the field Adopduri/Wsopduri will be used. If the Adopduri/Wsopduri value is different from the Adopdur/Wsopdur one, it means that the user changed duration value in Adopdur/Wsopdur and this field will be used.

Communication block address

This is the address returned by INIT request processing, which should remain unmodified for all following requests.

Return code

When EQQYCOM returns control, this fullword shows the outcome of the request:

0

The request was successful.

8

The request was unsuccessful. An error message has been written to the message log data set.

REPLACE request

The REPLACE request replaces an existing record in the application description or operator instruction database with a record provided by your program. If the record type is other than an application description record, then the record provided by your program must have the same key fields as a record on the database; otherwise, no replace is performed.

If the record type is an application description record, then the record provided by your program can have the *STATUS* field modified, even if this field is part of the key. In this case, you must supply the old STATUS value and the VALTO value of the application to be replaced in the arguments. You must also set the ADVERS argument value to Y in the OPTIONS request as Y as well.

The replacing record is placed in the data area by your program. Arguments are not used if the resource code is different from AD or if you set the ADVERS argument value in the OPTIONS request to N.

Action code

REPLACE

Resource code

The resource code identifies the record type you want to replace. You can specify these values:

AD

Application description record

AWSCL

All workstation closed record

CL

Calendar record

CSR

Current plan special resource

ETT

Event triggered tracking criteria record

JCLV

JCL variable table record

JS

Job control language record

OI

Operator instruction record

PR

Period record

RG

Run cycle group record

SR

Special resource record

WS

Workstation record

WSV

Virtual workstation destination record

Notes:

1. If you do not provide the application description (AD) record TYPE, or the AD record TYPE is not recognized, *application* is assumed. The priority field is not used for an AD group definition
2. The format of duration used in the data area, in Replace AD/WS will be defined by the DURSEC option, described in the paragraph OPTIONS request on page 92

Data area

You must put the address of your data area in the fullword whose address is in the parameter list. The data area consists of a header and the actual record to be written to the database. Ensure that the header and data record are in the correct format. For a description of the format for a header, see Header format on page 26. Appendix A. Program Interface Record Format describes the format for the data records.

In the CSRCOM segment of the CSR record only a subset of the fields can be changed:

CSRUSEDFOR
CSRONERROR
CSROVAV
CSROVQ
CSRDEVI
CSRDEFQUANT
CSRDEFAVAIL

The values in the rest of the CSRCOM fields are ignored and the values in the resource record are left unchanged.

Arguments

Replace AD arguments

Table 87. Replace AD Arguments

Arg names	Length	Data type	Description
STATUS	1	Char	Status: P=Pending; A=Active
VALTO	6	Char YYMMDD	Valid-to date

For resource codes other than AD no arguments are supported. The new record must be made available via the data address parameter.

Communication block address

This is the address returned by INIT request processing, which should remain unmodified for all following requests.

Return code

When EQQYCOM returns control, this fullword shows the outcome of the request:

0

The request was successful.

4

The record; AD, AWSCL, CL, ETT, JS, JCLPREP, JCLPREPA, JCLV, OI, PR, SR, or WS is being updated by another user. The record is replaced.

8

The request was unsuccessful. An error message has been written to the message log data set.

RESET request

The RESET request deletes the current MCP block. This effectively cancels a series of modify current plan requests that have been collected in an MCP block, if it is performed before an EXECUTE request.

RESET is required when an error occurs, if you have made more than 1 MODIFY or INSERT CPOC request before an EXECUTE request. If you do not specify RESET, successful MODIFY or INSERT requests are processed in the next EXECUTE MCPBLK request.

Action code

RESET

Resource code

About this task

MCPBLK

Data area

About this task

Not used.

Arguments

About this task

Not used.

Communication block address

About this task

This is the address returned by INIT request processing, which should remain unmodified for all following requests.

Return code

About this task

When EQQYCOM returns control, this fullword shows the outcome of the request:

0

The request was successful.

8

The request was unsuccessful. An error message has been written to the message log data set.

SELECT request

About this task

The SELECT request retrieves a record and makes it available to your program. You can:

- Retrieve a record directly from IBM Z Workload Scheduler by specifying field names and values in arguments, which identify the record you want to retrieve. When you retrieve a record directly from IBM Z Workload Scheduler, you can get the complete record rather than just the common segment that is available from a list.
- Retrieve one of the records from a list built by a previous LIST or SELECT request by providing the resource code (common segment name), and the pointer to the offset of the common segment data area that contains the common segment of the record. This pointer is in the header record of the common segment.

Action code

About this task

SELECT

Resource code

About this task

If you want to retrieve one of the records from a previously built list, *you must use the same resource code that you used when you built the list with the LIST request*. The arguments NEXT, PREV, FIRST, and LAST direct the selection to a list. The resource code shows which list previously built contains the required record. There can be a maximum of one active list for each resource code.

If you want to retrieve a record directly from IBM Z Workload Scheduler, the resource code indicates the record type. You can specify these values:

AD

Application description record

ADCOM

Application description, common segment only

AWSCL

All workstations closed record

CL

Calendar record

CLCOM

Calendar record, common segment

CPCOND

Current plan condition segment

CPCONDCO

Current plan condition common segment

CPOC, CPOCCOM

Current plan occurrence record

CPOP

Current plan operation record

CPOPCOM

Current plan operation record, common segment

CPOPSRU

Current plan operation segment with information about the operation in relation to a special resource

CPST

Current plan status record

CPUSRF

Current plan user field record

CPWS

Current plan workstation record

CPWSV

Current plan virtual workstation destination record

CPWSCOM

Current plan workstation record, common segment

CPWSCOM

Current plan virtual workstation destination record, common segment

CSR

Current plan special resource

CSRCOM

Current plan special resource, common segment

ETT

Event triggered tracking criteria record

JCLPREP

Retrieve promptable setup variables for the current operation

JCLPREPA

Resolve all nonpromptable setup variables for the current operation

JCLV

JCL variable table record

JCLVCOM

JCL variable table record, common segment

JLCOM

JS file job log common segment

JS

Job control language record

JSCOM

Job control language record, common segment

LTOC

LTP occurrence record

LTOCCOM

LTP occurrence record, common segment

OI

Operator instruction record

OICOM

Operator instruction record, common segment

PR

Period record

PRCOM

Period record, common segment

RG

Run cycle group record

RGCOM

Run cycle group common segment

SR

Special resource record

WS

Workstation description record

WSCOM

Workstation description record, common segment

WSV

Virtual workstation destination record

WSVCOM

Virtual workstation destination record, common segment

Note:

1. The SELECT JS and SELECT JSCOM requests try to retrieve JCL from the JCL repository. If no JCL is found, it is retrieved from the JCL library or through the job-library-read exit, EQQUX002. The full key is required, that is, the application ID, the input arrival time, and the operation number. You might need to precede the SELECT JS request by a LIST CPOPCOM request to get the key values.
2. LIST JSCOM requests try to retrieve JCL only from the JCL repository.
3. SELECT CPOPSRU can be issued for list elements only, from a list created by LIST CPOPSRU.

Data area

About this task

When EQQYCOM returns control to your program after a successful SELECT request, this fullword contains the address of the data area containing the requested record.

If you are retrieving a record from a list, only the common segment of the record is returned. A description of the fields in the common segment of each record can be found in Appendix A. Program Interface Record Format.

If you are retrieving a record directly from IBM Z Workload Scheduler, the complete record with all segments can be returned, depending on the resource type. A description of the segments in each record and the fields in each segment can be found in Appendix A. Program Interface Record Format.

The header section for this record contains, besides the normal header information, a field containing one of these items:

- The index number of the record in the list, if the record was retrieved from a LIST. For example, **1** for the first record in the list, **2** for the second.
- The length of the data area (header and data), if the record was not retrieved from a LIST.

This field is in the final header entry, that is, the entry that has a blank segment name field. The count is stored in the field that normally contains the segment offset. For a complete description of headers, see Header format on page 26.

Arguments

About this task

Retrieving a record from a list

If you want to retrieve a record from a list built by a previous LIST request, you must use one of these argument names:

NEXT

Retrieve the next record from the list.

PREV

Retrieve the previous record from the list.

FIRST

Retrieve the first record from the list.

LAST

Retrieve the last record from the list.

A corresponding argument value is not used.

When a LIST is created, IBM Z Workload Scheduler sets the first element in the list as the *current* element. Each time a SELECT request is performed on a list, the current element is updated according to which of the these argument names was used. If you have several lists active, IBM Z Workload Scheduler remembers the current element for each of them.

In combination with one of the above arguments, you can use one or more arguments described in Retrieving a record directly from IBM Z Workload Scheduler on page 106. This is best illustrated with an example:

Example

Figure 4. Example of arguments for processing a list

```
Action code:   SELECT

Resource code: ADCOM

Argument names:          Values:
                NEXT             -
                STATUS           A
                PRIORITY         9
```

Assuming a previously successful LIST request has executed for the ADCOM resource, the parameters in this example cause IBM Z Workload Scheduler to search the ADCOM list forward from the current element until it finds an element with STATUS A and PRIORITY 9.

This example gives you a mechanism for processing the list you have previously built using a LIST request. After a successful SELECT request, the required record from the list is available in the data area.

Retrieving a record directly from IBM Z Workload Scheduler

When you are retrieving a record directly from IBM Z Workload Scheduler as opposed to a record from a list, the arguments identify which record you want to retrieve. Two ways you can do this are:

- Specify field names of the record as argument names. The argument values specify values for these fields that identify the particular record you want to retrieve. Argument values can be:
  - Character values. A blank character terminates the field.
  - Numeric values, which must occupy a fullword.

  You must specify sufficient arguments to *uniquely* identify a record. You can use a comparison operator after the argument values. The default, an equals sign (=), is assumed if you do not.
- Use the common part of the record, which you have previously retrieved with a LIST or a SELECT request, to identify the required record. Here the argument name specifies the resource code (common segment name), and the argument value specifies the address of the common segment data area that contains the common segment of the record. See Table 4: Records Using a Common Segment on page 32.

CPST (current plan status) is only one record; therefore, select arguments are not required.

Note: The values of PIF arguments as dates depend on the PIF base year, which is defined by the PIFCWB keyword on the INTFOPTS statement, or the CWBASE keyword of the INIT statement. The value of the VALTO argument for default high date depends on the PIFHD keyword of the INTFOPTS statement or the HIGHDATE keyword of the INIT statement. For details about these statements, see *IBM Z Workload Scheduler: Customization and Tuning*.

You can specify the following values.

Select AD, ADCOM arguments

Table 88. Select AD, ADCOM Arguments

Arg names	Length	Data type	Description
ADID	16	Char	Application description ID
ADRULEP	8	Char	Name of period or run cycle group
GROUP	8	Char	Authority group name
GROUPDEF	16	Char	Group definition ID

Table 88. Select AD, ADCOM Arguments (continued)

Arg names	Length	Data type	Description
MONITOR	1	Char	Y=application with at least one operation monitored by an external product N=application with no operation monitored by an external product
OWNER	16	Char	Owner ID
PRIORITY	4	Integer	Priority
STATUS	1	Char	Status: P=Pending A=Active
TYPE	1	Char	Application type: A=Application G=Group. Default is A
VALFROM	6	Char YYMMDD	Valid-from date
VALTO	6	Char YYMMDD	Valid-to date

Note:

1. IBM Z Workload Scheduler assumes application type A if you do not specify the AD argument name TYPE.
2. The ADSAI segment is retrieved only if the system automation information is defined for the selected application.

Select AWSCL arguments

Table 89. Select AWSCL Arguments

Arg names	Length	Data type	Description
DATE	6	Char YYMMDD	Date

Select CL, CLCOM arguments

Table 90. Select CL, CLCOM Arguments

Arg names	Length	Data type	Description
CALENDAR	16	Char	Calendar ID

Note: If the name of the default calendar is specified in the EQQYPARM INIT statement, SELECT CL without the CALENDAR argument will return the default calendar. Otherwise CALENDAR is a required argument.

Select CPCOND, CPCONDCO arguments

Table 91. Select CPCOND, CPCONDCO Arguments

Arg names	Length	Data type	Description
ADID	16	Char	Application description ID
IA	10	Char	Input arrival date and time
OPNO	4	Integer	Operation number
CONDID	4	Integer	Condition ID. Valid values are from 1 to 999.
CONDVAL	4	Char	Final condition status: U = Undefined T = True F = False

Select CPOC, CPOCCOM arguments

Table 92. Select CPOC Arguments

Arg names	Length	Data type	Description
ADID	16	Char	Application description ID
GROUP	8	Char	Authority group
GROUPDEF	16	Char	Group definition ID
IA	10	Char YYMMDDHHMM	Input arrival date and time
MCPADDED	1	Char	MCP added, Y or N
MONITOR	1	Char	Y=occurrence with at least one operation monitored by an external product N=occurrence with no operation monitored by an external product
OWNER	16	Char	Owner ID
PRIORITY	4	Integer	Priority
RERUN	1	Char	Rerun requested, Y or N
STATUS	1	Char	Occurrence status

Note: When the STATUS argument is specified, its value can be W, S, C, E, U, D.

Select CPOP, CPOPCOM arguments

Table 93. Select CPOP, CPOPCOM Arguments

Arg names	Length	Data type	Description
ADID	16	Char	Application description ID.
CLNSTAT	1	Char	Data Set cleanup status.
CLNTYPE	1	Char	Data Set cleanup type.
CONDRJOB	1	Char	Conditional recovery job.
DPREM	1	Char	Removable by DP.
ERRCODE	4	Char	Error code.
EXECDEST	8	Char	Execution destination. To indicate a local destination, specify ********
EXPJCL	1	Char	Expanded JCL option.
EXTNAME	54	Char	Operation extended name.
EXTSE	16	Char	Scheduling Environment name.
GROUP	8	Char	Authority group.
IA	10	Char YYMMDDHHMM	Input arrival date and time.
JOBCRT	1	Char	Critical job: P=Critical path target W=Eligible for WLM assistance N=Not eligible for WLM assistance
JOBNAME	8	Char	Job name.
JOBPOL	1	Char	Workload monitor late job policy: ' ' (blank) = default L = Long duration D = Deadline S = Latest start time C = Conditional mode
LATEE	1	Char	Operations that are either late on their latest start time, or late on Not Started Alert/Action settings. Y or N.

Table 93. Select CPOP, CPOPCOM Arguments (continued)

Arg names	Length	Data type	Description
LATEL	1	Char	Operations that are late on their latest start time. Y or N.
LATEN	1	Char	Operations that are late on Not Started Alert/Action settings. Y or N.
MONITOR	1	Char	Y = Operation monitored by an external product N = Operation not monitored by an external product
OPNO	4	Integer	Operation number.
OWNER	16	Char	Owner ID.
PRIORITY	4	Integer	Priority.
SHADOWJ	1	Char	Shadow job, Y or N.
STATUS	1	Char	Operation status.
UNEXPRC	1	Char	Y=Unexpected RC is ON N=Unexpected RC is OFF
USRSYS	1	Char	User sysout support.
VIRTDEST	8	Char	Submission destination. To indicate a local destination, specify ********
WAITFORW	1	Char	Started on WAIT workstation, Y or N.
WAITNAME	1	Char	Waiting for Scheduling Environment, Y or N.
WLMSCLS	8	Char	WLM service class.
WMPRED	1	Char	Waiting for mandatory pending predecessors, Y or N.
WPMPRED	1	Char	Waiting for either mandatory pending or pending predecessors, Y or N.
WPPRED	1	Char	Waiting for pending predecessors, Y or N.
WSNAME	4	Char	Workstation name.

Note: The ADSAI segment is retrieved only if the system automation information is defined for the selected application.

Select CPUSRF arguments

By running the Select CPUSRF, the CPUSRFELEM segment is retrieved for all the user fields related to the operation.

Table 94. Select CPUSRF Arguments

Arg names	Length	Data type	Description
ADID	16	Char	Application description ID
IA	10	Char	Input arrival date and time
OPNO	4	Integer	Operation number

Select CPWS, CPWSCOM arguments

Table 95. Select CPWS, CPWSCOM Arguments

Arg names	Length	Data type	Description
WSAUTO	1	Char	Automation workstation, Y or N
WSNAME	4	Char	Workstation name
WSREP	1	Char	Workstation reporting attribute
WSRETYPE	1	Char	Remote engine type (Z, D, or blank)
WSTYPE	1	Char	Workstation type
WSVIRT	1	Char	Virtual workstation, Y or N
WSWAIT	1	Char	WAIT Workstation, Y or N
WSZCENTR	1	Char	z-centric workstation, Y or N

Select CPWSV, CPWSVCOM arguments

About this task

Table 96. Select CPWSV, CPWSVCOM Arguments

Arg names	Length	Data type	Description
WSNAME	4	Char	Virtual workstation name
WSDEST	8	Char	Virtual workstation destination. To indicate a local destination, specify ********

Select CSR, CSRCOM arguments

About this task

Table 97. Select CSR, CSRCOM Arguments

Arg names	Length	Data type	Description
RESALCS	1	Char	Whether or not any operation is currently allocating the resource shared, Y or N
RESAVAIL	1	Char	Whether or not the resource is available, Y or N
RESGROUP	8	Char	Resource group name
RESHIPER	1	Char	Whether or not it is a DLF control resource, Y or N
RESNAME	44	Char	Resource name
RESWAIT	1	Char	Whether or not any operation is waiting for the resource.

The argument MATCHTYP is supported.

Select ETT arguments

About this task

Table 98. Select ETT Arguments

Arg names	Length	Data type	Description
ADID	16	Char	Associated application ID
ETTNAME	44	Char	Name of trigger
ETTTYPE	1	Char	Type of trigger: 2=job 3=special resource

Select JCLPREP arguments

Table 99. Select JCLPREP Arguments

Arg names	Length	Data type	Description
ADID	16	Char	Application ID
IA	10	Char YYMMDDHHMM	Input arrival date and time
OPNO	4	Integer	Operation number

JCL preparation using PIF on page 119 describes JCL preparation using the program interface.

Select JCLPREPA arguments

About this task

Table 100. Select JCLPREPA Arguments

Arg names	Length	Data type	Description
ADID	16	Char	Application ID.
IA	10	Char YYMMDDHHMM	Input arrival date and time.
OPNO	4	Integer	Operation number.
SIMTIME	12	Char CCYYMMDDHHMM	Simulated time. CCYY can have the values 1984 to 2071.
SIMTYPE	8	Char *"FULL"* or *"PARTIAL"*	Simulation type.

JCL preparation using PIF on page 119 describes JCL preparation using the program interface.

Select JCLV, JCLVCOM arguments

Table 101. Select JCLV, JCLVCOM Arguments

Arg names	Length	Data type	Description
JCLVTAB	16	Char	JCL variable table ID

Select JLCOM arguments

About this task

Table 102. Select JLCOM Arguments

Arg names	Length	Data type	Description
ADID	16	Char	Application ID
IA	10	Char YYMMDDHHMM	Input arrival date and time
JOBNAME	8	Char	z/OS® job name
OPNO	4	Integer	Operation number
WSNAME	4	Char	Workstation name

Select JS, JSCOM arguments

About this task

Table 103. Select JS, JSCOM Arguments

Arg names	Length	Data type	Description
ADID	16	Char	Application ID
IA	10	Char YYMMDDHHMM	Input arrival date and time
JOBNAME	8	Char	z/OS® job name
OPNO	4	Integer	Operation number
WSNAME	4	Char	Workstation name

Select LTOC, LTOCCOM arguments

Table 104. Select LTOC, LTOCCOM Arguments

Arg names	Length	Data type	Description
ADID	16	Char	Application ID
GROUP	8	Char	Authority group
GROUPDEF	16	Char	Group definition ID
IAD	6	Char YYMMDD	Input arrival date
IAT	4	Char HHMM	Input arrival time
OWNER	16	Char	Owner ID

Select OI, OICOM arguments

Table 105. Select OI, OICOM Arguments

Arg names	Length	Data type	Description
ADID	16	Char	Application ID
OPNO	4	Integer	Operation number
VALTO	10	Char (YYMMDDHHMM)	Valid-to date and time

Select PR, PRCOM arguments

About this task

Table 106. Select PR, PRCOM Arguments

Arg names	Length	Data type	Description
PERIOD	8	Char	Period name
PRTYPE	1	Char	Period type

Select RG, RGCOM arguments

Table 107. Select RG, RGCOM Arguments

Arg names	Length	Data type	Description
RGID	8	Char	Run cycle group ID
RGOWNER	16	Char	Run cycle group owner
RGCALEND	16	Char	Run cycle group calendar
RGVARTAB	16	Char	Run cycle group variable table
RUNNAME	8	Char	Run cycle name
RUNCAL	16	Char	Run cycle calendar
RUNVTAB	16	Char	Run cycle variable table
RUNSETID	8	Char	Run cycle subset ID

Select SR, SRCOM arguments

About this task

Table 108. Select SR, SRCOM Arguments

Arg names	Length	Data type	Description
RESGROUP	8	Char	Special resource group ID
RESHIPER	1	Char	DLF resource indicator
RESNAME	44	Char	Special resource name

Select WS, WSCOM arguments

Table 109. Select WS, WSCOM Arguments

Arg names	Length	Data type	Description
WSAUTO	1	Char	Automation workstation, Y or N
WSNAME	4	Char	Workstation name
WSREP	1	Char	Workstation reporting attribute
WSRETYPE	1	Char	Remote engine type: D = distributed, Z = z/OS® or blank
WSTYPE	1	Char	Workstation type

Table 109. Select WS, WSCOM Arguments (continued)

Arg names	Length	Data type	Description
WSVIRT	1	Char	Virtual workstation,Y or N
WSWAIT	1	Char	WAIT workstation, Y or N
WSZCENTR	1	Char	z-centric workstation, Y or N

Select WSV, WSVCOM arguments

Table 110. Select WSV, WSVCOM Arguments

Arg names	Length	Data type	Description
WSNAME	4	Char	Virtual workstation name
WSDEST	8	Char	Virtual workstation destination. To indicate a local destination, specify ********

Selecting a record using a common segment

If you have already retrieved the common segment of a record but you then want to retrieve the entire record, you can specify the segment name as an argument name and the address of the previously retrieved common segment as the argument value address.

For current plan operations, segment CPOPSRU can be used as well as the common segment.

Communication block address

About this task

This is the address returned by INIT request processing, which should remain unmodified for all following requests.

Return code

About this task

When EQQYCOM returns control, this fullword shows the outcome of the request:

0

The request was successful.

4

The request was unsuccessful.

No records meet the criteria specified by the arguments.

6

You are not authorized to read the record. You specified a unique key in the SELECT request; the record exists, but you do not have authority to read it.

8

The request was unsuccessful. An error message has been written to the message log data set. This can occur if more than one record in the database satisfies the field values specified by your arguments. For example, you want to select an application description record with the ID APPL1, and there are two such application descriptions in the database with different validity dates. Your arguments must specify both the application ID and the valid-from date to uniquely identify the record.

SETSTAT request

The SETSTAT request changes the condition status from undecided to true or false, if the original status is undecided because of missing step-end information.

It produces the same result as the T and F commands available from the MCP dialog.

Action code

SETSTAT

Resource code

About this task

CPSIMP

Data area

About this task

Not used.

Arguments

About this task

The arguments identify which condition dependency with undefined status is to be reset.

The same arguments apply as for the INSERT CPSIMP request, listed in Table 41: Insert CPSIMP Arguments on page 59.

To identify the new status, use the following argument:

Table 111. Setstat CPSIMP Argument

Arg name	Length	Data type	Description
NEWSTAT	1	Char	Requested status:

Table 111. Setstat CPSIMP Argument (continued)

Arg name	Length	Data type	Description
			T = True F = False

Communication block address

This is the address returned by INIT request processing, which must remain unmodified for all following requests.

Return code

When EQQYCOM returns control, this fullword shows the outcome of the request:

0

The request was successful.

8

The request was unsuccessful. An error message has been written to the message log data set.

TERM request

About this task

The TERM request terminates the program interface session and performs this cleanup processing:

- FREEMAIN of storage
- Close data sets
- Detach subtasks
- Termination of the IBM Z Workload Scheduler session.

It must be the last request of a session. A TERM request is necessary if the INIT request executed successfully.

Action code

TERM

Resource code

Not used.

Data area

Not used.

Arguments

About this task

Not used.

Communication block address

This is the address returned by INIT request processing, which should remain unmodified for all following requests, including the TERM request.

Return code

When EQQYCOM returns control, this fullword shows the outcome of the request:

0

The request was successful. A program interface session has been successfully terminated.

8

The request was unsuccessful. An error message has been written to the message log data set.

Note: If EQQYCOM abends, processing is immediately interrupted and the control is passed to the system, therefore no return code is provided. Resolve the abend according to the programming language you are using.

JCL preparation using PIF

You can perform JCL preparation through the program interface using resource type **JS** for JCL records, or **JCLPREP** for promptable variables. You can use the resource type JCLPREPA rather than a combination of JS and JCLPREP requests. You can also use **JCLPREPA** to simulate variable substitution. This lets you perform *trial substitution* of your variables without updating a job.

For details about variable substitution and job tailoring, see *Managing the Workload*.

Substituting variables

JCL preparation can be:

SETUP=PROMPT

A user must assign the value.

SETUP=YES

A value is automatically assigned at JCL preparation, or at submit time if no JCL preparation is performed.

SETUP=NO

A value is assigned at submit time.

SELECT JCLPREP retrieves the promptable variables that do not have a value assigned. When returned, the data area parameter locates a JCL setup variable record, the header, a common segment, and a sequence of variable segments JSVV. The field JSVVVALUE of the JSVV segment can be assigned a new value.

INSERT JCLPREP is used to make the promptable variables in the JCL setup variable record assigned to the JCL record. When all promptable variables of the JCL record are assigned, SELECT JCLPREP receives a return code 4.

To update the JCL record, you must execute a SELECT JS request followed by an INSERT or REPLACE JS request. When the SELECT JS is returned, the retrieved JCL record will have promptable variables resolved if a value was assigned in the SELECT JCLPREP, INSERT JCLPREP sequence. Nonpromptable setup variables are also resolved, while submit variables remain unresolved. An INSERT or REPLACE JS request is required to have the updated JCL reflected in the database and must be complete to end the JCL preparation session.

If the JCL record is not present on the JS file, an INSERT JS request is required. A LIST JS request will get return code 4 if the JCL record is not found in the JS file. SELECT JS will retrieve the JCL from the job library EQQJBLIB.

If the JCL record does not contain promptable variables, SELECT JCLPREPA must be used to assign values to nonpromptable setup variables. So, if the first SELECT JCLPREP results in return code 4, a SELECT JCLPREPA must be executed instead of the SELECT JS before the INSERT or REPLACE JCL request.

The sample library member EQQPIFAP contains a sample program that resolves JCL variables using the program interface. See Sample library (SEQQSAMP) on page 489 for more information about individual members of the sample library.

Example of a PIF request logic flow:

```
INIT
OPTIONS
DO while(RC=0)
  SELECT JCLPREP      (opno of the JOB operation)
    set up the prompt var
      INSERT JCLPREP  (opno of the JOB operation)
END
LIST JSCOM            (opno and wsn of JOB operation)
SELECT JS             (opno and wsn of JOB operation)
 check RC from the LIST JSCOM
 if RC=0 then
  REPLACE JS
 if RC=4 then
  INSERT  JS
TERM
```

Note: If there is a SETUP operation for this computer operation, and if you want to set it to Complete, add the following statements before the TERM request:

```
MODIFY CPOC
MODIFY CPOP           (opno and wsn of SETUP operation)
EXECUTE
```

For a description of the SETUP and JOB setup operations, see *Managing the Workload*.

Simulating variable substitution

You can use JCLPREPA arguments to perform trial substitutions, before normal substitution by IBM Z Workload Scheduler. You might need to do this, for example, if you use a product that checks JCL.

You can request partial or full simulation. For partial simulation, only nonpromptable setup variables are substituted. For full simulation:

- Submit variables are substituted.
- Nonpromptable setup variables are substituted.
- Promptable setup variables are substituted using the default values. You must specify the defaults when calling PIF, otherwise no substitution takes place and the JCL might contain &, ?, and % characters.
- PHASE=SETUP directives are returned to the caller, even though IBM Z Workload Scheduler only simulates submission.
- You can supply a time value in the SIMTIME argument for IBM Z Workload Scheduler-supplied variables that contain a *current time* value. IBM Z Workload Scheduler uses the current time if you do not specify SIMTIME.
- JCL is returned even if errors were found, except for the case when the JCL exceeds the JS size. Error and warning messages are inserted in the JCL.

Chapter 2. The Application Programming Interface (API)

This chapter explains how you use the IBM Z Workload Scheduler application programming interface (API) to communicate with IBM Z Workload Scheduler. Through the API you can:

- Extract information about the current plan (GET request)
- Update or add current-plan operations (PUT request)
- Delete operations in the current plan (DEL request)
- Report events to IBM Z Workload Scheduler (CREATE request).

IBM Z Workload Scheduler uses the services of APPC to communicate with an application transaction program (ATP). Before you can use the API, IBM Z Workload Scheduler support for APPC must be active. For details, see *IBM® Z Workload Scheduler: Planning and Installation*.

This chapter describes CPI-C verbs that are supported by IBM Z Workload Scheduler. ATPs that use CPI-C are more easily integrated and transported across supported environments. For more information about CPI-C verbs, refer to *CPI-C Communications Reference*.

Samples are provided with IBM Z Workload Scheduler to help you set up and use the API. For a description of these samples, see Sample library (SEQQSAMP) on page 489.

Communicating with IBM Z Workload Scheduler

To establish communication with IBM Z Workload Scheduler, your ATP must initialize and then allocate a conversation. The ATP must supply all information that is required to initialize the conversation; for example, the partner transaction program (TP) name and its LU, and a user ID and password that is used for security checking. Supply TP name `EQQAPI` to communicate with IBM Z Workload Scheduler. For GET, PUT, and DEL requests, the LU that the ATP sends requests to (the target LU) must be owned by the Z controller. For CREATE requests, if the target LU is not owned by an IBM Z Workload Scheduler address space where an event writer task is started, the ATP must send requests so that the events are broadcast on the target z/OS system. Broadcasting events on page 137 describes how you broadcast events on the target system.

When communication is established, your ATP sends a request to IBM Z Workload Scheduler in a *send buffer*. IBM Z Workload Scheduler responds by issuing a receive, inviting more requests from your ATP while it is processing the request. When you have completed your requests, you should issue several receive requests to ensure all data is received by the ATP. In cases where the receive type is Receive_Immediate, or if the buffers are large, data is returned in *packets*.

When the request has been processed, IBM Z Workload Scheduler builds a buffer that is sent to your ATP the next time that the ATP issues a receive request. This buffer is called a *receive buffer*.

If there is more than one active request from your ATP at a given time, you can identify each request by setting the token field (APPTOKEN in the APP section) to a unique value. The value could be, for example, a time stamp.

You can continue to make requests while the conversation is established. When you want to end the conversation, your ATP must issue a deallocate verb.

Note: The data that you send to IBM Z Workload Scheduler must be in EBCDIC format. IBM Z Workload Scheduler returns the data in the same format. If you use ASCII code, ensure that your data is converted to EBCDIC before a request is sent to IBM Z Workload Scheduler, and converted to ASCII when data is received by the ATP. Also, binary values might have to be swapped because the order of the byte representation (high-low, low-high) is machine dependent.

The following publications contain detailed information about writing an application program in the APPC environment:

APPC and CPI-C Implementations
APPC Programming Considerations
APPC Application Examples

CPI-C support provided by IBM Z Workload Scheduler

Your ATPs can issue requests to IBM Z Workload Scheduler through the API using CPI-C. Although your programs can use any CPI-C verbs, you should consider this information before you write your programs. It describes how the partner TP, IBM Z Workload Scheduler, responds to certain verbs:

CMACCP

Accept_Conversation

CMACCP is not applicable because the ATP must initialize and allocate the conversation.

CMALLC

Allocate

CMALLC must be issued by the ATP to allocate the conversation.

CMCFMD

Confirmed

CMCFMD is returned by IBM Z Workload Scheduler when a confirm verb is issued by the ATP. But IBM Z Workload Scheduler does not perform additional processing for a confirm request. The confirmed verb is issued when the request is received.

CMINIT

Initialize_Conversation

CMINIT must be issued by the ATP to initialize the conversation.

CMRCV

Receive

The ATP should repeat CMRCV calls to ensure that it receives the requested data. This is because when IBM Z Workload Scheduler receives the send state from the ATP and has no data to send at that time, it issues a receive inviting the ATP to send more requests. So the ATP determines the frequency of the polling.

CMSED

Set_Error_Direction

CMSED can be issued but is not used by IBM Z Workload Scheduler.

CMSERR

Send_Error

CMSERR can be issued but is not used by IBM Z Workload Scheduler.

CMSLD

Set_Log_Data

CMSLD can be issued but is not used by IBM Z Workload Scheduler.

CMSTPN

Set_TP_Name

Specify TP name EQQAPI, which is the default name. IBM Z Workload Scheduler recognizes these TP names:

EQQTRK

Supplied by trackers that communicate with the Z controller through APPC

EQQAPI

Supplied by user programs (ATPs) that communicate with IBM Z Workload Scheduler through the API.

API buffer layouts

There are two buffer types, send buffers and receive buffers. All buffers are in EBCDIC format and must be in contiguous storage. The buffers can contain these sections:

APP

Fixed section

APPFLD

Field section

APPDAT

Data section

APPOBJ

Object section

APPSEL

Selection section

APPVAL

Selection value section

The sections that a send buffer should contain depends on the request that you make. Table 112: Contents of a Send Buffer on page 125 shows the sections that you can include for each request:

Table 112. Contents of a Send Buffer

	Buffer sections[1]					
Request	**APP**	**APPOBJ**	**APPSEL**	**APPVAL**	**APPFLD**	**APPDAT**
GET	Required	Optional	Optional	Optional	Optional	Not used
PUT	Required	Required	Required	Required	Required	Required
DEL	Required	Required	Required	Required	Not used	Not used
CREATE	Required	Required	Required	Required	Required[2]	Required[2]

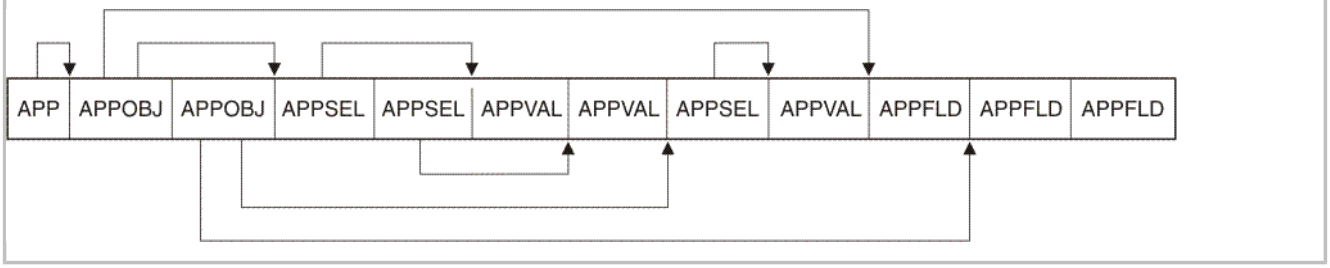
Note:

1. APP must be the first section in a buffer. There is no restriction on the order of other section types.
2. Not used for BACKUP_EVENT object.

Figure 5: Example of a send buffer layout for a GET request on page 125 is an example of the layout of a send buffer for a GET request. The arrows show the buffer parts that each section type points to. APP and APPOBJ point to related sections using triplet fields, which specify the offset, the length, and the number of the section type. APPSEL uses offset and length fields to point to an APPVAL section. All offsets are relative to the start of the buffer (offset 0).

Example

Figure 5. Example of a send buffer layout for a GET request



When a receive buffer is returned from IBM Z Workload Scheduler, the buffer contains the entire send buffer. Some fields are updated by IBM Z Workload Scheduler, for example, return codes and reason codes. For a GET request, data sections are also added if the requested information was found. One data section is added for each object instance found, and the data section triplet in APPOBJ is updated to point to the data.

If an error occurs during verification of the send buffer, IBM Z Workload Scheduler returns a receive buffer that contains the whole of the send buffer unaltered, plus an additional APP section at the start of the buffer. This additional APP section is updated to indicate the error type.

Each buffer section is described here in more detail.

APP - Fixed section

The buffer that your program passes to IBM Z Workload Scheduler must contain a fixed section, and it must be the first section in the buffer. It identifies the buffer, its size, the default request type, and points to object sections. The buffer must contain only 1 fixed section, even if multiple requests are passed in the same buffer.

The fixed section has this format:

Table 113. App-Fixed Section

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	80	APP	APPC BUFFER MAPPING
0	(0)	CHARACTER	4	APPDESC	BLOCK DESCRIPTOR (APP)
4	(4)	CHARACTER	2	APPVER	VERSION NUMBER (02)
6	(6)	BITSTRING	2	*	RESERVED
8	(8)	CHARACTER	3	APPTYPE	EYE CATCHER (DIA)
11	(B)	BITSTRING	1	APPFLAGS	RESERVED
12	(C)	SIGNED	4	APPTOTSZ	TOTAL SIZE
16	(10)	CHARACTER	8	APP_TYPE	DIALOG DATA TYPE (GET|PUT| DEL|CREATE)
24	(18)	SIGNED	4	APP_RETCODE	*RETURN CODE
28	(1C)	SIGNED	4	APP_RSNCODE	*REASON CODE
32	(20)		12	APP_OBJ_TRIPLET	OBJECT SECTION TRIPLET
32	(20)	SIGNED	4	APP_OBJ_OFF	OFFSET TO FIRST OBJECT SECTION
36	(24)	SIGNED	4	APP_OBJ_LEN	LENGTH OF AN OBJECT SECTION
40	(28)	SIGNED	4	APP_OBJ_NBR	NUMBER OF OBJECT SECTIONS
44	(2C)	SIGNED	4	APP_ERR_OFF	*OFFSET TO VERIFICATION ERROR
48	(30)	CHARACTER	8	*	RESERVED
56	(38)	CHARACTER	16	APPTOKEN	*TOKEN FIELD

Table 113. App-Fixed Section (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
72	(48)	CHARACTER	8	*	RESERVED
80	(50)	CHARACTER	8	APP_USERID	USER ID WHOSE AUTHORIZATION IS CHECKED TO RUN SRSTAT BY THE EQQUSIN SUBROUTINE
Note: Descriptions prefixed with an asterisk (*) indicate fields that IBM Z Workload Scheduler updates.					

In the fixed section:

APPDESC

Is the block descriptor and has the value APP.

APPVER

Is the version number and has the value 02.

Note: You can continue to use existing buffers with version number 01, but you cannot include new requests or fields in these buffers.

*****

Offset 6 (X'6'). Set this reserved field to binary zeros (X'00')

APPTYPE

Is the eye catcher and has the value DIA.

APPFLAGS

Set this reserved field to binary zeros (X'00').

APPTOTSZ

Is the total size of the buffer.

APP_TYPE

Is the request type that is the default for all requests. It is used if you do not provide a value for APPOBJ_TYPE in an object section of the buffer. If you set this field to blanks (X'40'), you must specify a request in each object section of the buffer.

APP_OBJ_TRIPLET

Contains the offset to the first APPOBJ section, the length of all sections, and the number of sections. If the APP_OBJ_NBR field contains binary zeros (X'00') for a GET request, IBM Z Workload Scheduler returns a *data*

dictionary. The data dictionary is a description of all objects and all fields that the API supports for a GET request. CREATE objects are not described.

APP_RETCODE

Is the return code that is set by IBM Z Workload Scheduler. In the send buffer, set this field to binary zeros (X'00'). For more information, see Return codes and reason codes generated by IBM Z Workload Scheduler on page 137.

APP_RSNCODE

Is the reason code that is set by IBM Z Workload Scheduler. In the send buffer, set this field to binary zeros (X'00'). For more information, see Return codes and reason codes generated by IBM Z Workload Scheduler on page 137.

APP_ERR_OFF

Is set by IBM Z Workload Scheduler when APP_RSNCODE indicates an error that has an offset associated with it. It is the offset in the buffer where a verification error was found. In the send buffer, set this field to binary zeros (X'00').

*****

Offset 48 (X'30'). Set this reserved field to binary zeros (X'00').

APPTOKEN

Is a value that your program can set to uniquely identify a buffer. It could be, for example, a time stamp. APPTOKEN can be useful if there is more than one active request from your ATP at a time.

APP_USERID

This value specifies the user ID whose authorization is checked by the EQQUSIN subroutine when the SRSTAT command is to be run. If not used, this field must set to blanks (X'40').

APPOBJ - Object section

This section identifies the object and optionally the request type. The buffer must contain an object section for all requests except a GET request. A buffer can contain more than one object section, but all object sections must be in contiguous storage; that is, they must follow one another. The part of the buffer containing object sections is pointed to by the APP_OBJ_TRIPLET in the fixed section. APPOBJ itself points to APPSEL, APPFLD, and APPDAT sections if they are specified in a send buffer.

If your send buffer does not contain an object section for a GET request, that is, it contains only the fixed section, the buffer that IBM Z Workload Scheduler returns contains a description of all objects and all fields that are supported by the API for a GET request.

The object section has this format:

Table 114. APPOBJ-Object Section

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	84	APPOBJ	OBJECT SECTION APPOBJ_PTR = ADDR(APP) + APP_OBJ_OFF
0	(0)		24	APPOBJ_ID	OBJECT IDENTIFIER
0	(0)	CHARACTER	16	APPOBJ_NAME	OBJECT NAME
16	(10)	CHARACTER	8	APPOBJ_KEY_TYPE	KEY TYPE
24	(18)		12	APPOBJ_FLD_TRIPLET	FIELD SECTION TRIPLET
24	(18)	SIGNED	4	APPOBJ_FLD_OFF	OFFSET TO FIRST FIELD SECTION
28	(1C)	SIGNED	4	APPOBJ_FLD_LEN	LENGTH OF A FIELD SECTION
32	(20)	SIGNED	4	APPOBJ_FLD_NBR	NUMBER OF FIELD SECTIONS
36	(24)		12	APPOBJ_SEL_TRIPLET	SELECTION SECTION TRIPLET
36	(24)	SIGNED	4	APPOBJ_SEL_OFF	OFFSET TO FIRST SELECTION SECTION
40	(28)	SIGNED	4	APPOBJ_SEL_LEN	LENGTH OF A SELECTION SECTION
44	(2C)	SIGNED	4	APPOBJ_SEL_NBR	NUMBER OF SELECTION SECTIONS
48	(30)		12	APPOBJ_DAT_TRIPLET	DATA SECTION TRIPLET
48	(30)	SIGNED	4	APPOBJ_DAT_OFF	OFFSET TO FIRST DATA SECTION
52	(34)	SIGNED	4	APPOBJ_DAT_LEN	LENGTH OF ALL DATA SECTIONS
56	(38)	SIGNED	4	APPOBJ_DAT_NBR	NUMBER OF DATA SECTIONS
60	(3C)	CHARACTER	8	APPOBJ_TYPE	DIALOG DATA TYPE (GET|PUT|DEL|CREATE)
68	(44)	SIGNED	4	APPOBJ_RET	*OBJECT LEVEL RETURN CODE
72	(48)	SIGNED	4	APPOBJ_RSN	*OBJECT LEVEL REASON CODE
76	(4C)	CHARACTER	8	APPOBJ_AUTH	*RACF AUTHORITY (READ or UPDATE)

Table 114. APPOBJ-Object Section (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
Note: Descriptions prefixed with an asterisk (*) indicate fields that IBM Z Workload Scheduler updates.					

In the object section:

APPOBJ_NAME

Identifies the object type. For a description of valid names, see Specifying object names on page 134.

APPOBJ_KEY_TYPE

Is the key type. If you set this field to blanks (X'40'), a default value is used. for a description of valid key types, see Specifying key types on page 135 .

APPOBJ_FLD_TRIPLET

Contains the offset to the first APPFLD section, the length of each section, and the number of sections. If the APPOBJ_FLD_NBR field contains all binary zeros (X'00') for a GET request, IBM Z Workload Scheduler returns all fields in the selected object instances.

APPOBJ_SEL_TRIPLET

Contains the offset to the first APPSEL section, the length of each section, and the number of sections. Set these fields to binary zeros (X'00') if there are no APPSEL sections.

APPOBJ_DAT_TRIPLET

Contains the offset to the first APPDAT section, the length of all sections, and the number of sections. Set these fields to binary zeros (X'00') if there are no APPDAT sections. IBM Z Workload Scheduler updates these fields if data is returned for a GET request.

APPOBJ_TYPE

Is the request type for this object. If you set this field to blanks (X'40'), APP_TYPE determines the request type.

APPOBJ_RET

Is the object level return code that is set by IBM Z Workload Scheduler. In the send buffer set this field to binary zeros (X'00'). For more information, see Return codes and reason codes generated by IBM Z Workload Scheduler on page 137.

APPOBJ_RSN

Is the object level reason code that is set by IBM Z Workload Scheduler. In the send buffer set this field to binary zeros (X'00'). For more information, see Return codes and reason codes generated by IBM Z Workload Scheduler on page 137.

APPOBJ_AUTH

Is the access authority (read or update) that your ATP has to the specified object. For GET, PUT, and DEL requests, IBM Z Workload Scheduler updates this field before the buffer is returned. It is not updated for a CREATE request. You could use APPOBJ_AUTH to establish your access by issuing a GET request for the object, before attempting further read or update requests. In the send buffer set this field to blanks (X'40').

APPSEL - Selection section

This section identifies a particular field in an object. By specifying a field name and a comparison operator in APPSEL, you can limit the instances of the object that IBM Z Workload Scheduler finds. APPSEL is pointed to by the APPOBJ_SEL_TRIPLET in its object section and must itself point to an APPVAL section where a selection value is specified. To identify one particular instance of an object, you might need to specify more than one APPSEL in the send buffer. The selection sections for a particular APPOBJ must be in contiguous storage.

If you do not specify APPSEL for a GET request, IBM Z Workload Scheduler returns all instances of the object.

The selection section has this format:

Table 115. APPSEL-Selection Section

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	36	APPSEL	SELECTION SECTION ADDRESS OF FIRST SELECTION SECTION FOR THIS OBJECT: APPSEL_PTR = ADDR(APP) + APPOBJ_SEL_OFF
0	(0)	CHARACTER	16	APPSEL_NAME	OBJECT FIELD NAME
16	(10)	CHARACTER	2	APPSEL_OPER	OPERATOR
18	(12)	CHARACTER	10	*	RESERVED
28	(1C)	SIGNED	4	APPSEL_VALUE_OFF	VALUE OFFSET
32	(20)	SIGNED	4	APPSEL_VALUE_LEN	VALUE LENGTH

In the selection section:

APPSEL_NAME

Is a field name in the object.

APPSEL_OPER

Is a comparison operator.

*

Offset 18 (X'12'). Set this reserved field to binary zeros (X'00').

APPSEL_VALUE_OFF

Is the offset to the APPVAL section.

APPSEL_VALUE_LEN

Is the length of the APPVAL section.

For more information, see Selecting object instances on page 135. Field names are described in API object fields on page 466.

APPVAL - Selection value section

This section contains a value that you want IBM Z Workload Scheduler to search for within the object, according to the selection criteria that you specified in APPSEL. APPVAL is pointed to by APPSEL; it must be included if APPSEL is specified in the buffer. One APPVAL is required for each APPSEL. Selection value sections need not be in contiguous storage.

Each APPVAL section can contain only one value. If you specify GN (generic compare) in the APPSEL_OPER field, the selection value can contain the generic search arguments asterisk (*) and percent (%). An asterisk represents a character string or a null string. The percent sign represents a single character. For a complete description of generic search argument, see *Managing the Workload*.

The selection value section has this format:

Table 116. APPVAL-Selection Value Section

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	*	APPVAL	DATA SECTION ADDRESS OF FIRST DATA SECTION FOR THIS OBJECT: APPVAL_PTR=ADDR(APP) + APPSEL_VALUE_OFF
0	(0)	(See note)	*	APPVAL_DAT	DATA

Note: The field type depends on the object field name that you specify in APPSEL_NAME. See API object fields on page 466.

APPFLD - Field section

For PUT and CREATE requests, each field section identifies a field in the selected object that you want to update; for example, the status of an operation in the current plan. APPFLD is not used for a CREATE request when the object name is BACKUP_EVENT, or for DEL requests.

For the GET request, you can use APPFLD sections to limit the data that is returned to particular object fields. You need supply only the APPFLD_NAME in a send buffer. IBM Z Workload Scheduler updates the APPFLD_LEN and APPFLD_TYPE fields before the buffer is returned. If you do not specify APPFLD for a GET request, the buffer returned contains all fields in the selected instances of the object.

Field sections are pointed to by the APPOBJ_FLD_TRIPLET in the object section. You can specify more than one APPFLD for each APPOBJ, but all field sections for a particular APPOBJ must be in contiguous storage.

The field section has this format:

Table 117. APPFLD-Field Section

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	24	APPFLD	FIELD SECTION ADDRESS OF FIRST FIELD SECTION FOR THIS OBJECT: APPFLD_PTR= ADDR(APP) + APPOBJ_FLD_OFF
0	(0)	CHARACTER	16	APPFLD_NAME	FIELD NAME
16	(10)	SIGNED	4	APPFLD_LEN	FIELD LENGTH
20	(14)	CHARACTER	4	APPFLD_TYPE	*FIELD DATA TYPE
Note: Descriptions prefixed with an asterisk (*) indicate fields that IBM Z Workload Scheduler updates.					

In the field section:

APPFLD_NAME

The name of the field. For a description of the fields that you can specify for each object type, see Selecting object fields to update or retrieve on page 137.

APPFLD_LEN

The length of the field and is used in identifying the value in APPDAT for this field. For a GET request, or when the object is BACKUP_EVENT, set this field to binary zeros (X'00').

APPFLD_TYPE

The data type and is updated by IBM Z Workload Scheduler before the buffer is returned. Set this field to blanks (X'40') in a send buffer.

APPDAT - Data section

For PUT and CREATE requests, APPDAT contains the new values for the fields identified in the APPFLD sections. Only one APPDAT must be specified for each APPOBJ. The values must be in the same order as the corresponding APPFLD sections.

For a GET request, data sections are found only in a receive buffer. IBM Z Workload Scheduler returns in the receive buffer one data section for each instance of the object. Each APPOBJ section in the send buffer is updated by IBM Z Workload Scheduler to point to associated data sections when the receive buffer is returned. The data sections are always the last sections in the receive buffer, and are returned in contiguous storage by object.

APPDAT is not used for DEL requests.

The data section has this format:

Table 118. APPDAT-Data Section

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	*	APPDAT	DATA SECTION ADDRESS OF FIRST DATA SECTION FOR THIS OBJECT: APPDAT_PTR=ADDR(APP) + APPOBJ_DAT_OFF
0	(0)	(See note)	*	APPDAT_DAT	DATA

Note: The field type depends on the object field name that you specify in APPFLD_NAME or that IBM Z Workload Scheduler retrieves. See API object fields on page 466.

Specifying object names

You identify the object type by specifying a name in the APPOBJ_NAME field of the object section. Table 119: API Object Names on page 134 describes the object names that you can specify:

Table 119. API Object Names

Object	Valid requests	Description
CP_STATUS	GET	Current plan status
CP_OPERATION	GET, PUT, DEL	Current plan operation
CP_RESOURCE	GET	Current plan operation special resource
CP_WORK_STATION	GET	Current plan workstation (common part)
CP_OPEN_INTERVAL	GET	Current plan workstation open interval
CP_OPER_EVENT	CREATE	Current plan operation event.
CP_OPINFO_EVENT	CREATE	Current plan operation user data event
CP_SR_EVENT	CREATE	Current plan special resource event
BACKUP_EVENT	CREATE	Backup event.

Table 119. API Object Names (continued)

Object	Valid requests	Description
CP_WS_EVENT	CREATE	Current plan workstation event.

Note:

1. You can add (PUT) an operation only to an existing application occurrence. You cannot add an occurrence through the API.
2. You cannot delete (DEL) an operation if it is the only operation in an occurrence. You cannot delete an occurrence through the API.

Selecting object instances

IBM Z Workload Scheduler uses these criteria to identify object instances:

Key type

Identifies the relationship between the instances of an object that are located using the selection criteria, and the object instances that you want IBM Z Workload Scheduler to find.

Selection field

The name of an object field that is used in locating instances of the object.

Selection value

The value in the selection field that is used in locating instances of the object.

Operator

A comparison operator that determines how the selection value is used in locating instances of the object.

Specifying key types

You can specify a key type in each object section of the buffer. If the key type field contains blanks (X'40'), a default is used. You can specify these key types:

SAME

The objects found are those matching the selection criteria. This value is valid for, and is the default for, these objects:

CP_OPERATION
CP_WORK_STATION
CP_STATUS
CP_OPER_EVENT
CP_OPINFO_EVENT

CP_SR_EVENT
BACKUP_EVENT
CP_WS_EVENT.

PRED

The objects found are those that are predecessors to the object matching the selection criteria. This value is valid for the CP_OPERATION object but only with a GET request.

SUCC

The objects found are those that are successors to the object matching the selection criteria. This value is valid for the CP_OPERATION object but only with a GET request.

OWNER

The objects found are those whose owner matches the selection criteria. This value is valid for, and is the default for, the objects CP_OPEN_INTERVAL (owner is CP_WORK_STATION) and CP_RESOURCE (owner is CP_OPERATION).

Specifying selection criteria

You specify selection criteria in the APPSEL and APPVAL sections of the buffer to limit the instances of an object that are located by IBM Z Workload Scheduler. APPSEL contains a selection field name and a comparison operator that determines how the value for this field is used. You supply the field value in the APPVAL section. API object fields on page 466 describes the field names and field values of each object, and the selection type of each field. There are select fields:

Required

For a GET request with a key type of OWNER, PRED, or SUCC, you must specify these fields and the operator must be EQ to ensure that there is only one possible match. When the key type is SAME, these fields are optional.

For PUT and DEL requests, you must specify these fields. Also, the key type must be SAME and the operator EQ.

Optional

You can specify the field in the APPSEL section, but it is not required.

Not supported

You must not specify the field in the APPSEL section.

You can specify these operators in the APPSEL section:

Table 120. Operators That You Can Specify in the APPSEL Section

Operator	Description
EQ or =	Equal to
NE or ^=	Not equal to

Table 120. Operators That You Can Specify in the APPSEL Section (continued)

Operator	Description
GT or >	Greater than
LT or <	Less than
GE or >=	Greater than or equal to
LE or <=	Less than or equal to
GN	Generic compare

Broadcasting events

The LU that your ATP sends requests to is owned by an IBM Z Workload Scheduler address space where the APPC subtask is started. When you send a CREATE request to the LU, the address space processes the request and creates an event. If you want to report an event to more than one IBM Z Workload Scheduler address space, or an event writer is not started in the address space that owns the target LU, you must broadcast the event.

To broadcast an event, specify `SUBSYSTEM_NAME` in the APPSEL section but do not provide a name in the APPVAL section, or provide the name `MSTR` in APPVAL. The event is sent using the subsystem interface (SSI) to all IBM Z Workload Scheduler address spaces started on the same z/OS image as the target.

Selecting object fields to update or retrieve

You select object fields to update or retrieve by specifying values in the APPFLD and APPDAT sections of a buffer.

The APPFLD section identifies an object field. For GET requests, APPFLD identifies a field in each located object instance that you want IBM Z Workload Scheduler to return in the receive buffer. For PUT and CREATE requests, APPFLD identifies the field that you want to update. APPFLD is not used for DEL requests or when the object for a CREATE request is BACKUP_EVENT.

For a GET request, the APPDAT section is not used in a send buffer. APPDAT sections are returned in a receive buffer if data is found. For PUT and CREATE requests, APPDAT contains the new values for the fields identified in APPFLD sections. You must specify only 1 APPDAT per APPOBJ.

For a description of the fields that you can update or retrieve, see API object fields on page 466.

Return codes and reason codes generated by IBM Z Workload Scheduler

If a request through the API causes a severe error in an IBM Z Workload Scheduler subtask, you receive one of these CPI-C return codes:

CM_PROGRAM_ERROR_NO_TRUNC
CM_PROGRAM_ERROR_PURGING.

The conversation is deallocated, and CPI-C return code CM_RESOURCE_FAILURE_NO_RETRY is set. Here, do not resend the buffer to IBM Z Workload Scheduler until problem determination establishes a reason for the previous error. For information about CPI-C return codes, refer to *CPI-C Communications Reference*.

Besides CPI-C return codes, IBM Z Workload Scheduler can generate return codes and reason codes for the various requests that are made. Your program can test the results of the call to IBM Z Workload Scheduler by inspecting return codes and reason codes in the APP and APPOBJ sections of the buffer.

Return codes and reason codes generated in the fixed section (APP)

A buffer always starts with a fixed section. Return codes and reason codes are generated in the fixed section when IBM Z Workload Scheduler validates the buffer. The APP_RETCODE field can contain one of these codes:

0

Execution successful.

4

Execution successful but no data was returned. Either there was no data that matched the GET request, or the ATP is not authorized to access the data matching the GET request.

12

Execution unsuccessful; the buffer is invalid. IBM Z Workload Scheduler has not attempted to process the request. A receive buffer is created that contains an APP control block followed by the entire send buffer. No updates are made to any fields in the send buffer. So this special receive buffer will start with 2 APP sections.

The APP_RSNCODE field can contain one of these codes:

0

Execution successful.

4

Buffer shorter than APP.

8

Eye catcher in APPDESC field is invalid. It must be APP.

12

Version number in APPVER field is invalid. It must be 02.

16

Type in APPTYPE field is invalid. It must be DIA.

20

APPTOTSZ invalid.

24

Data type invalid. Specify GET, PUT, DEL, or CREATE.

28

Object section not within buffer.

32

Object section overlays APP.

36

Selection section not within buffer.

40

Selection section overlays APP or object section.

44

Field section not within buffer.

48

Field section overlays APP or object section.

52

Required field not supplied.

56

Invalid object name in OBJ section.

60

Invalid field name in FLD section.

64

Invalid field name in SEL section.

68

APPTOKEN value invalid (duplicate).

Return codes and reason codes generated in the object section (APPOBJ)

The return codes and reason codes generated in the object section indicate an error after IBM Z Workload Scheduler validated the buffer. No return and reason codes are generated in the object section for CREATE requests. For GET, PUT, and DEL requests, the APPOBJ_RET field can contain one of these codes:

0

Execution successful.

12

Execution unsuccessful.

The APPOBJ_RSN field can contain one of these codes:

0

Execution successful.

4

The operation does not exist.

8

An invalid update was attempted.

12

A security violation occurred.

16

An error was detected. For more information, check the message log (EQQMLOG) of the IBM Z Workload Scheduler address space that the request was sent to.

Security

The access to IBM Z Workload Scheduler can be controlled through security mechanisms provided by:

- APPC and RACF®
- IBM Z Workload Scheduler and RACF®.

APPC and RACF®

The APPC security mechanism provides access control in these areas:

- Access to logical units (LUs)
- Access control for LU to LU communication
- Access to transaction programs
- Security within the network.

IBM Z Workload Scheduler recognizes these TP names:

EQQTRK

Supplied by trackers that communicate with the Z controller through APPC

EQQAPI

Supplied by user programs (ATPs) that communicate with IBM Z Workload Scheduler through the API.

For a detailed description of how to protect your APPC environment, see *APPC Management*.

For a detailed description of how to protect information that crosses the network, see *ICSF/MVS™ Programmer's Guide*.

IBM Z Workload Scheduler and RACF®

IBM Z Workload Scheduler performs security checking at the Z controller for GET, PUT, and DEL requests, for all ATPs that use the API. To establish a conversation, your ATP must supply a user ID and password, and optionally a profile that indicates the RACF® user group. The user ID must have the required level of access.

For CREATE requests, IBM Z Workload Scheduler does not perform security checking, because the request could be directed to more than one IBM Z Workload Scheduler subsystem where security rules differ. You can prevent unauthorized use of CREATE requests through APPC security mechanisms by protecting the LU and the TP name.

You can protect access to IBM Z Workload Scheduler resources at these levels:

1. The IBM Z Workload Scheduler subsystem resource
2. Fixed resources
3. Subresources.

Access at one level determines the default access to the next level. The default is used if the required resource is not protected at the following level. To use the API, you must have at least read access to the IBM Z Workload Scheduler subsystem, which is defined in the APPL class. GET, PUT, and DEL requests require this access to fixed resources:

GET

CP read. SR read is also required to retrieve special resource information.

PUT

CP update is required for CP_OPER_EVENT, CP_OPINFO_EVENT, and CP_WS_EVENT. Additionally, EXEC update is required to request the EXEC command. BKP update is required for BACKUP_EVENT.

DEL

Requires the same access as PUT.

You can further restrict access by specifying subresources, which are described in Table 121: Subresource Protection for Requests through the API on page 141.

Table 121. Subresource Protection for Requests through the API

Fixed resource	Subresource	Description
CP	CP.ADNAME	Application name
	CP.GROUP	Application authority group ID
	CP.JOBNAME	Operation job name
	CP.OWNER	Application owner
	CP.WSNAME	Workstation name
	CP.ZWSOPER	Workstation name used by an operation
	CP.CPGDDEF	Group definition ID name

Table 121. Subresource Protection for Requests through the API (continued)

Fixed resource	Subresource	Description
RL	RL.ADNAME	Occurrence name
	RL.OWNER	Occurrence owner ID
	RL.GROUP	Occurrence authority-group ID
	RL.WSNAME	Current-plan workstation name
SR	SR.SRNAME	Special resource name

Note: If you restrict access at the subresource level, selection criteria will find only those instances of an object that both match the selection criteria and that the user ID has access to.

If a request is denied for READ access to the IBM Z Workload Scheduler subsystem resource or to a fixed resource, you receive CPI-C return code CM_SECURITY_NOT_VALID and the conversation is deallocated. Other security failures result in an error buffer with reason code 512 and the conversation remains allocated.

For a detailed explanation of security considerations, see *Customization and Tuning*.

Part II. Programming tools

Chapter 3. Batch command interface tool

About this task

Using IBM Z Workload Scheduler you can control and automatically plan your production workload in your complex. You can use the program interface to issue various types of requests to the IBM Z Workload Scheduler subsystem. The program interface supports different requests to read and update resources in IBM Z Workload Scheduler databases. The resources can be:

- Operations and their dependencies
- Applications
- Operator instructions
- Calendars
- Periods
- Run cycle groups
- Workstations
- The current and long-term plans

Online tools

About this task

To update records in IBM Z Workload Scheduler databases, you can use ISPF dialogs. Online administration is performed on the controlling system (the dialog is available on the system in your configuration that is running the Z controller).

The batch command interface

About this task

The Batch Command Interface is a sample program that you can run in batch and that issues various types of request to the IBM Z Workload Scheduler subsystem.

Input to batch command interface

About this task

It is a PIF application. Note that dates provided in the INPUT must be in real format. If the PIFCWB and PIFHD parameters of the INTFOPTS initialization statement of the Z controller do not follow this rule, the EQQYPARM DD card invoking the BCIT must point to a member where INIT specifies the following:

- CWBASE (00)
- HIGHDATE (711231)

When the Batch Command Interface is called, it invokes PGM=EQQYCAIN, with an optional PARM field (the sample member EQQYCBAT contains an example of how this program can be invoked). The positional parameters that can be passed with the PARM field are as follows:

Table 122. Positional parameters that can be passed with the Batch Command Interface

Description	Maximum length	Default	Values
SUBSYSTEM NAME	4	OPCA	product name
WTO desired	6	MSGON	MSGOFF, MSGNONE
IA date	6	current date	CPSTDA/ yymmdd
IA time	4	current time	hhmm
CP needed	6	MUSTCP	MISSCP

EXAMPLES:

```
PARM='OPGQ, MSGOFF, CPSTDA, 1400'
```

```
PARM='OPGQ,,991207'
```

```
PARM='OPGQ,,,,MISSCP'
```

The meaning of the keywords is the following:

CPSTDA

Means that the default IA date is the starting date of the CP.

MISSCP

Means that the requested BCIT functions do not require the existence of a CP and that BCIT should not try to access it.

MSGNONE

Means that WTOs (including those issued in case of errors or for commands that do not have performance impacts) are not to be issued any longer.

MSGOFF

Means that WTOs (except those issued in case of errors or for commands that do not have performance impacts) are not to be issued any longer.

MSGON

Means that WTOs are to be issued in any case. It is strongly recommended not to use the default MSGON when the BCIT commands invoked might produce too many WTOs.

MUSTCP

Requests that BCIT accesses the CP, no matter if the requested functions need the existence of CP.

A set of actions, called a program, is specified in a file SYSIN referenced by the DD card: //SYSIN DD. If the SYSIN DD card refers to a DASD file, ensure that the file is defined with RECFM=FB and LRECL=80.

A program with instructions is input for this sample.

An instruction is an action on a resource with any arguments to identify it and other arguments to process it.

An argument has two parts; the right part is its identification and the left part is its value. These two parts are separated by the equal sign (=). An argument is finished by a comma (,) or semicolon (;) if this argument is the last of the instruction, and by a period if it is the last of the program. The blanks before the identifier are ignored but the blanks after the identifier are used. A blank is a character, not a delimiter, if it is before an argument or a value.

An instruction must be finished by a semicolon(;). Only the last instruction of the program is finished by a period (.). A program must be finished by a period (.).

The descriptive fields, such as DESC and EXTNAME, can include special characters only if the field is inserted within single quotation marks. For example,

```
  DESC='XXXX.AAAAA,BBBBB;'
```

Note: If you need to use delimiters, such as single quotation marks, in descriptive fields follow these rules:

- To use the single quotation mark within a field, you must use two quotation marks.
- To use the single quotation mark at the beginning or at the end of a field, you must use three quotation marks.

Example of program:

```
ACTION=OPTIONS,BL=N,LTP=N;
ACTION=LIST,RESOURCE=ADCOM,ADID=description,IA=9202121500;
ACTION=LIST,RESOURCE=ADCOM,ADID=A%%%B*,IA= 9202121500,
    PRIORITY=5;
ACTION=SELECT,
       RESOURCE=OICOM,
       ADID=ADABASE;
ACTION=OPTIONS,BL=Y;
ACTION=LIST,RESOURCE=LTOCCOM,
       ADID=ADABASE;
ACTION=COPY,RESOURCE=AD,ADID=ADABASE,STATUS=A,VALTO=991231,
 NADID=MATENCIOP,NSTATUS=P;
ACTION=INSERT,RESOURCE=LTPRE,ADID=MATENCIOP,IAD=920215,IAT=0915,
    PREADID=ADABASEOP,PREIAD=920215,PREIAT=0915;
ACTION=LIST,RESOURCE=LTOCCOM,ADID=MATENCIOP;
ACTION=INSERT,RESOURCE=LTOC,ADID=MATENCIOP,IAD=920215,IAT=0915,
    DEADLINE=9202150916;
ACTION=DELETE,RESOURCE=CPPRE,ADID=MATENCIOP,IA=9202062343,OPNO=30,
    PREADID=ADABASE,PREOPNO=10,PREIA=9201310700;
ACTION=LIST,RESOURCE=CPOPCOM,ADID=MATENCIOP,OPNO=30;
ACTION=MODIFY,RESOURCE=CPOP,ADID=MATENCIOP,IA=9202062343,OPNO=30,
  JOBNAME=MATENCIO,DESC='TEST MODIFY CPOP',EDUR=0100,PSUSE=1,R1USE=1,
  R2USE=1,JCLASS=B,OPIA=9202062344,OPDL=9202062345,WSNAME=WSTC,
  STATUS=A;
ACTION=MODIFY,RESOURCE=CPEXT,ADID=MATENCIOP,IA=9202062343,OPNO=30,
  EXTNAME='Operation Extended Name';
ACTION=MODIFY,RESOURCE=CPOC,ADID=MATENCIOP,IA=9202062300,
```

```
    IANEW=9202062343,DEADLINE=9202062345;
ACTION=INSERT,RESOURCE=CPPRE,ADID=MATENCIOP,IA=9202042342,OPNO=25,
    PREADID=ADABASE,PREOPNO=10,PREIA=9201310700;
ACTION=DELETE,RESOURCE=LTOC,ADID=MATENCIOP,IAD=920129,IAT=0700;
ACTION=IMPORT,RESOURCE=AD,ADID=MATENCIOP;
ACTION=INSERT,RESOURCE=AD,ADID=MATENCIOP;
ACTION=DELETE,RESOURCE=AD,ADID=MATENCIOP;
ACTION=SELECT,RESOURCE=AD,ADID=MATENCIOP;
ACTION=COPY,RESOURCE=OI,ADID=ADABASE,OPNO=10,
 NADID=MATENCIOP,NOPNO=10;
ACTION=INSERT,RESOURCE=CPOC,ADID=MATENCIOP,IA=9201291500;
ACTION=INSERT,RESOURCE=CPSR,ADID=MATENCIOP,IA=9201291500,RESNAME=name,
   RESUSAGE=X,OPNO=10,QUANTITY=1;
ACTION=INSERT,RESOURCE=CPOP,ADID=MATENCIOP,IA=9202042342,OPNO=25,
JOBNAME=MATENCIO,WSNAME=FTW1,EXTNAME='Operation Extended Name'.
```

DD CARD EQQMLIB contains the scheduler messages.

Note: In the above example, for simplicity many commands are included in the same SYSIN, but this is not a recommended practice. Because BCIT processes all the statements contained in the SYSIN and does not stop processing if any statement fails (unless ERROR=Y is specified in the OPTIONS card), it is recommended to avoid inserting commands which should not be run in case of failure in one of the previous commands, or to specify ERROR=Y in the OPTIONS card.

BCIT output

About this task

Each instruction is processed and a return code is displayed in the LOG. Depending on the action and options, the outputs are different, and are written in different files. The different output files are:

AD

If arguments BL=Y and BLPRT=N are coded in the action OPTIONS, the identifier, status, and VALTO of each listed application (AD) will be written in the file referenced by the AD DD card (LRECL=23).

BATCHL

If arguments BL=Y and BLPRT=Y are coded in the action, LIST and SELECT of ADs and OI results will be formatted as BATCH LOADER size and written in referenced files by the DD BATCHL card.

CPCOND

If argument BL=Y is coded in the OPTIONS action, the ACTION=LIST,RESOURCE=CPCONDCO,ADID=XXXX* result is written in the file referenced by CPCOND card (LRECL=80). It is the same for action=select.

CPOC

If argument BL=Y is coded in the OPTIONS action, the ACTION=LIST,RESOURCE=CPOC,ADID=XXXX* result is written in the file referenced by CPOC card (LRECL=80). It is the same for action=select.

CPOP

If argument BL=Y is coded in the OPTIONS action, the ACTION=LIST,RESOURCE=CPOPCOM,ADID=XXXX* result is written in the file referenced by CPOP card (LRECL=80). It is the same for action=select.

DATAFI

File that contains the output of LIST JCLVCOM

EQQDUMP

File that contains information to understand diagnostic data set error codes.

EQQMLOG

File that contains the scheduler messages for return code of 6 or higher.

EXPORTAD

If the action is an EXPORT and the resource is an AD, the file will contain structured AD segments as well, as it is easier to import them.

EXPORTOI

If the action is an EXPORT and the resource is an OI, the file will contain structured OI segments as well, as it is easier to import them.

EXPORTRG

If the action is an EXPORT and the resource is an RG, the file will contain structured RG segments as well, as it is easier to import them.

IMPORTAD

If the action is an IMPORT and the resource is an AD, the file will contain AD segments to be imported.

IMPORTOI

If the action is an IMPORT and the resource is an OI, the file will contain OI segments to be imported.

IMPORTRG

If the action is an IMPORT and the resource is an RG, the file will contain RG segments to be imported.

OI

If arguments BL=Y and BLPRT=N are coded in the action OPTIONS, the identifier and operation number of each listed operator instruction(OI) will be written in the file referenced by the OI DD card (LRECL=23).

RG

If arguments BL=Y and BLPRT=N are coded in the action OPTIONS, the identifier of each listed run cycle group (RG) will be written in the file referenced by the RG DD card (LRECL=23).

SYSPRINT

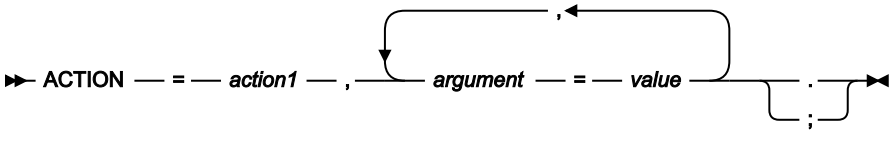
Required, it is used to print results of LIST and SELECT actions.

Instructions

About this task

There are two sorts of instructions:

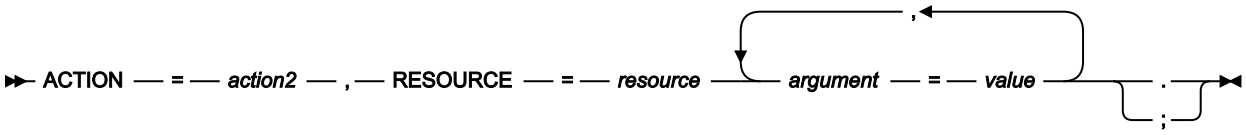
- The first does not refer to a resource and the action is still active during the whole program until the next instruction of the same type.



action1

OPTIONS

- The second refers to a resource and arguments that define this resource, and some active arguments. Some instructions are independent in their execution.



action2

LIST | SELECT | INSERT | DELETE | MODIFY | COPY | EXPORT | IMPORT | LISTSTAT

Return codes

The batch command interface returns a return code for each processed instruction, and the program keeps the highest return code.

Code

Explanation

0

All instructions are successfully processed.

2

End of file, a forgotten period at the end of program or empty file.

4

Resource not found, or not authorized, or the operation input arrival is earlier than the occurrence input arrival.

5

One or more of the related dependencies does not exist.

6

Refer back to instruction return code.

8

Refer back to instruction return code.

10

More than one occurrence matches the specified arguments.

32

Input arrival date less than CP end date (applies to DELETE LTOC, DELETE LTPRE, INSERT LTOC, and MODIFY LTOC).

100

The field of the parm operand, specified in the preceding WTO, is wrong.

105

Error during the load of the EQQYCTDI interface module.

110

Syntax error (see the following message).

120

A DD card is missing (see the log for more information about which card is missing).

130

No SYSIN DD card found.

140

Exceeded the word capacity (54 characters). For example, this might occur if in the last card of the SYSIN file there is a period and nothing else.

150

Options incompatible with the command.

160

None of the input records match the value specified in ADID.

170

Not enough storage available.

180

The file input to the import function has been produced with a level of code not compatible.

190

MATCHTYP not specified and VALTO or VALFROM specified.

191

VALFROM or VALTO specified and one (or more) permanent OI(s) found among those selected.

195

MATCHTYP specified in a DELETE AD or DELETE OI command, without VALFROM and VALTO.

200

Current plan does not exist.

210

The input arrival date and time are not valid. See the messages for more information.

300

BCIT received invalid data from the scheduler.

310

Erroneous status code.

400

ADVERS=Y and NADID omitted requires that at least one between NSTATUS and NVALFROM be specified with a new value.

401

ADVERS=Y and NADID specifying a name for which there are already existing versions requires that both NSTATUS and NVALFROM be specified and their combination must not exist.

402

ADVERS=N and NADID omitted does not allow new arguments different from NSTATUS to be specified.

403

ADVERS=N and NADID omitted requires NSTATUS to specify a value different from the status of the ADID being copied.

404

ADVERS=N and NADID specifying a name for which there are already existing versions is not supported.

405

COPY AD not allowed because maximum number of versions already exists.

406

NVALTO specified in COPY AD when ADVERS=Y is active.

COPY

The COPY instruction copies an AD, JCLV, OI, or RG, or any combination of AD, JCLV, OI, and RG to the same IBM Z Workload Scheduler subsystem with some argument change. This function quickly creates an AD, JCLV, OI, or RG from an extract

segment. It is useful if you want to modify some arguments, such as ADID, RGID, STATUS, IA. You can use the batch command interface to copy JCLV and change only the JCLV name.

COPY AD

Copy an application, changing arguments. New arguments begin with N.

```
ACTION=COPY,RESOURCE=AD,STATUS=ad_status,
                        ADID=application_description,
                        GROUP=authority_group_name,
                        NSTATUS=new_adstatus,
                        NADID=new_application_description,
                        NGROUP=new_authority_group_name,
                        NOWNER=new_owner,
                        NVALFROM=new_valid_from_date,
                        NVALTO=new_valid_to_date,
                        VALTO=valid_to_date;
```

adstatus

P or A (char(1))

application_description

Character (char(16))

authority_group_name

Character (char(8))

new_adstatus

P or A (char(1))

new_application_description

Character (char(16))

new_authority_group_name

Character (char(8))

new_owner

Character (char(16))

new_valid_from_date

Date(YYMMDD)

Note: The following rules apply when this command is run with the option ADVERS=Y active: NVALTO cannot be specified. If NADID is omitted, then NSTATUS must be specified and differ from the status of the ADID being copied or NVALFROM must be specified and differ from any valid from date of the ADID being copied. If NADID is specified and there is no existing version in the AD database, no restriction applies. If NADID is specified and there are existing versions in the AD database, both NSTATUS and NVALFROM must be specified and their combination must differ from any existing combination of NADID.

The following rules apply when this command is executed with the option ADVERS=N active: If NADID is omitted, the only new argument allowed is NSTATUS, which is also mandatory and must differ from the status of the ADID being copied. If NADID is specified and there is no existing version in the AD database, no restriction applies. If NADID is specified and there are existing versions in the AD database, the copy is not allowed.

new_valid_to_date

Date (YYMMDD)

valid_to_date

Date (YYMMDD)

COPY JCLV

Copy a JCL variables table. New arguments begin with N.

```
ACTION=COPY,RESOURCE=JCLV,JCLVTAB=jcl_variable_table_id,
                          NJCLVTAB=new_jcl_variable_table_id;
```

jcl_variable_table_id

Character (char(16)) can be generic (%*)

new_jcl_variable_table

Character (char(16)) can be generic (%*)

Return Codes (for COPY requests):

0

Instruction is successfully processed.

4

Resource not found or user ID does not have RACF® authorization.

8

Instruction failed and an error message is written to the EQQMLOG file.

>8

Refer to Return codes on page 149.

COPY OI

Copy an operation instruction with arguments change. New arguments begin with N.

```
ACTION=COPY,RESOURCE=OI,ADID=application_description,
                        NADID=new_application_description,
                        NJOBNAME=new_jobname,
                        NOPNO=new_operation_number,
                        OPNO=operation_number;
```

application_description

Character (char(16))

new_application_description

Character (char(16))

new_ jobname

Character (Char (8))

new_operation_number

Integer (integer(4))

operation_number

Integer (integer(4))

COPY RG

Copy a run cycle group, changing arguments. New arguments begin with N.

```
ACTION=COPY,RESOURCE=RG,RGID=run_cycle_group,
                        NRGID=new_run_cycle_group,
                        NRGOWNER=new_owner;
```

run_cycle_group

Character (char(8))

new_run_cycle_group

Character (char(8))

new_owner

Character (char(16))

DELETE

The DELETE instructions deletes a record from the database, current plan or long term plan.

AD

An application description.

CPCOND

Current plan condition.

CPLAT

Operation user-defined late information.

CPOC

An occurrence to current plan.

CPOCPRE

A predecessor of a current plan occurrence. The predecessor can be an occurrence or an external operation.

CPOP

An operation to current plan.

CPPRE

A predecessor of a current plan operation. The predecessor operation must be external.

CPSIMP

Current plan condition dependency.

CPSR

A special resource to a current plan operation.

JS

A JCL from the JCL repository.

JSCOM

Multiple JCL entries from the JS file.

LTCPRE

A conditional predecessor to long-term plan occurrence.

LTOC

An occurrence to long-term plan.

LTPRE

A predecessor to long-term plan occurrence.

OI

Operator instructions.

RG

A run cycle group.

DELETE AD

Suppress an application.

```
ACTION=DELETE,RESOURCE=AD,STATUS=adstatus,
                          ADID=application_description,
                          GROUP=authority_group_name,
                          GROUPDEF=group_definition,
                          MATCHTYP=match_type,
                          OWNER=owner,
                          PRIORITY=priority,
                          TYPE=type_of_ad,
                          VALFROM=valid_from_date,
                          VALTO=valid_to_date;
```

adstatus

A or P (char(1))

application_description

Character (char(16)) can be generic (%*)

authority_group_name

Character (char(8)) can be generic (%*)

group_definition

Character (char(16)) can be generic (%*)

match_type

Character (char(4)) can be generic (%*)

owner

Character (char(16)) can be generic (%*)

priority

One integer (integer(4))

type_of_ad

A or G or * (char(1))

valid_from_date

Date (YYMMDD)

valid_to_date

Date (YYMMDD)

Note: MATCHTYP can have the following values: EXA, LOW, or HIGH. If you specify MATCHTYP, you must specify at least VALTO or VALFROM. If MATCHTYP=EXA, then only the version with validity period exactly matching the specified value is deleted. If MATCHTYP=LOW, then all the versions with validity from or validity to lower or equal to the value specified by VALFROM or VALTO are deleted. If MATCHTYP=HIGH, then all the versions with validity from or validity to higher or equal to the value specified by VALFROM or VALTO are deleted.

The validity periods of the versions not deleted will remain unchanged (as default) or will be adjusted to preserve the same logic used by the ISPF dialog if the command is run with ADVERS=Y specified in the last OPTIONS card.

DELETE CPCOND

Suppress a current plan condition.

```
ACTION=DELETE,RESOURCE=CPCOND,ADID=application_description,
                              IA=input_arrival_datetime,
```

```
                              OPNO=operation_number,
                              CONDID=condition_ID;
```

application_description

Character (char(16)) can be generic (%*)

input_arrival_datetime

Date and time: YYMMDDHHMM

operation_number

Integer (integer(4))

condition_ID

Integer (integer(3))

Specify all the operands (the complete condition key).

DELETE CPLAT

Delete operation user-defined late information.

```
ACTION=INSERT,RESOURCE=CPLAT,ADID=application_description,
                              IA=input_arrival_datetime,
                              OPNO=operation_number;
```

application_description

Character (char(16)) can be generic (%*)

input_arrival_datetime

Date and time (YYMMDDHHMM)

operation_number

Integer (integer(4))

DELETE CPOC

Suppress a current plan occurrence.

```
ACTION=DELETE,RESOURCE=CPOC,ADID=application_description,
                              IA=input_arrival_datetime,
                              OSTATUS=oc_status;
```

application_description

Character (char(16)) can be generic (%*)

input_arrival_datetime

Date and time (YYMMDDHHMM) or 'X'

oc_status

Occurrence status: C, D, E, P, S, U, or W (char(1))

Note:

1. ADID and IA arguments identify an occurrence. If the value of IA (IA=X) is unknown, the OSTATUS argument completes the search for the occurrence. If there is more than one occurrence, the Batch Command Interface tool returns a return code of 10, and the occurrence is not modified. If IA is not specified as X, then OSTATUS is ignored.
2. If LTP=Y has been specified in the options card and the IA date specified or defaulted is higher than the CP end date, the occurrence will be deleted in the LTP. Of course, if IA=X is coded, the option LTP=Y is meaningless, so BCIT will only search if in the CP there is a single occurrence in the given OSTATUS.

DELETE CPOCPRE

Suppress a predecessor to a current plan occurrence.

```
ACTION=DELETE,RESOURCE=CPOCPRE,ADID=application_description,
                               IAD=input_arrival_date,
                               IA=input_arrival_datetime,
                               IAT=input_arrival_time,
                               PREADID=pre_application_description,
                               PREIAD=pre_input_arrival_date,
                               PREIA=pre_input_arrival_datetime,
                               PREIAT=pre_input_arrival_time,
                               PREOPNO=pre_operation_number;
```

application_description

Character (char(16)) can be generic (%*)

input_arrival_date

Date (YYMMDD)

input_arrival_datetime

Date and time (YYMMDDHHMM)

input_arrival_time

Time (HHMM)

pre_application_description

Character (char(16)) can be generic (%*)

pre_input_arrival_date

Date (YYMMDD)

pre_input_arrival_datetime

Date and time (YYMMDDHHMM)

pre_input_arrival_time

Time (HHMM)

pre_operation_number

Integer (integer(4))

ADID and IA arguments identify an occurrence.

The predecessor application is identified by PREADID, its occurrence by PREIA (if it is an occurrence predecessor) and its operation by PREOPNO (if it is an operation predecessor).

The operands ADID and PREADID are mandatory if the predecessor is external. The input arrival date and time is mandatory information, but it can be supplied in different ways:

1. Specifying IA in the normal way.
2. Specifying IA by entering 'YYMMDD' in place of the first six characters; the code will substitute them with the current date.
3. Specifying IAD and IAT separately.
4. Specifying only IAT; in which case the code will take the current date as the default for IAD.

If IA is omitted, then at least IAT is mandatory. The predecessor input arrival date and time is mandatory information only when there is an external predecessor. Also in this case it can be entered in different ways:

1. Specifying PREIA in the normal way.
2. Specifying PREIA by entering 'YYMMDD' in place of the first six characters; the code will substitute them with the current date.
3. Specifying PREIAD and PREIAT separately.
4. Specifying only PREIAT; in which case the code will take the current date as the default for PREIAD.

If PREIA is omitted and it is an external predecessor, then at least PREIAT is mandatory.

DELETE CPOP

Suppress an operation from current plan occurrence.

```
ACTION=DELETE,RESOURCE=CPOP,ADID=application_description,
                            IA=input_arrival_datetime,
                            OSTATUS=oc_status,
                            OPNO=operation_number;
```

application_description

Character (char(16)) can be generic (%*)

input_arrival_datetime

Date and time (YYMMDDHHMM) or 'X'

oc_status

Occurrence status. C, D, E, P, S, U, or W (char(1)).

operation_number

Integer (integer(4))

Note: ADID and IA arguments identify an occurrence. If the value of IA (IA=X) is specified as 'X' (IA=X), the following might occur:

- There is only one occurrence of the given ADID: in this case it is not needed to specify OSTATUS.
- There is more than one occurrence, but only one in a specified status: in this case it is necessary to specify OSTATUS.
- There is more than one occurrence in a given status: in this case IA=X cannot work because adding OSTATUS is not enough to identify the occurrence and IA must contain exactly the input arrival date and time of the occurrence for which the deletion of the operation is requested.

If more than one occurrence matches the specified arguments, the BCI tool returns a return code of 10 and the operation will not be deleted. The OPNO argument identifies the operation with the occurrence. If IA is not specified as X, then OSTATUS is ignored. It is not possible to use DELETE CPOP if there is only one operation in the occurrence.

DELETE CPPRE

Suppress a predecessor to a current plan operation.

```
ACTION=DELETE,RESOURCE=CPPRE,ADID=application_description,
                             IAD=input_arrival_date,
                             IA=input_arrival_datetime,
                             IAT=input_arrival_time,
                             OPNO=operation_number,
                             PREADID=pre_application_description,
                             PREIAD=pre_input_arrival_date,
                             PREIA=pre_input_arrival_datetime,
                             PREIAT=pre_input_arrival_time,
                             PREMAND=is_predecessor_mandatory,
                             PREOPNO=pre_operation_number;
```

application_description

Character (char(16)) can be generic (%*)

input_arrival_date

Date (YYMMDD)

input_arrival_datetime

Date and time (YYMMDDHHMM)

input_arrival_time

Time (HHMM)

is_predecessor_mandatory

Character (char(1)) can be Y or N

operation number

Integer (integer(4))

pre_application_description

Character (char(16)) can be generic (%*)

pre_input_arrival_date

Date (YYMMDD)

pre_input_arrival_datetime

Date and time (YYMMDDHHMM)

pre_input_arrival_time

Time (HHMM)

pre_operation_number

Integer (integer(4))

ADID and IA arguments identify an occurrence, OPNO identifies an occurrence operation.

The predecessor application is identified by PREADID, its occurrence by PREIA, and its operation by PREOPNO. If the predecessor is mandatory, set PREMAND to Y (the default is N).

The operands ADID and OPNO are mandatory. The PREOPNO argument is required only for an operation that depends on another operation. PREADID is mandatory if the predecessor is external.

The input arrival date and time is mandatory information, but it can be supplied in different ways:

1. Specifying IA in the normal way.
2. Specifying IA by entering 'YYMMDD' in place of the first six characters; the code will substitute them with the current date.
3. Specifying IAD and IAT separately.
4. Specifying only IAT; in which case the code will take the current date as the default for IAD.

If IA is omitted, then at least IAT is mandatory. The predecessor input arrival date and time is mandatory information only when there is an external predecessor. Also in this case it can be entered in different ways:

1. Specifying PREIA in the normal way.
2. Specifying PREIA by entering 'YYMMDD' in place of the first six characters; the code will substitute them with the current date.
3. Specifying PREIAD and PREIAT separately.
4. Specifying only PREIAT; in which case the code will take the current date as the default for PREIAD.

If PREIA is omitted and it is an external predecessor, then at least PREIAT is mandatory.

DELETE CPSIMP

Delete a condition dependency in the current plan.

```
ACTION=DELETE,RESOURCE=CPSIMP,ADID=application_description,
                          IA=input_arrival_datetime,
                          OPNO=operation_number,
                          CONDID=condition_ID,
                          COUNT=condition_counter,
                          DESC=descriptive_text,
                          PREADID=predecessor_application_description,
                          PREIA=predecessor_input_arrival_datetime,
                          PREOPNO=predecessor_operation_number,
                          PREPSTEP=step_name,
                          PRESTEP=procedure_invocation_step_name,
                          PRETYPE=check_type,
                          PRELOG=logical_operator,
                          PREVRC1=predecessor_return_code_value1,
                          PREVRC2=predecessor_return_code_value2,
                          PREVST=predecessor_status;
```

application_description

Character (char(16)) can be generic (%*)

input_arrival_datetime

Date and time: YYMMDDHHMM

operation_number

Integer (integer(4))

condition_ID

Integer (integer(3))

condition_counter

Integer (integer(3)). Use it to define the rule type:

0

All the condition dependencies in this condition must be true

n>0

At least *n* out of the condition dependencies in this condition must be true

The default is the current value.

descriptive_text

Character (char(16))

predecessor_application_description

Character (char(16))

predecessor_input_arrival_datetime

Date and time: YYMMDDHHMM

predecessor_operation_number

Integer (integer(4))

step_name

Character (char(8)). Use it to define a step level dependency. If the step is not in a procedure, this parameter identifies the job step name, otherwise it identifies the step name in the JCL procedure. It must correspond to the name of an `EXEC PGM=` statement.

procedure_invocation_step_name

Character (char(8)). Use it in conjunction with `PREPSTEP` when defining a step level dependency, only if the step is in a procedure, to identify the name of a step that invokes an in-stream or cataloged procedure. It must correspond to the name of an `EXEC PROC=` statement.

check_type

RC or ST (char(2))

logical_operator

Character (char(2)) can be:

GE

Greater than or equal to. Valid only for RC check type.

GT

Greater than. Valid only for RC condition type.

LE

Less than or equal to. Valid only for RC condition type.

LT

Less than. Valid only for RC check type.

EQ

Equal to.

NE

Not equal to. Use it to specify conditions on final statuses only.

RG

Range.

predecessor_return_code_value1

Character (char(4)). For values with less than four significant characters, use 0 as leading characters.

predecessor_return_code_value2

Character (char(4)) as second boundary in a range expressed by the RG logical operator. For values with less than four significant characters, use 0 as leading characters.

predecessor_status

Character (char(1)) valid only for ST check type

Specify the complete condition key that is the following operands: ADID, IA, OPNO, and CONDID.

DELETE CPSR

Suppress a special resource from a current plan operation.

```
ACTION=DELETE,RESOURCE=CPSR,ADID=application_description,
                            IA=input_arrival_datetime,
                            OPNO=operation_number,
                            RESNAME=resource_name;
```

application_description

Character (char(16)) can be generic (%*)

input_arrival_datetime

Date and time (YYMMDDHHMM)

operation number

Integer (integer(4))

resource_name

Character (char(44))

ADID and IA identify an occurrence, OPNO identifies an operation. RESNAME identifies operation special resource.

DELETE JS

Suppress a JCL from JCL repository. This action suppresses only one JCL at each time. Use JSCOM resource to perform a JCL mass deletion.

```
ACTION=DELETE,RESOURCE=JS,ADID=application_description,
                          IA=input_arrival_datetime,
                          OPNO=operation_number,
                          JOBNAME=jobname,
                          WSNAME=workstation_name;
```

application_description

Character (char(16)) can be generic (%*)

input_arrival_datetime

Date and time (YYMMDDHHMM) or 'X'

operation_number

Integer (integer(4))

jobname

Character (char(8))

workstation_name

Character (char(4))

Note: ADID and IA arguments identify an occurrence. If the value of IA (IA=X) is specified as 'X', the OSTATUS argument completes the search for the occurrence. If more than one occurrence matches the specified arguments, the BCI tool returns a return code of 10 and the occurrence will not be modified. The OPNO argument identifies the operation with the occurrence.

DELETE JSCOM

This command allows to delete JCLs entries from the JS file. If STATUS is F, the command will delete all the entries that satisfy the other operands and that have an input arrival datetime value lower than the one specified by the IA operand. If STATUS is O, the command will delete those entries that satisfy the previous criteria and whose status in the JS file is C or S. If STATUS is C, the command will delete those entries that satisfy the previous criteria and whose status in the JS file is C. If STATUS is omitted, an additional subset will be selected: it is necessary that the corresponding occurrence in the current plan does not exist.

```
ACTION=DELETE,RESOURCE=JSCOM,ADID=application_description,
                             IA=input_arrival_datetime,
                             JOBNAME=jobname,
                             OPNO=operation_number,
                             STATUS=status,
                             WSNAME=workstation_name;
```

application_description

Character (char(16)) can be generic (%*)

input_arrival_datetime

Date and time (YYMMDDHHMM)

jobname

Character (char(8))

operation_number

Integer (integer(4))

status

F, O or C (char(1))

workstation_name

Character (char(4))

DELETE LTCPRE

Suppress a conditional predecessor of an occurrence from long-term plan.

```
ACTION=DELETE,RESOURCE=LTCPRE,ADID=application_description,
                              IAD=input_arrival_date,
```

```
                                IAT=input_arrival_time,
                                PREADID=pre_application_description,
                                PREIAD=pre_input_arrival_date,
                                PREIAT=pre_input_arrival_time;
```

application_description

Character (char(16)) can be generic (%*)

input_arrival_date

Date (YYMMDD)

input_arrival_time

Time (HHMM)

pre_application_description

Character (char(16)) can be generic (%*)

pre_input_arrival_date

Date (YYMMDD)

pre_input_arrival_time

Time (HHMM)

PREADID, PREIAD, and PREIAT arguments identify conditional predecessor.

DELETE LTOC

Suppress a long-term plan occurrence.

```
ACTION=DELETE,RESOURCE=LTOC,ADID=application_description,
                            GROUPDEF=group_definition,
                            IAD=input_arrival_date,
                            IAT=input_arrival_time;
```

application_description

Character (char(16)) can be generic (%*)

group_definition

Character (char(16)) can be generic (%*)

input_arrival_date

Date (YYMMDD)

input_arrival_time

Time (HHMM)

ADID, IAD, and IAT identify an occurrence from long-term plan.

DELETE LTPRE

Suppress a predecessor of an occurrence from long-term plan.

```
ACTION=DELETE,RESOURCE=LTPRE,ADID=application_description,
                             IAD=input_arrival_date,
                             IAT=input_arrival_time,
                             PREADID=pre_application_description,
                             PREIAD=pre_input_arrival_date,
                             PREIAT=pre_input_arrival_time;
```

application_description

Character (char(16)) can be generic (%*)

input_arrival_date

Date (YYMMDD)

input_arrival_time

Time (HHMM)

pre_application_description

Character (char(16)) can be generic (%*)

pre_input_arrival_date

Date (YYMMDD)

pre_input_arrival_time

Time (HHMM)

PREADID, PREIAD, and PREIAT arguments identify predecessor.

DELETE OI

Suppress an operation instruction.

```
ACTION=DELETE,RESOURCE=OI,ADID=application_description,
                          OPNO=operation_number,
                          VALFROM=valid_from_datetime,
                          VALTO=valid_to_datetime,
                          MATCHTYP=type_of_match;
```

application_description

Character (char(16)) can be generic (%*)

operation_number

Integer (integer(4))

valid_from_datetime

Datetime (YYMMDDHHMM)

valid_to_datetime

Datetime (YYMMDDHHMM)

type_of_match

Character (char(4))

Note: MATCHTYP allows only three values to be specified: EXA , LOW, HIGH. If MATCHTYP is specified, then at least 1 operand between VALTO and VALFROM must be specified. If VALTO and/or VALFROM is specified, then MATCHTYP must be specified. VALTO and VALFROM cannot be specified if a permanent OI is selected among those to be deleted. If MATCHTYP=EXA, then only the version(s), with validity period(s) matching exactly the value(s) specified, will be deleted. If MATCHTYP=LOW, then all the version(s), with validity from and/or validity to lower or equal than the value(s) specified by VALFROM (and/or VALTO), will be deleted. If MATCHTYP=HIGH, then all the version(s), with validity from and/or validity to higher or equal than the value(s) specified by VALFROM (and/or VALTO), will be deleted.

DELETE RG

Suppress a run cycle group.

```
ACTION=DELETE,RESOURCE=RG,RGID=run_cycle_group_name,
                          RGOWNER=owner,
                          RGCALEND=calendar_name,
                          RGVARTAB=variable_table_name,
                          RUNNAME=run_cycle_name,
                          RUNCALrun_cycle_calendar_name,
                          RUNVTAB=run_cycle_variable_table_name,
                          RUNSETID=run_cycle_subset_name;
```

run_cycle_group_name

Character (char(8)) can be generic (%*)

owner

Character (char(16)) can be generic (%*)

calendar_name

Character (char(16)) can be generic (%*)

variable_table_name

Character (char(16)) can be generic (%*)

run_cycle_name

Character (char(8)) can be generic (%*)

run_cycle_calendar_name

Character (char(16)) can be generic (%*)

run_cycle_variable_table_name

Character (char(16)) can be generic (%*)

run_cycle_subset_name

Character (char(8)) can be generic (%*)

Return Codes (for DELETE requests)

0

Instruction is successfully processed.

4

Resource not found or user ID does not have RACF® authorization.

This return code is not applicable to DELETE CPPRE; when the predecessor does not exist, RC=8 is set.

8

Instruction failed and an error message is written to the EQQMLOG file.

>8

See Return codes on page 149.

EXPORT

The EXPORT instruction exports an AD, OI, RG, or all, from an IBM Z Workload Scheduler to the same scheduler or to another subsystem. The AD is written to the EXPORTAD file. The OI is written to the EXPORTOI file. The RG is written to the EXPORTRG file. You can export AD, OI, RG, or all, with a selected segment or with generic arguments.

EXPORT AD

The syntax is the same as for LIST ADCOM, except for the operands TYPE and GROUPDEF, that are not supported.

Note: The output file must have been previously allocated with RECFM=VB (LRECL and BLKSIZE can have any value, because they are automatically set by the program).

EXPORT OI

The syntax is the same as for LIST OICOM.

Note: The output file must have been previously allocated with RECFM=VB (LRECL and BLKSIZE can have any value, because they are automatically set by the program).

EXPORT RG

The syntax is the same as for LIST RGCOM.

Note: The output file must have been previously allocated with RECFM=VB (LRECL and BLKSIZE can have any value, because they are automatically set by the program).

GROUPDEF support

The parameter GROUPDEF has been added to the following functions:

LIST

Lists all applications in the AD, CP, or LTP that belong to the same GROUPDEF.

SELECT

Adds GROUPDEF as a selection criteria.

INSERT

If ADID=* is specified, GROUPDEF adds all occurrences that belong to the same group ID:

1. Current plan. If IA is not specified, the input arrival date is set to the current date, unless a different value is specified in the third parameter of the PARM field, and the input arrival time is set to the current time, unless a different value is specified in the forth parameter of the parm field.
2. Long-term plan. IAD and IAT must be specified.

DELETE

The GROUPDEF support for the deletion applies only to the application description data base: there is no support of the GROUPDEF deletion in the CP.

MODIFY

If ADID=* is specified, all applications or occurrences belonging to the same group ID are modified:

1. Current Plan:
   - If IA is not specified, all occurrences are modified.
   - If IA is specified, only corresponding occurrences will be modified.
2. Long-term plan:
   - If IAD is specified, all occurrences corresponding to the date are modified.
   - If IAD and IAT are specified, only corresponding occurrences are modified.

IMPORT

The IMPORT instruction imports AD, OI, and RG resources exported using the EXPORT action. You can import selected resources from a file created by EXPORT with one or more selected segments or with specified arguments.

IMPORT AD

Import an application that has been exported. During export, some arguments could be changed.

Note: The file produced by export cannot be modified manually. It can contain more than one record. To select the correct record in the file used in input by the import function, ADID must be specified if more than one ad has to be imported with the same job. In these cases, a record is selected if ADID matches the value in one of the input records and for this record also the values of group and owner (if specified) match the ones in input. It is possible now to change the name of the application using a new operand of import NEWADID.

If NEWADID is omitted and:

- ADVERS=Y is active, it will be imported using the name specified in ADID provided that there are no already existing versions or, if there are, a new version can be created on the basis of the other operands.
- ADVERS=N is active, it will be imported using the name specified in ADID provided that there are no already existing versions.

If NEWADID is specified and:

- ADVERS=Y is active, it will be imported using the name specified in NEWADID provided that there are no already existing versions or, if there are, a new version can be created on the basis of the other operands.
- ADVERS=N is active, it will be imported using the name specified in NEWADID provided that there are no already existing versions.

If VALFROM and VALTO values are specified, they are set in the imported application. This can cause the failure in importing an application with multiversioning: The new value is forced to overwrite the original multiversioned validity causing the message EQQY724I to be issued.

```
ACTION=IMPORT,RESOURCE=AD,STATUS=ad_status,
                          ADID=application_description,
                          GROUP=authority_group_name,
                          NEWADID=new_application_description,
                          OWNER=owner,
                          PRIORITY=priority,
                          VALFROM=valid_from_date,
                          VALTO=valid_to_date;
```

adstatus

P (pending) or A (ACTIVE) (char(1))

application_description

Character (char(16))

authority_group_name

Character (char(8))

new_application_description

Character (char(16))

owner

Character (char(16))

priority

Integer: 1 through 9 (integer(4))

valid_from_date

Date (YYMMDD)

valid_to_date

Date (YYMMDD)

IMPORT OI

Imports an operator instruction that has been exported. During export, some arguments could be changed.

Note: The file produced by export cannot be modified manually. it can contain more than one record. In order to select the correct record in the file used in input by the import function, adid must be specified if more than one OI has to be imported with the same job. In these cases a record is selected if adid matches the value in one of the input records and for this record also the values of opno (if specified) matches the ones in input. It will be possible now to change the name of the application using a new operand of import newadid. If omitted the OI will be imported using the name specified in adid, provided that the ad is already present in the ad file.

```
ACTION=IMPORT,RESOURCE=OI,ADID=application_description,
                          NEWADID=new_application_description,
                          OPNO=operation_number;
```

application_description

Character (char(16))

new_application_description

Character (char(16))

operation_number

Integer (integer(4))

IMPORT RG

Import a run cycle group that has been exported. During export, some arguments could be changed.

Note: The file produced by export cannot be modified manually. It can contain more than one record. To select the correct record in the file used in input by the import function, RGID must be specified if more than one run cycle group has to be imported with the same job. In these cases, a record is selected if RGID matches the value in one of the input records and for this record also the values of group and owner (if specified) match the ones in input. It is possible now to change the name of the run cycle group using a new operand of import NEWRGID.

```
ACTION=IMPORT,RESOURCE=RG,RGID=run_cycle_group,
                          NEWRGID=new_run_cycle_group,
                          OWNER=owner;
```

run_cycle_group

Character (char(8))

new_run_cycle_group

Character (char(8))

owner

Character (char(16))

INSERT

The INSERT instruction inserts resources into the corresponding tables.

AD

An application description (use BATCH LOADER or COPY or EXPORT/IMPORT) in ADs database.

OI

Operator instructions (use BATCH LOADER or COPY or EXPORT/IMPORT) in OI database.

RG

A run cycle group (use BATCH LOADER or COPY or EXPORT/IMPORT) in RG database.

CPCOND

Current plan condition.

CPLAT

Operation user-defined late information.

CPOC

An occurrence in the current plan.

CPOCPRE

A predecessor to the current plan occurrence. The predecessor can be an occurrence or an external operation.

CPOP

An operation in the current plan.

CPPRE

A predecessor to the current plan operation. Application of predecessor must be external.

CPSIMP

Current plan condition dependency.

CPSR

A special resource in the current plan operation.

LTOC

An occurrence in the long-term plan.

LTPRE

A predecessor to the long-term plan occurrence.

Notes:

1. It is impossible to use the SDUR and EDUR arguments at the same time.
2. WSNAME and JOBNAME are mandatory operands. An ADID is identified by ADID and IA. If there is only one occurrence (regardless of its status) specify the correct value of IA or IA=X. If there is only one occurrence in a given status, specify the correct value of IA or IA=X and the OSTATUS to identify the occurrence. If there is more than one occurrence in the status specified by the OSTATUS operand, a return code of 10 is sent.
3. The character R in restartable_operation is sent to PIF as blank.

INSERT CPCOND

Insert a condition in the current plan.

```
ACTION=INSERT,RESOURCE=CPCOND,ADID=application_description,
                          IA=input_arrival_datetime,
                          OPNO=operation_number,
                          CONDID=condition_ID,
                          COUNT=condition_counter,
                          DESC=descriptive_text,
                          PREADID=predecessor_application_description,
                          PREIA=predecessor_input_arrival_datetime,
                          PREOPNO=predecessor_operation_number,
                          PREPSTEP=step_name,
                          PRESTEP=procedure_invocation_step_name,
                          PRETYPE=check_type,
                          PRELOG=logical_operator,
                          PREVRC1=predecessor_return_code_value1,
                          PREVRC2=predecessor_return_code_value2,
                          PREVST=predecessor_status;
```

application_description

Character (char(16)) can be generic (%*)

input_arrival_datetime

Date and time: YYMMDDHHMM

operation_number

Integer (integer(4))

condition_ID

Integer (integer(3))

condition_counter

Integer (integer(3)). Use it to define the rule type:

0

All the condition dependencies in this condition must be true

n>0

At least *n* out of the condition dependencies in this condition must be true

The default is 0.

descriptive_text

Character (char(16))

predecessor_application_description

Character (char(16))

predecessor_input_arrival_datetime

Date and time: YYMMDDHHMM

predecessor_operation_number

Integer (integer(4))

step_name

Character (char(8)). Use it to define a step level dependency. If the step is not in a procedure, this parameter identifies the job step name, otherwise it identifies the step name in the JCL procedure. It must correspond to the name of an `EXEC PGM=` statement.

procedure_invocation_step_name

Character (char(8)). Use it in conjunction with `PREPSTEP` when defining a step level dependency, only if the step is in a procedure, to identify the name of a step that invokes an in-stream or cataloged procedure. It must correspond to the name of an `EXEC PROC=` statement.

check_type

RC or ST (char(2))

logical_operator

Character (char(2)) can be:

GE

Greater than or equal to. Valid only for RC check type.

GT

Greater than. Valid only for RC condition type.

LE

Less than or equal to. Valid only for RC condition type.

LT

Less than. Valid only for RC check type.

EQ

Equal to.

NE

Not equal to. Use it to specify conditions on final statuses only.

RG

Range.

predecessor_return_code_value1

Character (char(4)). For values with less than four significant characters, use 0 as leading characters.

predecessor_return_code_value2

Character (char(4)) as second boundary in a range expressed by the RG logical operator. For values with less than four significant characters, use 0 as leading characters.

predecessor_status

Character (char(1)) valid only for ST check type

COUNT, DESC, and PREIA are optional argument.

The other argument are required. PREIA also is required for external predecessors.

To create an internal dependency, do not specify either PREADID or PREIA.

INSERT CPLAT

Insert operation user-defined late information.

```
ACTION=INSERT,RESOURCE=CPLAT,ADID=application_description,
                             IA=input_arrival_datetime,
                             LATACT=action_taken_if_op_not_started,
                             LATACTDT=datetime_by_which_op_must_start_before_
                    action_is_taken,
```

```
                    LATALEDT=datetime_by_which_op_must_start_before_
alert_is_issued,
                    OPNO=operation_number;
```

application_description

Character (char(16)) can be generic (%*)

input_arrival_datetime

Date and time (YYMMDDHHMM)

action_taken_if_op_not_started

Y or N (char(1))

datetime_by_which_op_must_start_before_action_is_taken

Date and time (YYMMDDHHMM)

datetime_by_which_op_must_start_before_alert_is_issued

Date and time (YYMMDDHHMM)

operation_number

Integer (integer(4))

INSERT CPOC

Insert an occurrence to current plan.

```
ACTION=INSERT,RESOURCE=CPOC,ADID=application_description,
                            IA=input_arrival_datetime,
                            DEADLINE=deadline_date_datetime,
                            PRIORITY=priority,
                            ERRCODE=error_code,
                            DESC=descriptive_text,
                            GROUP=authority_group_name,
                            OWNER=owner,
                            ODESC=descriptive_text_of_owner,
                            GROUPDEF=application_groupid,
                            JCLVTAB= JCL_variable_table_id;
```

application_description

Character (char(16)) can be generic (%*)

application_groupid

Character (char 16))

authority_group_name

Character (char(8)) can be generic (%*)

deadline_date_datetime

Date and time (YYMMDDHHMM)

descriptive_text

Character (char(24))

descriptive_text_of_owner

Character (char(24))

error_code

Character (char(4))

input_arrival_datetime

Date and time (YYMMDDHHMM)

jcl_variable_table_id

Character (char(16)) can be generic (%*)

owner

Character (char(16)) can be generic (%*)

priority

One integer (integer(4))

Note:

1. If LTP=Y was specified in the options card and the IA date specified or defaulted is higher than the CP end date, the occurrence will be inserted in the LTP. In this case the operands DESC, GROUP, OWNER, and ODESC, if specified, will be ignored.
2. The following facts must be carefully considered when GROUPDEF is specified:
    - Each of the applications belonging to the group is inserted separately, in alphabetical order (if CPDEPR=N is specified or defaulted in the options card) or in the order required to correctly resolve the dependencies (if CPDEPR=Y is specified in the options card). If any errors occur, BCIT ends and this results in only a partial insertion of the applications belonging to GROUPDEF.
    - If there are more than one version of a given application in the AD, you must check that the specified or defaulted IA falls within one of the corresponding validity periods; otherwise, the BCIT job will terminate with RC=8.
    - If CPDEPR was set to Y:
        - It is not allowed to specify for an operation an external dependency on itself or on another operation of the same application.
        - Cross dependencies are not supported. That is, if APPL1 contains an external dependency on an operation of APPL2 and APPL2 contains an external dependency on an operation of APPL1, this will be detected and the BCIT job will terminate with RC=8 (in this case, the INSERT will not be performed).

- It is your responsibility to ensure that the CP does not contain occurrences of the applications belonging to GROUPDEF: if they exist, the dependencies will be resolved in a different way from expected.
- It is not allowed to update the applications belonging to GROUPDEF at the same time as the batch job execution: the outcome of such concurrent update is unpredictable.

INSERT CPOCPRE

Insert a predecessor to a current plan occurrence.

```
ACTION=INSERT,RESOURCE=CPOCPRE,ADID=application_description,
                               IAD=input_arrival_date,
                               IA=input_arrival_datetime,
                               IAT=input_arrival_time,
                               OSTATUS=occurrence_status,
                               PREADID=pre_application_description,
                               PREIAD=pre_input_arrival_date,
                               PREIA=pre_input_arrival_datetime,
                               PREIAT=pre_input_arrival_time,
                               PREOSTAT=pre_occurrence_status,
                               PREOPNO=pre_operation_number;
```

application_description

Character (char(16)) can be generic (%*)

input_arrival_date

Date (YYMMDD)

input_arrival_datetime

Date and time (YYMMDDHHMM) or X

input_arrival_time

Time (HHMM)

occurrence_status

Occurrence status (char(1))

pre_application_description

Character (char(16)) can be generic (%*)

pre_input_arrival_date

Date (YYMMDD)

pre_input_arrival_datetime

Date and time (YYMMDDHHMM) or X

pre_input_arrival_time

Time (HHMM)

pre_occurrence_status

Occurrence PRED status (char(1))

pre_operation_number

integer (integer(4))

Note: An ADID is identified by ADID and IA (instead of IA it is possible to use IAD and IAT). If there is only one occurrence (regardless of its status) specify the correct value of IA or IA=X. If there is only one occurrence in a given status, specify the correct value of IA or IA=X and OSTATUS to identify the occurrence. If there is more than one occurrence in the status specified by the OSTATUS operand, a return code of 10 is sent. The same is true for the predecessor occurrence, which is identified by PREADID and PREIA (instead of PREIA it is possible to use PREIAD and PREIAT). If there is only one occurrence (regardless of its status) specify the correct value of PREIA or PREIA=X. If there is only one occurrence in a given status, specify the correct value of PREIA or PREIA=X and PREOSTAT to identify the occurrence. If there is more than one occurrence in the status specified by the PREOSTAT operand, a return code of 10 is sent.

INSERT CPOP

Insert an operation in current plan occurrence.

```
ACTION=INSERT,RESOURCE=CPOP,ADID=application_description,
                            AEC=automatic_error_completion,
                            AJR=automatic_job_hold/release,
                            ASUB=automatic_job_submission,
                            AUTFUNC=automated_function,
                            CLATE=cancel_if_late,
                            CLNTYPE=cleanup_type,
                            COIFBNDD=complete_if_bind_fails,
                            COIFBNDZ=complete_if_bind_fails,
                            COMMTEXT=command_text,
                            COMPINFO=completion_info,
                            CONDRJOB=conditional_recovery_job,
                            DEADWTO=issue_deadline_wto,
                            DESC=descriptive_text,
                            DURATION=duration,
                            EDUR=estimated_duration,
                            EXPJCL=expanded_jcl,
                            EXTNAME= extended_name
                            EXTSE= scheduling_environment_name
                            FORM=form_number,
                            HRC=highest_successful_retcode,
                            IA=input_arrival_datetime,
                            JCLASS=jobclass,
                            JOBCRT=critical_job,
                            JOBNAME=jobname,
                            JOBPOL=WLM_job_policy,
                            MONITOR=externally_monitored,
                            OPDL=op_deadline_datetime,
                            OPDLACT=op_deadline_action,
                            OPIA=op_input_arrival_datetime,
                            OPNO=operation_number,
                            OSTATUS=occurrence_status,
```

```
                                PSUSE=paralleles_server_required,
                                R1USE=resource1_required,
                                R2USE=resource2_required,
                                READID=remote_job_appl_desc,
                                REJOBNM=remote_job_name,
                                REJSNM=remote_job_stream_name,)
                                REJSWS=remote_ws_name,
                                REOPNO=rem_job_op_num,
                                RERUT=reroutable_operation,
                                RESTA=restartable_operation,
                                SECELEM=security_element,
                                SDUR=estimated_duration_inseconds,
                                STATUS=op_status,
                                TIMEDEP=timedependent_job,
                                USRSYS=user_sysout,
                                WLMSCLS=WLM_service_class,
                                WSNAME=wsname;
```

application_description

Character (char(16)) can be generic (%*)

automated_function

Character (char(8))

automatic_error_completion

A character: Y or N (char(1))

automatic_job_hold/release

A character: Y or N (char(1))

automatic_job_submission

A character: Y or N (char(1))

cancel_if_late

A character: Y or N (char(1))

cleanup_type

A character: A, M, I or N (char(1))

command_text

Character (char(255))

complete_if_bind_fails

A character: Y or N (char(1)). Use `COIFBNDD` for an IBM Workload Scheduler remote job, use `COIFBNDZ` for an IBM Z Workload Scheduler remote job.

completion_info

Character (char(64))

Note: This keyword is positional. For details, see *Managing the Workload*.

conditional_recovery_job

A character: Y or N (Char(1))

critical_job

A character: P, W, or N (char(1))

descriptive_text

Character (char(24))

duration

Time (Integer (4)) number of hundredths of a second

estimated_duration

Time (HHMM)

estimated_duration_inseconds

(Integer (4)) number of seconds

expanded_jcl

A character: Y or N (char(1))

extended_name

Character (char(54))

externally_monitored

A character: Y or N (char(1))

form_number

Character (char(8))

highest_successful_retcode

Integer (integer (4)): allowed values from 0 to 4095

input_arrival_datetime

Date and time (YYMMDDHHMM) or X

issue_deadline_Wto

A character: Y or N (char(1))

jobclass

A character (A–Z, 0–9) (char(1))

jobname

Character (char(8)) can be generic (%*)

occurrence status

C, D, E, P, S, U, or W (Char(1))

op_deadline_datetime

Date and time (YYMMDDHHMM)

op_deadline_action

A character (char(1)):

A

Only an alert message is issued.

C

The operation is set to Complete, if its status allows it. Otherwise, it is NOPed.

E

The operation is set to Error with ODEA, if its status allows it. Otherwise, this setting is postponed at the time when the status allows it.

N

The operation and all its internal successors are NOPed, if their status allows NOPing. Otherwise, it is ignored.

op_input_arrival_datetime

Date and time (YYMMDDHHMM)

op_status

A character (A, R, S, C, D, I, E, W, U, or *) (char(1))

operation_number

integer (integer(4))

paralleles_server_requi

Integer (integer(4))

remote_job_appl_desc

Character (char(16)). Only for an IBM Z Workload Scheduler remote job.

remote_job_name

Character (char(40)). Only for an IBM Workload Scheduler remote job.

remote_job_op_num

Integer (integer(4)). Only for an IBM Z Workload Scheduler remote job.

remote_job_stream_name

Character (char(16)). Only for an IBM Workload Scheduler remote job.

remote_ws_name

Character (char(16)). Only for an IBM Workload Scheduler remote job.

reroutable_operation

A character: Y or N (char(1))

resource1_required

Integer (integer(4))

resource2_required

Integer (integer(4))

restartable_operation

A character: Y, N or R (char(1))

scheduling_environment_name

Character (char(16))

security_element

Character (char(8))

timedependent_job

A character: Y or N (char(1))

user_sysout

A character: Y or N (char(1))

WLM_job_policy

L, D, S, or C (char(1))

WLM_service_class

Character (char(8)) can be generic (%*)

workstation_name

Character (char(4)) can be generic (%*)

INSERT CPPRE

Insert a predecessor to a current plan operation.

```
ACTION=INSERT,RESOURCE=CPPRE,ADID=application_description,
                             IAD=input_arrival_date,
                             IA=input_arrival_datetime,
                             IAT=input_arrival_time,
                             OSTATUS=occurrence_status,
                             OPNO=operation_number,
                             PREADID=pre_application_description,
                             PREIAD=pre_input_arrival_date,
                             PREIA=pre_input_arrival_datetime,
                             PREIAT=pre_input_arrival_time,
```

```
                    PREOSTAT=pre_occurrence_status,
                    PREOPNO=pre_operation_number,
                    TRPTTIME=transport_time;
```

application_description

Character (char(16)) can be generic (%*)

input_arrival_date

Date (YYMMDD)

input_arrival_datetime

Date and time (YYMMDDHHMM) or X

input_arrival_time

Time (HHMM)

occurrence_status

Occurrence status (char(1))

operation number

Integer (integer(4))

pre_application_description

Character (char(16)) can be generic (%*)

pre_input_arrival_date

Date (YYMMDD)

pre_input_arrival_datetime

Date and time (YYMMDDHHMM) or X

pre_input_arrival_time

Time (HHMM)

pre_occurrence_status

Occurrence PRED status (char(1))

pre_operation_number

integer (integer(4))

transport_time

Integer (integer(4))

Note:

1. An ADID is identified by ADID and IA (instead of IA it is possible to use IAD and IAT). If there is only one occurrence (regardless of its status) specify the correct value of IA or IA=X. If there is only one occurrence in a given status, specify the correct value of IA or IA=X and OSTATUS to identify the occurrence. If there is

more than one occurrence in the status specified by the OSTATUS operand, a return code of 10 is sent. The same is true for the predecessor occurrence, which is identified by PREADID and PREIA (instead of PREIA it is possible to use PREIAD and PREIAT). If there is only one occurrence (regardless of its status) specify the correct value of PREIA or PREIA=X. If there is only one occurrence in a given status, specify the correct value of PREIA or PREIA=X and PREOSTAT to identify the occurrence. If there is more than one occurrence in the status specified by the PREOSTAT operand, a return code of 10 is sent.

2. To create an internal dependency, do not specify either PREADID or PREIA.

INSERT CPSIMP

Insert a condition dependency in the current plan.

```
ACTION=INSERT,RESOURCE=CPSIMP,ADID=application_description,
                          IA=input_arrival_datetime,
                          OPNO=operation_number,
                          CONDID=condition_ID,
                          COUNT=condition_counter,
                          DESC=descriptive_text,
                          PREADID=predecessor_application_description,
                          PREIA=predecessor_input_arrival_datetime,
                          PREOPNO=predecessor_operation_number,
                          PREPSTEP=step_name,
                          PRESTEP=procedure_invocation_step_name,
                          PRETYPE=check_type,
                          PRELOG=logical_operator,
                          PREVRC1=predecessor_return_code_value1,
                          PREVRC2=predecessor_return_code_value2,
                          PREVST=predecessor_status;
```

application_description

Character (char(16)) can be generic (%*)

input_arrival_datetime

Date and time: YYMMDDHHMM

operation_number

Integer (integer(4))

condition_ID

Integer (integer(3))

condition_counter

Integer (integer(3)). Use it to define the rule type:

0

All the condition dependencies in this condition must be true

n>0

At least *n* out of the condition dependencies in this condition must be true

The default is the current value for this condition.

descriptive_text

Character (char(16))

predecessor_application_description

Character (char(16))

predecessor_input_arrival_datetime

Date and time: YYMMDDHHMM

predecessor_operation_number

Integer (integer(4))

step_name

Character (char(8)). Use it to define a step level dependency. If the step is not in a procedure, this parameter identifies the job step name, otherwise it identifies the step name in the JCL procedure. It must correspond to the name of an `EXEC PGM=` statement.

procedure_invocation_step_name

Character (char(8)). Use it in conjunction with `PREPSTEP` when defining a step level dependency, only if the step is in a procedure, to identify the name of a step that invokes an in-stream or cataloged procedure. It must correspond to the name of an `EXEC PROC=` statement.

check_type

RC or ST (char(2))

logical_operator

Character (char(2)) can be:

GE

Greater than or equal to. Valid only for RC check type.

GT

Greater than. Valid only for RC condition type.

LE

Less than or equal to. Valid only for RC condition type.

LT

Less than. Valid only for RC check type.

EQ

Equal to.

NE

Not equal to. Use it to specify conditions on final statuses only.

RG

Range.

predecessor_return_code_value1

Character (char(4)). For values with less than four significant characters, use 0 as leading characters.

predecessor_return_code_value2

Character (char(4)) as second boundary in a range expressed by the RG logical operator. For values with less than four significant characters, use 0 as leading characters.

predecessor_status

Character (char(1)) valid only for ST check type

COUNT, DESC, and PREIA are optional argument.

The other argument are required. PREIA also is required for external predecessors.

To create an internal dependency, do not specify either PREADID or PREIA.

INSERT CPSR

Insert a special resource in the current plan.

```
ACTION=INSERT,RESOURCE=CPSR,ADID=MATENCIOP,IA=9201291500,RESNAME=name,
RESUSAGE=X,OPNO=10.
ACTION=INSERT,RESOURCE=CPSR,ADID=application_description,
                            IA=input_arrival_datetime,
                            ONCOMPL=resource_on_complete,
                            ONERROR=resource_on_error,
                            OPNO=operation_number,
                            RESNAME=resource_name,
                            RESUSAGE=resource_usage
                            QUANTITY=resource_quantity;
```

application_description

Character (char(16)) can be generic (%*)

input_arrival_datetime

Date and time (YYMMDDHHMM)

resource_on_complete

Character (char(1)) On complete action: Y, N, or R

resource_on_error

Character (char(1)) On error action: Y or N

operation_number

Integer (integer(4))

resource_name

Character (char(44))

resource_usage

Character: S or X (char(1))

resource_quantity

Integer (integer(4))

INSERT LTOC

Insert an occurrence in the long-term plan.

```
ACTION=INSERT,RESOURCE=LTOC,ADID=application_description,
                            DEADLINE=deadline_datetime,
                            ERRCODE=error_code,
                            GROUPDEF=group_definition,
                            IAD=input_arrival_date,
                            IAT=input_arrival_time,
                            PRIORITY=priority;
```

application_description

Character (char(16)) can be generic (%*)

deadline_datetime

Date and time (YYMMDDHHMM)

error_code

Character (char(4))

group_definition

Character (char(16)) can be generic (%*)

input_arrival_date

Date (YYMMDD)

input_arrival_time

Time (HHMM)

priority

One integer (integer(4))

INSERT LTPRE

Insert a predecessor in a long-term plan operation.

```
ACTION=INSERT,RESOURCE=LTPRE,ADID=application_description,
                             IAD=input_arrival_date,
                             IAT=input_arrival_time,
                             PREADID=pre_application_description,
```

```
                                PREIAD=pre_input_arrival_date,
                                PREIAT=pre_input_arrival_time;
```

application_description

Character (char(16)) can be generic (%*)

input_arrival_date

Date (YYMMDD)

input_arrival_time

Time (HHMM)

pre_application_description

Character (char(16)) can be generic (%*)

pre_input_arrival_date

Date (YYMMDD)

pre_input_arrival_time

Time (HHMM)

Return codes

0

Instruction successfully processed.

4

Resource not found or user ID have no RACF® authorization.

8

Instruction failed and an error message is written to EQQMLOG.

>8

See Return codes on page 149.

LIST

With the LIST instruction, you can list all the resources of the scheduler. The LIST request retrieves all the segments (COMMON SEGMENT and ADDITIONAL SEGMENTS) of a resource. If you want only the common segment, you must use SELECT instruction for this resource. You can see that the resources names are suffixed by COM.

This action also permits you to give AD, OI, and RG in BATCH LOADER (options BL=Y and BLPRT=Y) or only extract their keys (options BL=Y and BLPRT=N) for using to build SELECT action with specific resources.

The resources used by the LIST action are as follows:

ADCOM

Application descriptions

ADKEY

Application description keys

CLCOM

Calendars

CPCONDCO

Current plan conditions

CPOC, CPOCCOM

Current plan occurrences

CPOPCOM

Current plan operations

CPWSCOM

Current plan workstations

CPWSVCOM

Current plan virtual workstations

CRITSUCS

Operation critical successors

JCLVCOM

JCL variable tables

JSCOM

JCL segments

LTOCCOM

Long-term plan occurrences

OICOM

Operator instructions

PRCOM

Periods

RGCOM

Run cycle groups

RGKEY

Run cycle group keys

WSCOM

Workstations

WSVCOM

Virtual workstations

LIST ADCOM

List of application descriptions.

```
ACTION=LIST,RESOURCE=ADCOM,STATUS=adstatus,
                           ADID=application_description,
                           GROUP=authority_group_name,
                           GROUPDEF=groupdef,
                           OWNER=owner,
                           PRIORITY=priority,
                           TYPE=type,
                           VALFROM=valid_from_date,
                           VALTO=valid_to_date;
```

adstatus

A or P (char(1))

application_description

Character (char(16)) can be generic (%*)

authority_group_name

Character (char(8)) can be generic (%*)

groupdef

Character (char(16))

owner

Character (char(16)) can be generic (%*)

priority

One integer (integer(4))

type

A or G (char(1))

valid_from_date

Date: YYMMDD

valid_to_date

Date: YYMMDD

An application key is the argument concatenation ADID, STATUS, and VALTO Further arguments only complete the list.

Note: The result stays in referenced files as:

SYSPRINT(LRECL=132)

If OPTIONS BL=N (detailed format)

AD (LRECL=23)

If OPTIONS BL=Y and BLPRT=N (ADs keys)

BATCHL (LRECL=80)

If OPTIONS BL=Y & BLPRT=Y(BATCH LOADER FORMAT)

See GROUPDEF support on page 170 for details on GROUPDEF support.

LIST ADKEY

List ADs Keys.

```
ACTION=LIST,RESOURCE=ADKEY,ADID=application_description,
                           STATUS=adstatus,
                           VALTO=valid_to_date;
```

application_description

Character (char(16)) can be generic (%*)

adstatus

A or P (char(1))

valid_to_date

Date: YYMMDD

LIST CLCOM

List calendars.

```
ACTION=LIST,RESOURCE=CLCOM,CALENDAR=calendar;
```

calendar

Character (char(16)) can be generic (%*)

LIST CPCONDCO

List current plan conditions.

```
ACTION=LIST,RESOURCE=CPCONDCO,ADID=application_description,
                              IA=input_arrival_datetime,
                              OPNO=operation_number,
                              CONDID=condition_ID,
                              CONDVAL=condition_status;
```

application_description

Character (char(16)) can be generic (%*)

input_arrival_datetime

Date and time: YYMMDDHHMM

operation_number

Integer (integer(4))

condition_ID

Integer (integer(3))

condition_status

U, T, or F (char(1))

The workstation key contains ADID, IA, OPNO, and CONDID. Specify at least one of these operands.

LIST CPOC, CPOCCOM

List current plan occurrences.

In the RESOURCE keyword, you can specify CPOC or CPOCCOM because they are equivalent. CPOCCOM was introduced together with the support of application dependencies, to identify the part of the occurrence in the plan without the variable section of dependencies.

```
ACTION=LIST,RESOURCE=CPOC,ADID=application_description,
                          GROUP=authority_group_name,
                          GROUPDEF=group_definition,
                          IA=input_arrival_datetime,
                              MCPADDED=mcp_added,
                          MONITOR=externally_monitored,
                          STATUS=oc_status,
                          OWNER=owner,
                          PRIORITY=priority,
                          RERUN=rerun;
```

application_description

Character (char(16)) can be generic (%*)

authority_group_name

Character (char(8)) can be generic (%*)

externally_monitored

Y or N (char(1))

group_definition

Character(char(16))

input_arrival_datetime

Date and time: YYMMDDHHMM

mcp_added

Y or N (char(1))

oc_status

C, D, E, P, S, U, or W (char(1))

owner

Character (char(16)) can be generic (%*)

priority

One integer (integer(4))

rerun

Y or N (char(1))

IA and ADID must be supplied.

Note: Results are written in referenced files as:

SYSPRINT(LRECL=132)

If OPTIONS BL=N (detailed format)

CPOC (LRECL=80)

If OPTIONS BL=Y (ADID,IA,STATUS)

LIST CPOPCOM

List of operations current plan occurrences.

```
ACTION=LIST,RESOURCE=CPOPCOM,ADID=application_description,
                             GROUP=authority_group_name,
                             CONDRJOB=conditional_recovery_job,
                             DPREM=removable_by_DP,
                             ERRCODE=error_code,
                             EXTNAME=extended_name
                             EXTSE=scheduling_environment_name
                             IA=input_arrival_datetime,
                             JOBNAME=jobname,
                             LATEE=op_late_on_latest_start_time_
                                or_not_started_action_or_not_started_alert,
                             LATEL=op_late_on_latest_start_time,
                             LATEN=op_late_on_not_started_alert_or_not_started_act,
                             MATCHTYP=match_type,
                             MONITOR=externally_monitored,
                             OPNO=operation_number,
                             OWNER=owner,
                             PRIORITY=priority,
                             SHADOWJ=shadow_job,
                             STATUS=op_status,
                             UNEXPRC=unexpected_rc,
                             WAITFORW=started_on_WAIT_workstation
```

```
                        WAITSE=waiting_for_SE,
                        WMPRED=waiting_for_Mandatory_Pending_Predecessors,
                        WPMPRED=waiting_for_Pending_OR_
    Mandatory_Pending_Predecessors,
                        WPPRED=waiting_for_Pending_Predecessors,
                        WSNAME=workstation_name;
```

application_description

Character (char(16)) can be generic (%*)

authority_group_name

Character (char(8)) can be generic (%*)

conditional_recovery_job

Y or N (char(1))

removable_by_DP

Y or N (char(1))

error_code

Character (char(4))

extended_name

Character (char(54))

scheduling_environment_name

Character (char(16))

externally_monitored

Y or N (char(1))

input_arrival_datetime

Date and time: YYMMDDHHMM

jobname

Character (char(8)) can be generic (%*)

op_late_on_latest_start_time_or_not_started_alert_or_not_started_action

Y or N(char(1))

op_late_on_latest_start_time

Y or N(char(1))

op_late_on_not_started_alert_or_not_started_action

Y or N(char(1))

match_type

Character (char(3))

op_status

*, A, R, S, C, D, I, E, W, or U (char(1))

operation_number

Integer (integer(4))

owner

Character (char(16)) can be generic (%*)

priority

One integer (integer(4))

shadow_job

Y or N (char(1))

started_on_WAIT_workstation

Y or N (char(1))

unexpected_rc

Y or N (char(1))

waiting_for_Mandatory_Pending_Predecessors

Y or N (char(1))

waiting_for_Pending_OR_Mandatory_Pending_Predecessors

Y or N (char(1))

waiting_for_Pending_Predecessors

Y or N (char(1))

waiting_for_SE

Y or N (char(1))

workstation_name

Character (char(4)) can be generic (%*)

Note: There is only one value accepted for MATCHTYP: EXA. If MATCHTYP=EXA is specified, STATUS=* will be interpreted as a normal character instead of as a generic matching character.

ADID and IA identify an occurrence with an operation identified by the OPNO parameter. Further arguments complete the list.

Note: Results are referenced in files as:

SYSPRINT(LRECL=132)

If OPTIONS BL=N (detailed format)

CPOP (LRECL=80)

If OPTIONS BL=Y (ADID,IA,OPNO,STATUS, JOBNAME)

LIST CPWSCOM

List current plan workstations.

```
ACTION=LIST,RESOURCE=CPWSCOM,WSAUTO==automation_workstation,
                             WSNAME=workstation_name,
                             WSREP=workstation_rep_attri,
                             WSTYPE=workstation_type,
                             WSVIRT=virtual_workstation,
                             WSWAIT=wait_workstation,
                             WSZCENTR=z-centric_workstation;
```

automation_workstation

Y or N (char(1))

workstation_name

Character (char(4)) can be generic (%*)

workstation_rep_attri

A, S, C, or N (char(1))

workstation_type

C, G, R, or P (char(1))

virtual_workstation

Y or N (char(1))

wait_workstation

Y or N (char(1))

z-centric_workstation

Y or N (char(1))

The workstation key contains WSNAME, WSTYPE, WSREP and WSTWS.

At least one of the operands starting with WS must be specified.

LIST CPWSVCOM

List current plan virtual workstations.

```
ACTION=LIST,RESOURCE=CPWSVCOM,WSNAME=workstation_name,
                              WSDEST=workstation_destination;
```

workstation_name

Character (char(4)) can be generic (%*)

workstation_destination

Character (char(8)) can be generic (%*). To indicate a local destination, specify ********

The workstation key contains WSNAME and WSDEST.

At least one of the two operands must be specified.

LIST JCLVCOM

List of JCL variable tables.

```
    ACTION=LIST,RESOURCE=JCLVCOM,JCLVTAB=jcl_variable_table_id
```

jcl_variable_table_id

Character (char(16)) can be generic (%*)

The output is written in the file referenced by the DATAFI DD card.

LIST JSCOM

List JCLs from current plan operations.

```
ACTION=LIST,RESOURCE=JSCOM,ADID=application_description,
                           IA=input_arrival_datetime,
                           JOBNAME=jobname,
                           OPNO=operation_number,
                           WSNAME=workstation_name;
```

application_description

Character (char(16)) can be generic (%*)

input_arrival_datetime

Date and time: YYMMDDHHMM

jobname

Character (char(8)) can be generic (%*)

operation_number

Integer (integer(4))

workstation_name

Character (char(4)) can be generic (%*)

LIST LTOCCOM

List long-term plan occurrences.

```
ACTION=LIST,RESOURCE=LTOCCOM,ADID=application_description,
                             GROUP=authority_group_name,
                             IAD=input_arrival_date,
                             IAT=input_arrival_time,
                             OWNER=owner;
```

application_description

Character (char(16)) can be generic (%*)

authority_group_name

Character (char(8)) can be generic (%*)

input_arrival_date

Date: YYMMDD

input_arrival_time

Time: HHMM

owner

Character (char(16)) can be generic (%*)

ADID, IAD, and IAT identify an occurrence in the long-term plan.

LIST OICOM

List operator instructions.

```
ACTION=LIST,RESOURCE=OICOM,ADID=application_description,
                           OPNO=operation_number;
```

application_description

Character (char(16)) can be generic (%*)

operation_number

Integer (integer(4))

Note: Result created in referenced files with:

SYSPRINT(LRECL=132)

If OPTIONS BL=N (detailed result)

OI (LRECL=23)

If OPTIONS BL=Y and BLPRT=N (OI Keys)

BATCHL (LRECL=80)

If OPTIONS BL=Y and BLPRT=Y (BATCH LOADER format)

LIST PRCOM

List periods.

```
ACTION=LIST,RESOURCE=PRCOM,PERIOD=period_name,
                           PRTYPE=period_type;
```

period_name

Character (char(8)) can be generic (%*)

period_type

A, W, or N (char(1))

LIST RGCOM

List of run cycle groups.

```
ACTION=LIST,RESOURCE=RGCOM,RGID=run_cycle_group_name,
                           RGOWNER=owner,
                           RGCALEND=calendar_name,
                           RGVARTAB=variable_table_name,
                           RUNNAME=run_cycle_name,
                           RUNCALrun_cycle_calendar_name,
                           RUNVTAB=run_cycle_variable_table_name,
                           RUNSETID=run_cycle_subset_name;
```

run_cycle_group_name

Character (char(8)) can be generic (%*)

owner

Character (char(16)) can be generic (%*)

calendar_name

Character (char(16)) can be generic (%*)

variable_table_name

Character (char(16)) can be generic (%*)

run_cycle_name

Character (char(8)) can be generic (%*)

run_cycle_calendar_name

Character (char(16)) can be generic (%*)

run_cycle_variable_table_name

Character (char(16)) can be generic (%*)

run_cycle_subset_name

Character (char(8)) can be generic (%*)

LIST RGKEY

List of run cycle group keys.

```
ACTION=LIST,RESOURCE=RGKEY,RGID=run_cycle_group_name,
                           RGOWNER=owner,
                           RGCALEND=calendar_name,
                           RGVARTAB=variable_table_name,
                           RUNNAME=run_cycle_name,
                           RUNCALrun_cycle_calendar_name,
                           RUNVTAB=run_cycle_variable_table_name,
                           RUNSETID=run_cycle_subset_name;
```

run_cycle_group_name

Character (char(8)) can be generic (%*)

owner

Character (char(16)) can be generic (%*)

calendar_name

Character (char(16)) can be generic (%*)

variable_table_name

Character (char(16)) can be generic (%*)

run_cycle_name

Character (char(8)) can be generic (%*)

run_cycle_calendar_name

Character (char(16)) can be generic (%*)

run_cycle_variable_table_name

Character (char(16)) can be generic (%*)

run_cycle_subset_name

Character (char(8)) can be generic (%*)

LIST WSCOM

List workstations in WS database.

```
ACTION=LIST,RESOURCE=WSCOM,WSAUTO=automation_workstation
                           WSNAME=workstation_name,
                           WSTYPE=workstation_type,
                           WSREP=workstation_rep_attri,
                           WSVIRT=virtual_workstation,
                           WSWAIT=wait_workstation,
```

```
                                WSZCENTR=z-centric_workstation,
                                WSRETYPE=remotengine_workstation;
```

automation_workstation

Y or N (char(1))

remote-engine_workstation

Z or D (char(1))

virtual_workstation

Y or N (char(1))

wait_workstation

Y or N (char(1))

workstation_name

Character (char(4)) can be generic (%*)

workstation_rep_attri

A, S, C, or N (char(1))

workstation_type

C, G, R, or P (char(1))

z-centric_workstation

Y or N (char(1))

Note:

1. At least one of the operands starting with WS must be specified.
2. If you specify WSNAME, this setting supersedes all the other arguments.

LIST WSVCOM

List virtual workstations in WS database.

```
ACTION=LIST,RESOURCE=WSVCOM,WSNAME=workstation_name,
                             WSDEST=workstation_destination;
```

workstation_name

Character (char(4)) can be generic (%*)

workstation_destination

Character (char(8)) can be generic (%*). To indicate a local destination, specify ********

At least one of the two operands must be specified.

Return codes

0

The instruction is successfully processed.

4

The resource was not found or the user ID does not have the required RACF® authorization.

8

The instruction failed, and an error message has been written to the EQQMLOG file.

>8

See Return codes on page 149.

LISTSTAT

The LISTSTAT instruction returns a code that reflects the status of the resource. You can list the status for CPOC or CPOPCOM resources.

LISTSTAT CPOC

Return a code according to the current status of an occurrence

```
ACTION=LISTSTAT,RESOURCE=CPOC,ADID=application_description,
                              GROUP=authority_group_name,
                              IA=input_arrival_datetime,
                              MCPADDED=mcp_added,
                              OWNER=owner,
                              PRIORITY=priority,
                              RERUN=rerun;
```

application_description

Character (char(16))

authority_group_name

Character (char(8))

input_arrival_datetime

Date and time (YYMMDDHHMM) or X

mcp_added

Y or N (char(1))

owner

Character (char(16))

priority

One integer (integer(4))

rerun

Y or N (char(1))

valid_from_date

Date (YYMMDD)

valid_to_date

Date (YYMMDD)

Notes:

1. The occurrence must be uniquely identified.
2. The correct value of IA or IA=X must be specified when there is only one occurrence of the given adid to avoid that the value specified or defaulted through the parm invocation parameter be passed to PIF.

LISTSTAT CPOPCOM

Return a code according to the current status of an operation.

```
ACTION=LISTSTAT,RESOURCE=CPOPCOM,ADID=application_description,
                                 GROUP= authority_group_name,
                                 ERRCODE=error_code,
                                 IA=input_arrival_datetime,
                                 JOBNAME=jobname,
                                 OPNO=operation_number,
                                 OWNER=owner,
                                 PRIORITY=priority,
                                 WSNAME=wsname;
```

application_description

Character (char(16))

authority_group_name

Character (char(8))

error_code

Character (char(4))

input_arrival_datetime

Date and time (YYMMDDHHMM) or X

jobname

Character (char(8))

operation_number

Integer (integer(4))

owner

Character (char(16))

priority

One integer (integer(4))

workstation_name

Character (char(4))

Note: IA is the input arrival date and time of the occurrence, not of the operation. There are no mandatory operands, but those supplied must be enough to uniquely identify the operation. The correct value of IA or IA=X must be specified when there is only one occurrence of the given adid to avoid that the value specified or defaulted through the parm invocation parameter be passed to PIF.

Return codes

0

Instruction successfully processed

4

Resource not found or user ID has no RACF® authorization

8

Instruction failed and a error message is written to EQQMLOG file

31

Occurrence status C (completed)

32

Occurrence status D (deleted)

33

Occurrence status E (ended in error)

34

Occurrence status P (processor pending)

35

Occurrence status S (started)

36

Occurrence status U (undecided)

37

Occurrence status W (no started operations)

40

Operation status *

41

Operation status A (waiting for input to arrive)

42

Operation status R (ready)

43

Operation status S (started)

44

Operation status C (completed)

45

Operation status D (deleted)

46

Operation status I (interrupted)

47

Operation status E (ended in error)

48

Operation status W (waiting for a predecessor)

49

Operation status U (undecided)

>49

See Return codes on page 149.

MODIFY

The MODIFY instruction modifies some resources in the corresponding database. The current version of the Batch Command Interface modifies the following resources:

CPCOND

Current plan condition.

CPEXT

An extended name for the current plan operation.

CPOC

An occurrence in the current plan.

CPOP

An operation in the current plan.

CPREND

Information about an IBM Workload Scheduler remote job.

CPRENZ

Information about an IBM Z Workload Scheduler remote job.

CPSAI

System automation information for the current plan operation.

LTOC

An occurrence in the long-term plan.

MODIFY CPCOND

Modify a condition in the current plan.

```
ACTION=MODIFY,RESOURCE=CPCOND,ADID=application_description,
                              IA=input_arrival_datetime,
                              OPNO=operation_number,
                              CONDID=condition_ID,
                              COUNT=condition_counter,
                              DESC=descriptive_text;
```

application_description

Character (char(16)) can be generic (%*)

input_arrival_datetime

Date and time: YYMMDDHHMM

operation_number

Integer (integer(4))

condition_ID

Integer (integer(3))

condition_counter

Integer (integer(3)). Use it to define the rule type:

0

All the condition dependencies in this condition must be true

n>0

At least *n* out of the condition dependencies in this condition must be true

The default is the current value.

descriptive_text

Character (char(16))

Specify the complete condition key that is the following operands: ADID, IA, OPNO, and CONDID.

MODIFY CPEXT

Create, modify, or delete the extended name of an operation in the current plan.

```
ACTION=MODIFY,RESOURCE=CPEXT,ADID=application_description,
                             EXTNAME=extended_name,
                             EXTSE=scheduling_environment_name,
                             IA=input_arrival_datetime,
                             OPNO=operation_number;
```

application_description

Character (char(16)) can be generic (%*)

extended_name

Character (char(54)). To delete the operation extended name, enter `EXTNAME=' '`

scheduling_environment_name

Character (char(16)).

input_arrival_datetime

Date and time (YYMMDDHHMM)

operation_number

Integer (integer(4))

MODIFY CPOC

Modify an occurrence in the current plan. If the attempt fails, BCIT will perform at a predefined time interval other attempts (for a maximum of 3).

```
ACTION=MODIFY,RESOURCE=CPOC,ADID=application_description,
                            ALLMON=all_operations_externally_monitored,
                            IA=input_arrival_datetime,
                            OSTATUS=oc_status,
                            IANEW=input_arrival_datetime,
                            DEADLINE=deadline_datetime,
                            PRIORITY=priority,
                            ERRCODE=error_code,
                            STATUS=new_status,
                            GROUPDEF=group_definition,
                            JCLVTAB=jcl_variable_table_id;
```

all_operation_externally_ monitored

A character: Y or N (char(1))

application_description

Character (char(16)) can be generic (%*)

deadline_datetime

Date and time (YYMMDDHHMM)

error_code

Character (char(4))

group_definition

Character (char(16)) can be generic (%*)

input_arrival_datetime

Date and time (YYMMDDHHMM) or 'X'

jcl_variable_table_id

Character (char(16)) can be generic (%*)

new_status

New status for the occurrence (char(1))

oc_status

Occurrence status: C, D, E, P, S, U, or W (char(1))

priority

One integer (integer(4))

Note: ADID and IA arguments identify an occurrence. If the value of IA is specified as 'X' (IA=X), the OSTATUS argument completes the search for the occurrence. If IA=X is specified and more than one occurrence matches the specified arguments, the BCI tool returns a return code of 10 and the occurrence will not be modified.

MODIFY CPOP

Modify an operation in the current plan. If the attempts fails, BCIT will perform other attempts at a predefined time interval (for the maximum of 3 total).

```
ACTION=MODIFY,RESOURCE=CPOP,ADID=application_description,
                            AEC=automatic_error_completion,
                            AJR=automatic_job_hold/release,
                            ASUB=automatic_job_submission,
                            CLATE=cancel_if_late,
                            CLNTYPE=cleanup_type,
                            CONDRJOB=conditional_recovery_job,
                            COIFBNDD=complete_if_bind_fails,
                            COIFBNDZ=complete_if_bind_fails,
                            DEADWTO=issue_deadline_wto,
                            DESC=descriptive_text,
                            DURATION=duration,
                            EDUR=estimated_duration,
```

```
                                ERRCODE=error_code,
                                EXPJCL=expanded_jcl,
                                FORM=form_number,
                                GROUP=authority_group_name,
                                HRC=highest_successful_retcode,
                                IA=input_arrival_datetime,
                                JCLASS=jobclass,
                                JOBCRT=critical_job,
                                JOBNAME=jobname,
                                JOBPOL=WLM_job_policy,
                                MONITOR=externally_monitored,
                                OPCMD=operation_command,
                                OPDL=op_deadline_datetime,
                                OPDLACT=op_deadline_action,
                                OPIA=op_input_arrival_datetime,
                                OPNO=operation_number,
                                OSTATUS=oc_status,
                                PSUSE=paralleles_server_required,
                                R1USE=resource1_required,
                                R2USE=resource2_required,
                                READID=remote_job_appl_desc,
                                REJOBNM=remote_job_name,
                                REJSNM=remote_job_stream_name,
                                REJSWS=remote_ws_name,
                                REOPNO=remote_job_op_num,
                                RERUT=reroutable_operation,
                                RESTA=restartable_operation,
                                SDUR=estimated_duration_inseconds,
                                STATUS=op_status,
                                TIMEDEP=timedependent_job,
                                USRSYS=user_sysout,
                                WLMSCLS=WLM_service_class,
                                WSNAME=workstation_name;
```

application_description

Character (char(16)) can be generic (%*)

automatic_error_completion

A character: Y or N (char(1))

automatic_job_hold/release

A character: Y or N (char(1))

automatic_job_submission

A character: Y or N (char(1))

cancel_if_late

A character: Y or N (char(1))

cleanup_type

A character: A, M, I or N (char(1))

complete_if_bind_fails

A character: Y or N (char(1)). Use `COIFBNDD` for an IBM Workload Schedulerremote job, use `COIFBNDZ` for anIBM Z Workload Scheduler remote job.

conditional_recovery_job

A character: Y or N (Char(1))

critical_job

A character: P, W, or N (char(1))

descriptive_text

Character (char(24))

duration

Time (Integer (4)) number of hundredths of a second

error_code

Character (char(4))

estimated_duration

Time (HHMM)

estimated_duration_inseconds

(Integer (4)) number of seconds

expanded_jcl

A character: Y or N (char(1))

externally_monitored

A character: Y or N (char(1))

form_number

Character (char(8))

highest_successful_retcode

Integer (integer (4)): allowed values from 0 to 4095.

input_arrival_datetime

Date and time (YYMMDDHHMM) or 'X'

issue_deadline_wto

A character: Y or N (char(1))

jobclass

A character: A–Z, 0–9 (char(1))

jobname

Character (char(8)) can be generic (%*)

oc_status

Occurrence status: C, D, E, P, S, U, or W (char(1))

op_deadline_datetime

Date and time (YYMMDDHHMM)

op_deadline_action

A character (char(1)):

A

Only an alert message is issued.

C

The operation is set to Complete, if its status allows it. Otherwise, it is NOPed.

E

The operation is set to Error with ODEA, if its status allows it. Otherwise, this setting is postponed at the time when the status allows it.

N

The operation and all its internal successors are NOPed, if their status allows NOPing. Otherwise, it is ignored.

op_input_arrival_datetime

Date and time (YYMMDDHHMM)

op_status

A character: A, R, S, C, D, I, E, W, U, or * (char(1))

operation_command

Character (char(2) can be:

- EX = Execute operation
- KJ = Kill operation[1]
- MH = Hold operation
- MR = Release operation
- NP = NOP operation
- PN = Prompt reply no
- PY = Prompt reply yes
- UN = Un-NOP operation

Note:

1. Applies only to operations running on IBM Z Workload Scheduler Agents (z-centric agents).

operation_number

Integer (integer(4))

paralleles_server_required

Integer (integer(4))

remote_job_appl_desc

Character (char(16)). Only for aIBM Z Workload Scheduler remote job.

remote_job_name

Character (char(40)). Only for an IBM Workload Scheduler remote job.

remote_job_op_num

Integer (integer(4)). Only for an IBM Z Workload Scheduler remote job.

remote_job_stream_name

Character (char(16)). Only for an IBM Workload Scheduler remote job.

remote_ws_name

Character (char(16)). Only for an IBM Workload Scheduler remote job.

reroutable_operation

A character: Y or N (char(1))

resource1_required

Integer (integer(4))

resource2_required

Integer (integer(4))

restartable_operation

A character: Y, N or R (Char(1))

timedependent_job

A character: Y or N (char(1))

user_sysout

A character: Y or N (Char(1))

WLM_job_policy

L, D, S, or C (char(1))

WLM_service_class

Character (char(8)) can be generic (%*)

workstation_name

Character (char(4)) can be generic (%*)

Note:

1. The parameters of the MODIFY CPOP request are the same as the parameters of the INSERT CPOP request. In addition, the MODIFY CPOP request has the ERRCODE and the OPCMD parameters.
2. An ADID is identified by ADID and IA. If there is only one occurrence (regardless of its status) specify the correct value of IA or IA=X. If there is only one occurrence in a given status, specify the correct value of IA or IA=X and OSTATUS to identify the occurrence. If there is more than one occurrence in the status specified by the OSTATUS operand, a return code of 10 is sent and the occurrence will not be modified. The OPNO argument identifies the operation within the occurrence.
3. It is impossible to use SDUR and EDUR arguments at the same time.
4. The character R in WLM_job_policy and in restartable_operation will be sent to PIF as blank.

MODIFY CPREND

Modify the information about a job scheduled to run on a remote IBM Workload Scheduler engine.

```
ACTION=MODIFY,RESOURCE=CPREND,REJSNM=rem_job_stream_name,
                              REJOBNM=rem_job_name,
                              REJSWS=rem_ws_name,
                              COIFBNDF=comp_if_bind_fails;
```

comp_if_bind_fails

A character: Y or N (Char(1))

rem_job_name

Character (char(40))

rem_job_stream_name

Character (char(16))

rem_ws_name

Character (char(16))

MODIFY CPRENZ

Modify the information about a job scheduled to run on an IBM Z Workload Scheduler remote engine.

```
ACTION=MODIFY,RESOURCE=CPRENZ,READID=rem_job_appl_desc,
                              REOPNO=rem_job_op_num,
                              COIFBNDF=comp_if_bind_fails;
```

comp_if_bind_fails

A character: Y or N (Char(1))

rem_job_appl_desc

Character (char(16))

rem_job_op_num

Integer (integer(4))

MODIFY CPSAI

Create, modify, or delete the system automation information for an operation in the current plan.

```
ACTION=MODIFY,RESOURCE=CPSAI,ADID=application_description,
                             AUTFUNC=automated_function,
                             COMMTEXT=command_text,
                             COMPINFO=completion_info;
                             IA=input_arrival_datetime,
                             OPNO=operation_number,
                             SECELEM=security_element;
```

application_description

Character (char(16)) can be generic (%*).

automated_function

Character (char(8)). To delete the operation automated function, enter AUTFUNC=' '.

command_text

Character (char(255)).

completion_info

Character (char(64)). To delete the operation completion information, enter COMPINFO=' '.

Note: This keyword is positional. For details, see *Managing the Workload*.

input_arrival_datetime

Date and time (YYMMDDHHMM).

operation_number

Integer (integer(4)).

security_element

Character (char(8)). To delete the operation security element, enter SECELEM=' '.

MODIFY LTOC

Modify an occurrence in the long-term plan.

```
ACTION=MODIFY,RESOURCE=LTOC,ADID=application_description,
                            DEADLINE=deadline_datetime,
                            ERRCODE=error_code,
                            GROUPDEF=group_definition
                            IAD=input_arrival_date,
```

```
                    IAT=input_arrival_time,
                    PRIORITY=priority;
```

application_description

Character (char(16)) can be generic (%*)

deadline_datetime

Date and time (YYMMDDHHMM)

error_code

Character (char(4))

group_definition

Character(char(16))

input_arrival_date

Date (YYMMDD)

input_arrival_time

Time (HHMM)

priority

One integer (integer(4))

ADID, IAD, and IAT identify an occurrence to the long-term plan.

Return codes (for MODIFY requests)

0

Instruction is successfully processed.

4

Resource not found or user ID does not have RACF® authorization.

8

Instruction failed and an error message displays on EQQMLOG file.

10

More than one occurrence matches the specified arguments.

>10

See Return codes on page 149.

OPTIONS

The OPTIONS instruction gives you a choice of process options; you can specify options to be used when performing PIF requests. You can use these options to automatically resolve dependencies when adding current plan or long-term plan occurrences with INSERT.

```
ACTION=OPTIONS,ADOICHK=adoichk,
                        ADVERS=advers,
                        BL=batchl,
                        BLPRT=blprt,
                        CPDEPR=cpdepr,
                        ERROR=error,
                        LTDEPR=ltdepr,
                        LTP=ltp,
                        RETRY=number_of_retries;
```

adoichk

Y or N(char(1)). Specify whether or not you want AD/OI consistency checks to be made every time an application is deleted or modified. Consistency checks involve looking in the application description data base for matches for all the operator instructions in the application. Any operator instructions without match are deleted. Default: N.

advers

Y or N (char(1)).

When Y is specified, in all the cases in which the running of the requested command impacts the AD database for applications having more than one validity period, BCIT ensures that the resulting different versions of the application have consecutive validity intervals, preserving the same logic used by the ISPF dialogs.

Note: Mixed use of ADVERS=Y and ADVERS=N is not allowed. Before running any BCIT command with ADVERS=Y, you must update the versions affected using the ISPF dialogs ensuring the basic versioning rule is followed. If any ADID has only one version in a given status, its value must be 711231; if there are more versions in a given status, the more recent one must a value of 711231. Default: N.

batchl

Y or N (char(1)). Ask the batchl format.

- If BL=Y and BLPRT=Y, the commands LIST ADCOM or OICOM produce the following output:
  - AD (OI) statements (in the format that can be used in input to the batch loader utility) are written in the file referenced by the BATCHL DD card. Note that the PREJOBN keyword will never be produced because the BL utility accepts it as an alternative to PREOPNO, which is always supplied for all the predecessors.
  - AD (OI) keys are written in the file referenced by the AD (OI) DD card.
- If BL=Y and BLPRT=N, only AD and OI keys are written inside their corresponding file (referenced by DD AD or DD OI calls)
- If BL=Y, the LIST CPOC or CPOPCOM result is written in referenced file (CPOC or CPOP). Default: N

blprt

Y or N (char(1)) write AD or OI keys in AD and OI files if BL=Y and BLPRT=N. Default: Y if BL=Y.

cpdepr

Y or N (char(1)) resolve automatically external dependences (extra application) during an insert of a new occurrence in a current plan. Default: N

error

Y or N (char(1)) when Y is specified, processing of SYSIN statements stops in case of failure. Default: N

ltdepr

Y or N (char(1)) resolve automatically external dependencies (extra application) during an insert of a new occurrence in an long-term plan. Default: N

ltp

Y or N (char(1)). When Y is specified, if INSERT or DELETE actions are requested for resource CPOC and the input arrival time specified or defaulted is later than the current plan end time, the requested action will be performed on the long-term plan. Default: N

number_of_retries

Integer (integer(4)): allowed values from 0 to 9999. Specifies the number of times that BCIT tries to update an operation or occurrence that is being modified by another user, before it ends in error. The default value is 3; the value 0 is replaced by the default 3.

REPLACE

Use the BATCH LOADER.

SELECT

The SELECT instruction selects a resource. If the resource is not suffixed with COM, all segments will be selected and printed; otherwise only the common segment will be processed.

Whereas LIST retrieves all resources, SELECT retrieves only the selected resource. You have to identify the key of the selected resource, you cannot use generic keys. The list below, gives the resources retrieved by SELECT action.

The SELECT action also gives to you AD, OI, and RG in batch loader format (options BL=Y and BLPRT=Y). From keys created by a list action (options BL=Y and BLPRT=N), you can define SELECT instructions for AD, OI, and RG and send them like SYSIN to the Batch Command Interface tool.

The resources authorized by the SELECT action are as follows:

AD, ADCOM

Application description

CL, CLCOM

Calendars

CPCOND, CPCONDCO

Current plan condition

CPOC, CPOCCOM

Current plan occurrences

CPOP, CPOPCOM

Current plan operations

CPST

Current plan status

CPWS, CPWSCOM

Current plan workstations

JCLPREP

Promptable variables setup to current plan

JCLPREPA

Automatic variables setup to current plan

JCLV, JCLVCOM

JCL variable table

JS, JSCOM

JCL segment

LTOC, LTOCCOM

Long-term plan (LTP) occurrences

OI, OICOM

Operator instructions

PR, PRCOM

Periods (period table)

RG, RGCOM

Run cycle groups

WS, WSCOM

Workstations (workstation table)

SELECT AD

Select an application description.

```
ACTION=SELECT,RESOURCE=AD, STATUS=adstatus,
                           ADID=application_description,
                           GROUP=authority_group_name,
                           OWNER=owner,
                           PRIORITY=priority,
```

```
                              VALFROM=valid_from_date,
                              VALTO=valid_to_date;
```

adstatus

Status: A or P (char(1))

application_description

Character (char(16))

authority_group_name

Character (char(8))

owner

Character (char(16))

priority

One integer (integer(4))

valid_from_date

Date (YYMMDD)

valid_to_date

Date (YYMMDD)

Note: The result is created in the referenced file with:

SYSPRINT(LRECL=132)

If OPTIONS BL=N (detailed format)

BATCHL (LRECL=80)

If OPTIONS BL=Y and BLPRT=Y(BATCH LOADER format)

SELECT CL

Retrieve a calendar.

```
ACTION=SELECT,RESOURCE=CL,CALENDAR=calendar;
```

calendar

Character (char(16))

SELECT CPCOND

Retrieve current plan conditions.

```
ACTION=SELECT,RESOURCE=CPCOND,ADID=application_description,
                              IA=input_arrival_datetime,
                              OPNO=operation_number,
```

```
                              CONDID=condition_ID,
                              CONDVAL=condition_status;
```

application_description

Character (char(16)) can be generic (%*)

input_arrival_datetime

Date and time: YYMMDDHHMM

operation_number

Integer (integer(4))

condition_ID

Integer (integer(3))

condion_status

U, T, or F (char(1))

Specify the complete condition key that is the following operands: ADID, IA, OPNO, and CONDID.

SELECT CPOC, CPOCCOM

Retrieve a current plan occurrence.

In the RESOURCE keyword, specify CPOC or CPOCCOM. CPOCCOM was introduced together with the support of application dependencies, to identify the part of the occurrence in the plan without the variable section of dependencies. If you specify CPOC, the variable section of dependency is returned, when present. If you specify CPOCCOM, the variable section of dependency is not returned.

```
ACTION=SELECT,RESOURCE=CPOC,ADID=application_description,
                            GROUP=authority_group_name,
                            GROUPDEF=group_definition,
                            IA=input_arrival_datetime,
                            MCPADDED=mcp_added,
                            MONITOR=externally_monitored,
                            STATUS=oc_status,
                            OWNER=owner,
                            PRIORITY=priority,
                            RERUN=rerun;
```

application_description

Character (char(16)

authority_group_name

Character (char(8))

externally_monitored

Y or N (char(1))

group_definition

Character(char(16))

input_arrival_datetime

Date and time: YYMMDDHHMM

mcp_added

Y or N (char(1))

oc_status

The IBM Z Workload Scheduler status code: C, D, E, P, S, U, or W (char(1))

owner

Character (char(16))

priority

One integer (integer(4))

rerun

Y or N (char(1))

Note: The result is created in the referenced file by:

SYSPRINT(LRECL=132)

If OPTIONS BL=N (detailed format)

CPOC (LRECL=80)

If OPTIONS BL=Y (ADID,IA,STATUS)

SELECT CPOP

Retrieve an operation of current plan occurrence.

```
ACTION=SELECT,RESOURCE=CPOP,ADID=application_description,
                            CONDRJOB=conditional_recovery_job,
                            DPREM=removable_by_DP,
                            ERRCODE=error_code,
                            EXTNAME=extended_name,
                            EXTSE=scheduling_environment_name,
                            GROUP=authority_group_name
                            IA=input_arrival_datetime,
                            JOBNAME=jobname,
                            LATEE=op_late_on_latest_start_time_
                  or_not_started_action_or_not_started_alert,
                            LATEL=op_late_on_latest_start_time,
                            LATEN=op_late_on_not_started_alert_or_not_started_act,
                            MATCHTYP=match_type,
                            MONITOR=externally_monitored,
                            OPNO=operation_number,
                            OWNER=owner,
```

```
                    PRIORITY=priority,
                    SHADOWJ=shadow_job,
                    STATUS=oc_status,
                    UNEXPRC=unexpected_rc
                    WAITFORW=started_on_WAIT_workstation,
                    WAITSE=waiting_for_SE,
                    WMPRED=waiting_for_Mandatory_Pending_Predecessors,
                    WPMPRED=waiting_for_Pending_OR_Mandatory_
        Pending_Predecessors,
                    WPPRED=waiting_for_Pending_Predecessors,
                    WSNAME=workstation_name;
```

application_description

Character (char(16))

authority_group_name

Character (char(8))

conditional_recovery_job

Y or N (char(1))

removable_by_DP

Y or N (char(1))

error_code

Character (char(4))

extended_name

Character (char(54))

scheduling_environment_name

Character (char(16))

externally_monitored

Y or N(char(1))

input_arrival_datetime

Date and time (YYMMDDHHMM)

jobname

Character (char(8))

op_late_on_latest_start_time_or_not_started_alert_or_not_started_action

Y or N(char(1))

op_late_on_latest_start_time

Y or N(char(1))

op_late_on_not_started_alert_or_not_started_action

Y or N(char(1))

match_type

Character (char(3))

op_status

*, A, R, S, C, D, I, E, W, or U (char(1))

operation_number

Integer (integer(4))

owner

Character (char(16))

priority

One integer (integer(4))

shadow_job

Y or N (char(1))

started_on_WAIT_workstation

Y or N (char(1))

unexpected_rc

Y or N (char(1))

waiting_for_Mandatory_Pending_Predecessors

Y or N (char(1))

waiting_for_Pending_OR_Mandatory_Pending_Predecessors

Y or N (char(1))

waiting_for_Pending_Predecessors

Y or N (char(1))

waiting_for_SE

Y or N (char(1))

workstation_name

Character (char(4))

Note: There is only one value accepted for MATCHTYP: EXA. If MATCHTYP=EXA is specified, STATUS=* will be interpreted as a normal character instead of as a generic matching character.

Note: The result is created in the referenced file by:

SYSPRINT(LRECL=132)

If OPTIONS BL=N (detailed format)

CPOP (LRECL=80)

If OPTIONS BL=Y (ADID,IA,OPNO,STATUS, JOBNAME)

SELECT CPST

Retrieve the current plan status.

```
ACTION=SELECT,RESOURCE=CPST
```

SELECT CPWS

List a current plan workstation.

```
ACTION=SELECT,RESOURCE=CPWS,WSAUTO=automation_workstation,
                            WSNAME=workstation_name,
                            WSTYPE=workstation_type,
                            WSREP=workstation_rep_attri,
                            WSVIRT=virtual_workstation,
                            WSWAIT=wait_workstation,
                            WSZCENTR=z-centric_workstation
                            WSRETYPE=remotengine_workstation;
```

automation_workstation

Y or N (char(1))

remote-engine_workstation

Z or D (char(1))

virtual_workstation

Y or N (char(1))

wait_workstation

Y or N (char(1))

workstation_name

Character (char(4))

workstation_rep_attri

A, S, C, or N (char(1))

workstation_type

C, G, R, or P (char(1))

z-centric_workstation

Y or N (char(1))

At least one of the operands starting with WS must be specified.

SELECT CPWSV

Retrieve a current plan virtual workstation.

```
ACTION=SELECT,RESOURCE=CPWSV,WSNAME=workstation_name,
                             WSDEST=workstation_destination;
```

workstation_name

Character (char(4))

workstation_destination

Character (char(8)). To indicate a local destination, specify ********

Specify the complete workstation key that is both the operands.

SELECT JCLPREP

Retrieve a JCLPREP of an operation.

```
ACTION=SELECT,RESOURCE=JCLPREP,ADID=application_description,
                               IA=input_arrival_datetime,
                               OPNO=operation_number;
```

application_description

Character (char(16))

input_arrival_datetime

Date and time (YYMMDDHHMM)

operation_number

Integer (integer(4))

SELECT JCLPREPA

Retrieve a JCLPREPA of an operation.

```
ACTION=SELECT,RESOURCE=JCLPREPA,ADID=application_description,
                                IA=input_arrival_datetime,
                                OPNO=operation_number;
```

application_description

Character (char(16))

input_arrival_datetime

Date and time (YYMMDDHHMM)

operation_number

Integer (integer(4))

SELECT JCLV

Retrieve a variable table.

```
ACTION=SELECT,RESOURCE=JCLV,JCLVTAB=jcl_variable_table_id;
```

jcl_variable_table_id

Character (char(16)) can be generic (%*)

SELECT JS

List JCL of an operation.

```
ACTION=SELECT,RESOURCE=JS,ADID=application_description,
                          IA=input_arrival_datetime,
                          JOBNAME=jobname,
                          OPNO=operation number,
                          WSNAME=workstation_name;
```

application_description

Character (char(16))

input_arrival_datetime

Date and time (YYMMDDHHMM)

jobname

Character (char(8))

operation_number

Integer (integer(4))

workstation_name

Character (char(4)) can be generic (%*)

SELECT LTOC

Retrieve a long-term plan occurrence.

```
ACTION=SELECT,RESOURCE=LTOC,ADID=application_description,
                            GROUP=authority_group_name,
                            IAD=input_arrival_date,
                            IAT=input_arrival_time,
                            OWNER=owner;
```

application_description

Character (char(16))

authority_group_name

Character (char(8))

input_arrival_date

Date (YYMMDD)

input_arrival_time

Time (HHMM)

owner

Character (char(16))

SELECT OI

Retrieve operator instructions of an operation.

```
ACTION=SELECT,RESOURCE=OI,ADID=application_description,
                          OPNO=operation_number,
                          VALTO=valid_to_date;
```

application_description

Character (char(16))

operation_number

Integer (integer(4))

valid_to_date

date (YYMMDDHHMM)

Note:

1. VALTO is an optional operand, which must be specified if more than one temporary OI satisfies the other operands and must not be specified if the request is issued for a permanent OI.
2. The result is created in the referenced files by:

   SYSPRINT(LRECL=132)

   If OPTIONS BL=N (detailed result)

   BATCHL(LRECL=80)

   If OPTIONS BL=Y and BLPRT=Y (BATCH LOADER format)

SELECT PR

Retrieve a period.

```
ACTION=SELECT,RESOURCE=PR,PERIOD=period_name,
                          PRTYPE=period_type;
```

period_name

Character (char(8))

period_type

A,W,N (char(1))

SELECT RG

Select a run cycle group.

```
ACTION=SELECT,RESOURCE=RG,RGID=run_cycle_group_name,
                          RGOWNER=owner,
                          RGCALEND=calendar_name,
                          RGVARTAB=variable_table_name,
                          RUNNAME=run_cycle_name,
                          RUNCALrun_cycle_calendar_name,
                          RUNVTAB=run_cycle_variable_table_name,
                          RUNSETID=run_cycle_subset_name;
```

run_cycle_group_name

Character (char(8)) can be generic (%*)

owner

Character (char(16)) can be generic (%*)

calendar_name

Character (char(16)) can be generic (%*)

variable_table_name

Character (char(16)) can be generic (%*)

run_cycle_name

Character (char(8)) can be generic (%*)

run_cycle_calendar_name

Character (char(16)) can be generic (%*)

run_cycle_variable_table_name

Character (char(16)) can be generic (%*)

run_cycle_subset_name

Character (char(8)) can be generic (%*)

SELECT WS

List a workstation.

```
ACTION=SELECT,RESOURCE=WS,WSAUTO=automation_workstation,
                          WSNAME=workstation_name,
                          WSTYPE=workstation_type,
                          WSREP=workstation_rep_att,
                          WSVIRT=virtual_workstation,
                          WSWAIT=wait_workstation,
                          WSZCENTR=z-centric_workstation,
                          WSRETYPE=remotengine_workstation;
```

automation_workstation

Y or N (char(1))

remote-engine_workstation

Z or D (char(1))

virtual_workstation

Y or N (char(1))

wait_workstation

Y or N (char(1))

workstation_name

Character (char(4))

workstation_rep_attri

A, S, C, or N (char(1))

workstation_type

C, G, R, or P (char(1))

z-centric_workstation

Y or N (char(1))

Note:

1. At least one of the operands starting with WS must be specified.
2. If you specify WSNAME, this setting supersedes all the other arguments.

SELECT WSV

Retrieve a virtual workstation.

```
ACTION=SELECT,RESOURCE=WSV,WSNAME=workstation_name,
                           WSDEST=workstation_destination;
```

workstation_name

Character (char(4))

workstation_destination

Character (char(8))

Specify the complete workstation key that is both the operands.

Return codes

0

Instruction successfully processed

4

Resource not found or user ID has no RACF® authorization

8

Instruction failed and an error message is written to the EQQMLOG file

>8

See Return codes on page 149.

SETSTAT

The SETSTAT request changes the condition status from undecided to true or false, if the original status is undecided because of missing step-end information. It applies to the following resource:

CPSIMP

Current plan condition dependency

SETSTAT CPSIMP

Insert a condition dependency in the current plan.

```
ACTION=SETSTAT,RESOURCE=CPSIMP,ADID=application_description,
                           IA=input_arrival_datetime,
                           OPNO=operation_number,
                           CONDID=condition_ID,
                           COUNT=condition_counter,
                           DESC=descriptive_text,
                           NEWSTAT=new_status,
                           PREADID=predecessor_application_description,
                           PREIA=predecessor_input_arrival_datetime,
                           PREOPNO=predecessor_operation_number,
                           PREPSTEP=step_name,
                           PRESTEP=procedure_invocation_step_name,
                           PRETYPE=check_type,
                           PRELOG=logical_operator,
                           PREVRC1=predecessor_return_code_value1,
                           PREVRC2=predecessor_return_code_value2,
                           PREVST=predecessor_status;
```

application_description

Character (char(16)) can be generic (%*)

input_arrival_datetime

Date and time: YYMMDDHHMM

operation_number

Integer (integer(4))

condition_ID

Integer (integer(3))

condition_counter

Integer (integer(3)). Use it to define the rule type:

0

All the condition dependencies in this condition must be true

n>0

At least *n* out of the condition dependencies in this condition must be true

The default is the current value for this condition.

descriptive_text

Character (char(16))

new_status

Character (char(1)) can be:

T

True

F

False

predecessor_application_description

Character (char(16))

predecessor_input_arrival_datetime

Date and time: YYMMDDHHMM

predecessor_operation_number

Integer (integer(4))

step_name

Character (char(8)). Use it to define a step level dependency. If the step is not in a procedure, this parameter identifies the job step name, otherwise it identifies the step name in the JCL procedure. It must correspond to the name of an `EXEC PGM=` statement.

procedure_invocation_step_name

Character (char(8)). Use it in conjunction with `PREPSTEP` when defining a step level dependency, only if the step is in a procedure, to identify the name of a step that invokes an in-stream or cataloged procedure. It must correspond to the name of an `EXEC PROC=` statement.

check_type

RC or ST (char(2))

logical_operator

Character (char(2)) can be:

GE

Greater than or equal to. Valid only for RC check type.

GT

Greater than. Valid only for RC condition type.

LE

Less than or equal to. Valid only for RC condition type.

LT

Less than. Valid only for RC check type.

EQ

Equal to.

NE

Not equal to. Use it to specify conditions on final statuses only.

RG

Range.

predecessor_return_code_value1

Character (char(4)). For values with less than four significant characters, use 0 as leading characters.

predecessor_return_code_value2

Character (char(4)) as second boundary in a range expressed by the RG logical operator. For values with less than four significant characters, use 0 as leading characters.

predecessor_status

Character (char(1)) valid only for ST check type

COUNT, DESC, and PREIA are optional arguments.

The other argument are required. PREIA also is required for external predecessors.

Return codes

0

Instruction successfully processed.

4

Resource not found or user ID have no RACF® authorization.

8

Instruction failed and an error message is written to EQQMLOG.

>8

See Return codes on page 149.

Chapter 4. Control Language (OCL)

IBM Z Workload Scheduler Control Language (OCL) is a programming language that increases the usability of the program interface (PIF) and offers a comprehensive set of instructions to control the objects managed by the scheduler.

Using OCL, you can adjust and experiment with:

- JCL variables
- Applications and related dependencies
- Application occurrences status
- Special resources
- Applications structure
- User-supplied dates

For more information about OCL as a REXX program, see IBM Z Workload Scheduler control language on page 19.

What you can do using OCL

Using OCL, you can do the following:

- Through its IF-THEN-ELSE instruction, better control your workload and take action appropriate to the result.
- Dynamically modify the behavior of your batch applications, without changing the databases of the scheduler.
- Complete, delete, add, release, modify, and control occurrences or operations in the scheduler plans.
- Kill jobs in STARTED state on IBM Z Workload Scheduler Agents or on distributed workstations that are directly connected to the end-to-end server (OPCMASTER). Also kill recovery jobs in EXECUTING state on distributed workstations that are directly connected to the end-to-end server (OPCMASTER).
- Dynamically change the dependencies among the operations in the current plan, by adding or deleting predecessor occurrences, and releasing the internal and external successors.
- Release all or some of the external successors of an occurrence.
- Integrate the JSUACT, OPSTAT, SRSTAT, and WSSTAT TSO commands provided by the scheduler that can be issued in accordance with the result of a relational test.
- Update the value of the variables you can use in your jobs.
- Verify or compare a part of a variable value, such as the third character of the value.
- Use a variable value as a condition for scheduling an application.
- Validate a date you supply, then derive from it about 210 variable values, which you can use in your job.

Advantages of OCL

Some of the main advantages of using OCL are as follows:

- It further automates the scheduling process.
- It increases the usability of the PIF.
- It offers more flexibility for managing the workload.

- It facilitates migration from non-IBM scheduling products to IBM Z Workload Scheduler.
- It increases the number of recovery actions offered by the Automatic Job Recovery (AJR) function of the scheduler: the OCL instructions could be executed as part of the recovery actions.

Summary of OCL instructions

The OCL instructions are specified in a data set referenced by the SYSIN DD card. For an example of an OCL job, the EQQYRJCL member of the EQQMISC data set, see Sample job and procedure on page 365.

Here is a brief description of the instructions:

ADD

Inserts applications or an application group into the scheduler current plan or into the long-term plan.

ADDCOND

Adds a condition definition to an operation in the scheduler current plan.

ADDOP

Inserts an operation into an occurrence in the current plan.

ADDPRED

Adds a predecessor dependency definition to an occurrence in the scheduler current plan.

ADDRES

Adds a special resource to an operation in the scheduler current plan.

ADDSIMP

Adds a single dependency definition to a condition in the scheduler current plan.

ADOPSAI

Specifies system automation information for the operation (command text, automated function, security element, and completion information).

CALL

Invokes a REXX procedure.

CHGEXTNAME

Defines or modifies the extended job information associated with the operations of an occurrence in the scheduler current plan. You can also use CHGEXTNAME to delete the extended job information associated with the operations of an occurrence in the scheduler current plan.

CHGJOB

Changes the jobname associated with the operations of an occurrence in the scheduler current plan.

CHGOPSAI

Defines or modifies the system automation information associated with the operations of an occurrence in the scheduler current plan. You can use CHGOPSAI also to delete the system automation information.

CHKAPPL

Verifies the presence of occurrences in the scheduler current plan; when keywords STATUS(E) and ALERT(YES) are set, it checks operations that ended in error and sends to TSO users a message containing the related error information.

CHKDATE

Checks the validity of an input date in the following formats: YYYY/MM/DD, YY/MM/DD, YYYYMMDD, YYMMDD, DD/MM/YYYY, DD/MM/YY, DDMMYYYY, DDMMYY, and assigns to variables values it derives from the input date.

COMPL

Completes occurrences or specific operations of the occurrences in the scheduler current plan.

DEL

Deletes an occurrence or an operation from the scheduler current plan or from the long-term plan.

DELCOND

Deletes a condition from an operation in the scheduler current plan.

DELPRED

Deletes a dependency definition from an occurrence in the scheduler current plan with a predecessor operation.

DELRES

Deletes a special resource from an operation in the scheduler current plan.

DELSIMP

Deletes a single dependency definition from a condition in the scheduler current plan.

EXIT

Sets the program exit return code.

FORCE

Forces the execution of the application.

GOTO

Moves to a point specified by a LABEL statement within the OCL program.

HOLD

Sets the operations of an occurrence to HOLD status.

IF-THEN-ELSE

Performs a relational test and, depending on the result, executes one of the following instructions: SET, UPD, SETUPD, GOTO, CALL, EXIT, CHKAPPL, ADD, ADDOP, DEL, COMPL, ADDPRED, DELPRED, ADDRES, DELRES, CHGJOB, JSUACT, OPSTAT, SRSTAT, WSSTAT, CHKDATE, RELSUCC, RELOP, FORCE, RELEASE, HOLD, NOP, UNNOP, MODOP, or IF-THEN-ELSE.

JSUACT

Activates or inactivates the job submission function in the z/OS® environment, in the distributed environment, or in both environments.

INIT

Specifies the ID of the scheduler variable table in which the job variables are described and specifies the scheduler subsystem ID.

KILLJOB

Stops a job that is already running. Applies only to operations running on IBM Z Workload Scheduler Agents or on distributed workstations that are directly connected to the end-to-end server (OPCMASTER).

KILLREC

Stops a recovery job that is already running. Applies only to operations running on distributed workstations that are directly connected to the end-to-end server (OPCMASTER).

This action can be taken only on recovery jobs that are in the EXECUTING status, so that their operation number is known. The application number is required (the operation number is optional, but if you do not specify it, all the operations in the application are killed) to identify the operation that is to be killed.

LABEL

Defines labels within the OCL program SYSIN.

MODCOND

Modifies the condition defined for an operation in the scheduler current plan.

MODOP

Modifies the operations details.

NOP

Removes an operation from the current plan. When a NOP operation is ready to be started, it is immediately set to C status.

OPSTAT

Changes the status of an operation in the scheduler current plan.

PROMPTN

Specifies that NO is the reply to a recovery prompt issued for an abended operation.

PROMPTY

Specifies that YES is the reply to a recovery prompt issued for an abended operation.

RELEASE

Releases the operations of an occurrence that are in HOLD status.

RELOP

Releases the internal successors of an operation.

RELSUCC

Releases the successors of the occurrence by deleting the external dependency definition.

SET

Sets a user variable.

SETUPD

Sets a user variable and updates its default value in an scheduler variable table. This instruction is equivalent to the SET and UPD instructions together.

SRSTAT

Modifies the availability status of a special resource.

UNNOP

Restores an operation from NOP status.

UPD

Updates the default value of a user variable in an scheduler variable table.

WSSTAT

Modifies the status of a workstation.

WTO

Displays messages on the system console and waits for a reply.

For a full description of all OCL instructions, see Description of OCL instructions on page 251.

If the OCL PARMLIB TSOCMD parameter is set to YES, you can also specify a TSO command or a REXX expression other than those listed here. For example:

```
DO X = 1 TO 3;,
  SAY 'HELLO';,
END
Select;,
  when var1 = 'OK' then ADD APPL(TEST01);,
  when var1 = 'KO' then EXIT 16;,
  otherwise nop;,
End
```

For punctuation rules, see Specifying OCL instructions on page 246.

Customizing OCL

Follow these steps to customize the OCL function.

1. Customize the initialization parameters in the OCL library member EQQYRPRM. The initialization parameters are as follows:

   SUBSYS()

   Default subsystem ID for Z controller. This is overridden by the SUBSYS specified in the EQQYPARM DD card and by the SUBSYS() keyword specified in the OCL instructions (for details, see Specifying input arrival dates and times on page 247).

   OPCTRK()

   Default subsystem ID for tracker.

   LUNAME()

   LU node name of the server communicating with the controller system.

   REMHOST()

   The server host name for the program interface TCP/IP session. REMHOST and LUNAME are mutually exclusive.

   REMPORT()

   The server port number for the program interface TCP/IP session. REMPORT and LUNAME are mutually exclusive.

   DEFCONDID()

   Default condition identifier; for example, 001.

   DEFIAT()

   Default Input Arrival time in *HHMM* format; for example, 1800.

   A blank value is allowed. If DEFIAT is set to blank and an IA time is not specified, the following rules apply:

   - All the occurrences matching the other specified parameters are searched for, by applying the mechanism set by the SORT parameter (MIN or MAX).
   - If the action is an add occurrence, the blank value is replaced by the current time.

   DEFOPNO()

   Default operation number; for example, 001.

   DEFPREOPNO()

   Default external predecessor operation number; for example, 099. This is used only for external dependencies.

   DUBPROC()

   Determines the parameter with which the BPX1SDD routine is to be invoked. It can be Y or N.

   Y

   The BPX1SDD routine is invoked by using the DUBPROCESS parameter.

N

The BPX1SDD routine is invoked by using the DUBPROCESSDEFER parameter.

SORT()

If you do not specify the occurrence Input Arrival date and time in the OCL instructions, SORT() determines which application occurrence in the current plan to consider. SORT is also used when *only* the IA date is set; in this case, after the MAX or MIN criterion is applied, the IA date is used to filter the matching occurrences.

MAX

Determines the occurrence with the latest IA date and time.

MIN

Determines the occurrence with the earliest IA date and time.

TSOCMD()

The external TSO command and REXX instruction activator. It can be YES or NO.

YES

You can specify TSO commands and REXX instructions other than those provided by OCL.

NO

You can specify only OCL instructions.

INCVALUE()

Indicates the increasing value for the variable calculation. OCL will add days to or subtract days from an input date up to the INCVALUE value. It will calculate the new date ±*n* calendar days, where *n* is a value from 1 to the value of INCVALUE, which can be in the range 1-30.

This parameter is used only by CHKDATE instruction.

LOOPDEP()

OCL does not add a group of applications if the group contains applications that are mutually dependent. In this case, the operation fails and message EQQCL7QE is issued. Set LOOPDEP to NO if you do not want OCL to check whether the applications in a group are mutually dependent.

The default is YES.

For information about how to specify initialization parameters, see Specifying the initialization parameters on page 243.

2. If you want to enable the logging function, allocate the log data set. The log data set must be a sequential data set with LRECL 133 and RECFM FB. For information about the logging function, see Logging executed instructions on page 245.
3. Customize all the JCL cards in the OCL library member EQQYRPRC, in accordance with your installation standards.
4. Customize all the JCL cards in the OCL library member EQQYRJCL, in accordance with your installation standards.

5. Specify the OCL instructions in the data set referred to by the SYSIN DD card in the EQQYRJCL member.

   Example:

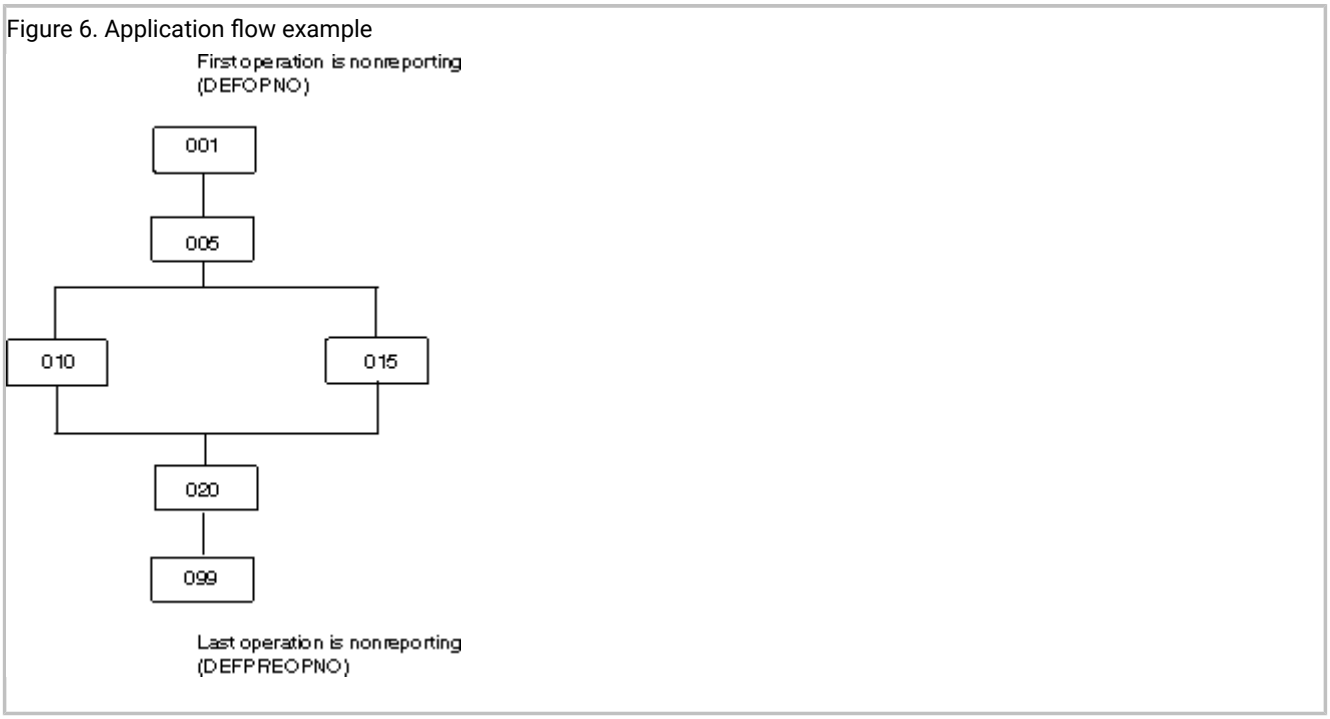
   ```
   //EQQOCL.SYSIN   DD *
   CHKAPPL APPL(TEST*) STATUS(E)
   * …
   ```

6. Submit the job.

Specifying the initialization parameters

You can use the DEFOPNO and DEPREOPNO parameters if your application's first and last operations are nonreporting operations. This is illustrated in Figure 6: Application flow example on page 243.

Figure 6. Application flow example



OCL will use these parameters as default operation numbers.

Example 1

If DEFOPNO is 001 and DEFPREOPNO is 099, the instruction:

```
ADDPRED APPL(TEST02) PA(TEST01)
```

adds a predecessor dependency between operation 099 of application TEST01 and operation 001 of application TEST02.

Example 2

Assuming that the application shown in Figure 6: Application flow example on page 243 is TEST03, the instruction:

```
RELOP APPL(TEST03) OP(5)
```

deletes the dependency between operation 005 and its internal successors (operations 010 and 015) and defines operation 001 (DEFOPNO) as an internal predecessor of these two operations.

Example 3

The DEFIAT parameter is used when you specify the IADATE parameter in the OCL instruction. For example:

```
COMPL APPL(TEST01) IADATE(=)
```

The COMPL instruction will complete the application occurrence TEST01 with an input arrival date equal to the current date and with an input arrival time equal to the DEFIAT parameter value.

Example 4

The SORT parameter specifies whether OCL is to consider the application occurrence with the latest or the earliest input arrival date and time. For example:

```
COMPL APPL(TEST01)
```

The current plan includes the following occurrences of application TEST01:

```
Input arrival date: 97/12/01 time: 18.00
Input arrival date: 97/12/01 time: 20.00
Input arrival date: 97/12/03 time: 18.00
```

If SORT is MAX, OCL selects the occurrence with input arrival 97/12/03 1800 If SORT is MIN, OCL selects the occurrence with input arrival 971201 1800

The SORT parameter is used only if IA, IADATE, and IATIME keywords are not specified in the OCL instruction.

Obtaining access authorization

Many of the OCL instructions use the scheduler programming interface. You need authorization to use the programming interface requests. If you are not authorized to use the instruction you need, give the relevant access type and RACF® resource code to your administrator. Table 123: Access Authorizations on page 244 lists the OCL instructions that access the scheduler resources and the access authorization you need for each.

Note: Use the MODOP instruction as follows:

```
MODOP APPL(applname) IADATE(iadate) IATIME(iatime) OPNO(operationnumber)
```

One of the following parameters must be specified:

```
WSNAME, ASUB, AJR, TIMEDEP, CLATE, OPIA, and OPDL
```

Otherwise warning message EQQCL4VW “No parameter was specified? is issued.

Table 123. Access Authorizations

Instruction	Access Authorization Required
INIT	None
CHKAPPL CHKDATE	Read
ADD ADDCOND ADDOP ADDPRED ADDRES ADDSIMP CHGEXTNAME CHGJOB CHGOPSAI COMPL DEL DELCOND	Update

Table 123. Access Authorizations (continued)

Instruction	Access Authorization Required
DELPRED DELSIMP DELRES FORCE HOLD KILLJOB KILLREC JSUACT MODCOND MODOP NOP OPSTAT RELEASE RELOP RELSUCC SETUPD SRSTAT UNNOP UPD WSSTAT	

OCL libraries can be protected by a security product, such as RACF®, which allows only authorized users can run the OCL program. Because OCL instructions can be issued from within IBM Z Workload Scheduler-scheduled jobs, the product should be authorized to access the OCL libraries.

Use the TSOCMD parameter to avoid having to specify TSO commands or REXX instructions other than those provided by OCL.

Logging executed instructions

For each instruction executed, OCL writes messages in the data sets referred to by the SYSTSPRT DD card and by the OCLLOG DD card. These messages describe the instruction that have been executed by OCL. The data set referred to by the OCLLOG DD card must be allocated with LRECL 133 and RECFM FB and used with disposition MOD.

The logging function requires the OCLLOG DD card to be specified in the OCL procedure. For an example OCL procedure, see EQQYRPRC sample procedure on page 366.

Note: Instructions are logged only if the OCLLOG DD card is specified.

An example of the log records follows:

```
07/30 13:41:33 EQQCL00I Instruction : INIT VARTAB(&OADID)
07/30 13:41:34 EQQCL02I INIT    instruction executed : RC=0
07/30 13:41:34 EQQCL00I Instruction : SETUPD VAR1 = 'PIPPO'
07/30 13:41:34 EQQCL02I SETUPD  instruction executed : RC=0
07/30 13:41:34 EQQCL00I Instruction : COMPL APPL(TEST01)
07/30 13:41:35 EQQCL02E COMPL   instruction executed : RC=8
07/30 13:41:35 EQQCL00I Instruction : CHKAPPL APPL(TEST02)
07/30 13:41:35 EQQCL02W CHKAPPL instruction executed : RC=4
07/30 13:41:35 EQQCL00I Instruction : IF RESULT = 8 THEN NOP
```

An example of the messages written in the SYSTSPRT DD card follows:

```
EQQCL01I ====================================================================
EQQCL00I Processing: CHKAPPL APPL(TEST01) STATUS(S)
EQQCL0JI Searching for the occurrence TEST01 in CP
EQQCL0OI Occurrence found: APPL(TEST01) IA(9709080800) STATUS(S)
EQQCL0OI Occurrence found: APPL(TEST01) IA(9709080930) STATUS(S)
EQQCL0KI Total n. of matching occurrences: 2
EQQCL01I ====================================================================
EQQCL00I Processing: IF RESULT=0 THEN ADD APPL(TEST02)
EQQCL03I True condition: IF RESULT = 0
EQQCL01I ====================================================================
EQQCL00I Processing: ADD APPL(TEST02)
EQQCL0AI The occurrence was successfully added: APPL(TEST02) IA(9709091657
```

Specifying OCL instructions

To correctly specify OCL instructions in your JCL, observe the following rules:

1. When you specify two or more REXX instructions on the same line, separate one from the next by a semicolon followed immediately by one or more blank characters (;). For example:

   ```
        IF VAR1 = 'OK' THEN ADD APPL(TEST01); ELSE GOTO ESCI
   ```

2. Specify all the instructions in column 1 through column 72.
3. If the instruction continues on the next line, end the first line with a comma (the continuation character). For example:

   ```
   IF RESULT > 0 THEN,
     DEL APPL(TESTAPPL2)
   IF SUBSTR(VAR2,3,1) = '0' THEN COMPL APPL(TEST03);,
     ELSE ADD APPL(TEST04)
   ```

4. If you continue a list of keyword values on the next line, end the first line with two consecutive commas („). For example:

   ```
   CHKAPPL APPL(TEST01,TEST02,TEST03,TEST04,,
     TEST05)
   ```

5. If you need to use IBM Z Workload Scheduler-managed variables, specify the JCL tailoring directive (for example, //* %OPC SCAN). These variables will be substituted by the scheduler before it runs the OCL program. For example:

   ```
   //… JOB …
   //*%OPC SCAN
   //…
   //EQQOCL.SYSIN DD *
   if substr(&OYMD1,5,2) = 31 then goto lastdd
   ⋮
   ```

   In this example, the scheduler substitutes &OYMD1 with the occurrence input arrival date in YYMMDD format.
6. Prefix a comment line with an asterisk (*). For example:

   ```
   //EQQOCL.SYSIN DD *
   * This is my comment line
   ⋮
   ```

7. The following instructions store their return code in the variable RESULT:

ADD	ADDOP	ADDPRED	ADDRES	ADDRES
CALL	CHGEXTNAME	CHKAPPL	CHKDTE	COMPL
CHGJOB	CHGOPSAI	COMPL	DEL	DELSIMP
DELPRED	DELRES	FORCE	HOLD	MODCOND
JSUACT	KILLJOB	KILLREC	MODOP	RELSUCC
NOP	OPSTAT	RELEASE	RELOP	
RELSUCC	SRSTAT	UNNOP	WSSTAT	

   For example:

```
CHKAPPL APPL(test01,test02)
IF RESULT = 0 then …
```

8. Do not include delimiters, such as parentheses, in descriptive fields, such as DESC and EXTNAME.

Specifying input arrival dates and times

About this task

In the scheduler, the input arrival time forms part of the key that uniquely identifies each occurrence of the application in the long-term plan and in the current plan; it is not the time that the scheduler attempts to start the occurrence, unless you specify the first operation as time-dependent. The input arrival time of the occurrence is the default input arrival time for operations making up that occurrence. Some of the OCL instructions use the input arrival date and time to uniquely identify the occurrences in the scheduler plans. The keywords that specify the input arrival date and time are:

Table 124. Input Arrival Date and Time Keywords

Keyword	Abbreviation	Purpose
IADATE	IAD	Specifies the input arrival date, in *YYMMDD* format.
IATIME	IAT	Specifies the input arrival time, in *HHMM* format.
IA		Specifies the input arrival date and time, in *YYMMDDHHMM* format.

These keywords are optional. You can specify either or both the keywords IADATE and IATIME, but if you do you cannot then specify the IA keyword. OCL does not check for duplicate keywords in the OCL statements. If a keyword is specified more than once, the values from the last occurrence will be used.

The initialization parameters that affect the calculation of the input arrival date and time if you omit either specification are:

DEFIAT

Specified in the EQQYRPRM member; it determines the default occurrence input arrival time (for example, 1800).

A blank value is allowed. If DEFIAT is set to blank and an IA time is not specified, the following rules apply:

- All the existing occurrences matching the other specified parameters are searched for, by applying the mechanism set by the SORT parameter (MIN or MAX).
- If the action is an add occurrence, the blank value is replaced by the current time.

SORT

Specified in the EQQYRPRM member or in the INIT instruction. It is used only if IADATE, IATIME, and IA keywords are not specified in the OCL instruction, or if IADATE is set and IATIME is not. In this latter case, after the MAX or MIN criterion is applied, the IA date is used to filter the matching occurrences.

The SORT parameter determines whether OCL is to consider the earliest or the latest date and time of the occurrence in the current plan. For example:

```
COMPL APPL(TEST01)
```

The following occurrences of the application TEST01 are in the current plan:

```
Input arrival date: 211201 time: 18.00
Input arrival date: 211201 time: 20.00
Input arrival date: 211203 time: 18.00
```

If SORT is MAX, OCL selects the occurrence with input arrival 21/12/03 18.00. If SORT is MIN, OCL selects the occurrence with input arrival 21/12/01 18.00.

Table 125: How OCL Uses the IADATE, IATIME, and IA Keywords on page 248 describes how OCL uses the IADATE, IATIME, and IA keywords:

Table 125. How OCL Uses the IADATE, IATIME, and IA Keywords

Keywords	Input arrival date and time calculated by OCL	Comment
IAD(971203)	9712031800	Uses the default IA time, as specified in the DEFIAT parameter (for example, 1800). If DEFIAT is not set: • It uses the current time (for example, 1015) if the action is an occurrence add. • Time is left blank and all the occurrences matching the IA date are searched for, by applying the mechanism set by the SORT parameter.
IAD(971203) IAT(2030)	9712032030	
IAD(=)	9712041800	Uses the current date and the default IA time (for example, 1800), if it is specified in the DEFIAT parameter. If DEFIAT is not set: • It uses the current time (for example, 1015) if the action is an occurrence add. • Time is left blank and all the occurrences matching the IA date are searched for, by applying the mechanism set by the SORT parameter.
IAD(=) IAT(=)	9712041015	Uses the current date and time
IAD(971203) IAT=()	9712031015	Uses the specified date and current time.
IAT(=)	9712041015	Uses the current date and specified time.
IAT(2030)	9712042030	Uses the current date.
IA(9712032030)	971232030	
IA(=)	9712031015	Uses the current date and time.

Note: If no keyword is specified, OCL selects the earliest or the latest occurrence of the application in the scheduler plans, according to the SORT parameter specified in the EQQYRPRM member or in the INIT instruction.

If the IA date is set and the IA time is not, after the SORT parameter is applied the IA date is used to filter the matching occurrences.

The keywords listed in Table 125: How OCL Uses the IADATE, IATIME, and IA Keywords on page 248 can be specified in the following OCL instructions:

ADD	ADDCOND	ADDOP	ADDPRED	ADDRES
ADDSIMP	CHGEXTNME	CHGJOB	CHKAPPL	COMPL
DEL	DELCOND	DELPRED	DELRES	DELSIMP
FORCE	HOLD	KILLJOB	KILLREC	MODCOND
MODOP	NOP	RELEASE	RELOP	RELSUCC
UNNOP				

You can use the SUBSYS() keyword to specify the scheduler subsystem name in any of the following:

- The EQQYRPRM member
- The INIT instruction
- The EQQYPARM DD card if it is specified in the OCL procedure

OCL uses the following order to override the scheduler subsystem name:

1. SUBSYS() specified in the data set referenced by the EQQYPARM DD card
2. SUBSYS() specified in an OCL instruction (for example, OPSTAT)
3. SUBSYS() specified in the INIT instruction
4. SUBSYS() specified in the OCL PARMLIB

The SUBSYS() keyword specified in the EQQYPARM overrides all the scheduler subsystem name specifications.

If duplicate keywords are specified in the OCL instructions, OCL takes the rightmost specification. For example:

```
CHKAPPL  APPL(TEST01) IAD(=) APPL(TEST05) STATUS(E)
```

In this example, OCL checks for occurrences of application TEST05.

Substituting variables

About this task

IBM Z Workload Scheduler supports the automatic substitution of variables during job setup and at job submission. IBM Z Workload Scheduler also supplies several standard variables, which you can use in your job. You can create your own variables, which are stored in variable tables in the scheduler database.

The scheduler supplies variables that you can use in the OCL instructions. For example:

```
COMPL APPL(TEST02) IAD(&OYMD1) IAT(&OHHMM)
IF &OHHMM > '1930' THEN ADD APPL(TEST01)
```

The scheduler variables are organized into four groups:

- Occurrence-related variables
- Operation-related variables
- Date-related variables
- Dynamic-format variables

For details on the IBM Z Workload Scheduler-supplied variables, see the section about the variable substitution in *Managing the Workload*.

Table 126: IBM Z Workload Scheduler-supplied Variables on page 250 lists the most common IBM Z Workload Scheduler-supplied variables.

Table 126. IBM Z Workload Scheduler-supplied Variables

Variable	Description
OADID	Application ID
ODD	Occurrence input arrival day of month in DD format
ODDD	Occurrence input arrival day of the year in DDD format
ODMY1	Occurrence input arrival date in DDMMYY format
ODMY2	Occurrence input arrival date in DD/MM/YY format
OHHMM	Occurrence input arrival hour and minute in HHMM format
OMMYY	Occurrence input arrival month and year in MMYY format
OYMD	Occurrence input arrival date in YYYYMMDD format
OYMD1	Occurrence input arrival date in YYMMDD format
OYMD2	Occurrence input arrival date in YY/MM/DD format
OYMD3	Occurrence input arrival date in YYYY/MM/DD format
OJOBNAME	Operation job name
OOPNO	Operation number within the occurrence, right-justified and padded with zeros
OWSID	Workstation ID for current operation

Table 126. IBM Z Workload Scheduler-supplied Variables (continued)

Variable	Description
CDD	Current day of month in DD format
CDDD	Day number in the current year
CHHMM	Current hour and minute in HHMM format
CHHYY	Current month within year in MMYY format
CYMD	Current date in YYYYMMDD format
CYYMMDD	Current date in YYMMDD format

Description of OCL instructions

About this task

This section described each OCL instruction.

ADD

The ADD instruction adds occurrences or a group of occurrences to the current plan or to the long-term plan. It modifies the current plan or long-term plan, depending on the date and time specified or defaulted to. If the date and time being used is after the end of the current plan, the modification will be made to the long-term plan instead of the current plan.

Table 127: Keywords Used in the Add Instruction on page 251 describes the keywords that can be used.

Table 127. Keywords Used in the Add Instruction

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
APPL()	Required[1]	Yes	Name of the application to be added. APPL() and GROUP() are mutually exclusive.		
DL()	Optional	No	Deadline date and time of the application occurrence or group.		YYMMDDHHMM
GDEPRES()	Optional	No	Group dependencies resolution option (Y or N). If not specified it defaults to "Y". To be used only when the versioning is present.		

Table 127. Keywords Used in the Add Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
GROUP()	Required[1]	No	The application group name to be added. GROUP() and APPL() are mutually exclusive.		
IADATE()	Optional	No	Input arrival date of the application occurrence. If not specified, it defaults to the current date. IADATE() and IA() are mutually exclusive.	IAD	YYMMDD
IATIME()	Optional[1]	No	Input arrival time of the application occurrence. It defaults to the current time or, if an occurrence of the application already exists in the current plan, to the first minute after the current time. IATIME() and IA() are mutually exclusive.	IAT	HHMM
IA()	Optional[1]	No	Input arrival date and time of the application occurrence or group. It defaults to the current date and time. IA() cannot be used with IADATE() or IATIME().		YYMMDDHHMM
VARTAB()	Optional	No	The IBM Z Workload Scheduler variable table you want to associate with the group or application.		
Note: [1] Mutually exclusive with another keyword.					

Examples:

ADD APPL(TEST01)

Application TEST01 will be added to the current plan or to the long-term plan

ADD GROUP(GROUPTEST01)

All the applications associated with the application group GROUPTEST01 will be added to the current plan or to the long-term plan

ADD APPL(TEST01) IA(=)

Application TEST01 will be added to the current plan or to the long-term plan with input arrival date and time that corresponds to the current date and time

ADD APPL(TEST01) IAD(970708) IAT(0900)

Application TEST01 will be added to the current plan or to the long-term plan with input arrival date 970708 and input arrival time 0900

ADD APPL(TEST01,TEST02,TEST03,TEST04,TEST05,TEST06)

Applications TEST01, TEST02, TEST03, TEST04, TEST05, and TEST06 will be added to the current plan or to the long-term plan

 **Note:** The following message is not to be considered an error by OCL:

EQQM018E OCCURRENCE ALREADY EXISTS - NOT ADDED

It is issued when OCL tries to add an occurrence that already exists in the current plan. In this case, OCL tries again to add the occurrence with an input arrival time incremented by 1 minute.

The ADD instruction returns one of the following return codes:

RESULT = 0

Occurrences added to the current plan or to the long-term plan.

RESULT = 4

Occurrences added to the long-term plan, but unable to resolve all external dependencies.

RESULT = 8

Invalid instruction or PIF problem. Refer to the error messages.

ADDCOND

The ADDCOND instruction adds a condition to an operation in the current plan. It modifies the current plan, depending on the date and time specified or defaulted to.

Table 128: Keywords used in the Addcond Instruction on page 254 describes the keywords that can be used.

Table 128. Keywords used in the Addcond Instruction

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
APPL()	Required	No	The name of the application to be modified.		
OPNO()	Optional	No	The number of the operation to be modified. It defaults to the default operation number specified in the OCL program DEFOPNO.	OP	
IADATE()	Optional[1]	No	The input arrival date of the application occurrence. IADATE(=) assumes the current date. IADATE() and IA() are mutually exclusive.	IAD	YYMMDD
IATIME()	Optional[1]	No	The input arrival time of the application occurrence. If IADATE(=) is specified, IATIME() assumes the default input arrival time specified in the DEFIAT parameter of the EQQYRPRM member. IATIME() and IA() are mutually exclusive.	IAT	HHMM
IA()	Optional	No	The input arrival date and time of the application occurrence. IA(=) assumes the current date and time. It defaults to the earliest or latest input arrival date and time of the application occurrence. occurrence found in the current plan,		YYMMDDHHMM

Table 128. Keywords used in the Addcond Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
			depending on the SORT parameter specified in PARMLIB or in the INIT instruction. IA() cannot be used with IADATE() or IATIME().		
PREOPNO()	Optional	No	The predecessor operation number. It defaults to the default predecessor operation number specified in the DEFPREOPNO parameter of the EQQYRPRM member.	PO	
PREAPPL()	Optional	No	The predecessor application name. It defaults to the application name specified in the application occurrence.	PA	
PREIADATE()	Optional	No	The predecessor application input arrival date. It defaults to the date specified in IADATE().	PIAD	YYMMDD
PREIATIME()	Optional	No	The predecessor application time. It defaults to the time specified in IATIME().	PIAT	HHMM
CONDID()	Optional	No	The number of the condition to be inserted. It defaults to the default condition id specified in the OCL program DEFCONDID.		

Table 128. Keywords used in the Addcond Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
COUNT()	Optional	No	Condition counter. Use it to define the rule type: 0 = All the condition dependencies in this condition must be true *n>0* = At least *n* out of the condition dependencies in this condition must be true The default is 0.		
DESC()	Optional	No	Descriptive text.		
CHKTYPE()	Required	No	Check type. Possible values are: **RC** Return code **ST** Status The default is ST.		
LOG()	Required	No	Logical operator: GE = Greater than or equal to. Valid only for RC check type. GT = Greater than. Valid only for RC check type.		

Table 128. Keywords used in the Addcond Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
			LE = Less than or equal to. Valid only for RC check type. LT = Less than. Valid only for RC check type. EQ = Equal to. NE = Not equal to. Use it to specify conditions on final statuses only. RG = Range. The default is EQ.		
VALRC()	Required	No	Return code, valid only for RC check type. The default is 0000.		
VALRC2()	Required	No	Return code, valid only for RC check type, as second boundary in a range expressed by the RG logical operator. The default is 9999.		
VALST()	Required	No	Condition status, Valid only for ST check type. The default is C.		

Note: [1] Mutually exclusive with another keyword.

The ADDCOND instruction returns one of the following return codes:

RESULT = 0

Predecessor dependency definition added.

RESULT = 4

Predecessor dependency not found in the current plan.

RESULT = 8

Invalid instruction or PIF problem. Refer to the error messages.

ADDOP

The ADDOP instruction adds an operation into an occurrence in the current plan.

Table 129: Keywords used in the Addop Instruction on page 258 describes the keywords that can be used.

Table 129. Keywords used in the Addop Instruction

Keyword	Requirement	List of Values Allowed	Description	Abbrevi-ation	Format
APPL()	Required	No	The name of the application to be modified.		
OPNO()	Required	No	The operation to be added.	OP	
WSNAME()	Required	No	The name of the workstation on which the added operation is executed.	WS	
ASUB()	Optional	No	Automatic job submission option (Y or N).		
AUTFUNC	Optional	No	Automated function for the system automation integration. It can be up to 8 characters.		
CLNTYPE()	Optional	No	Cleanup type: **A** Automatic **I** Immediate **M** Manual **N** None The default is N.		

Table 129. Keywords used in the Addop Instruction (continued)

Keyword	Requirement	List of Values Allowed	Description	Abbrevi-ation	Format
COMMTEXT	Optional[2]	No	Command text for the system automation integration. It can be up to 255 characters.		
COMPINFO	Optional	No	Completion information for the system automation integration. It can be up to 64 characters.		
CONDRJOB()	Optional	No	Specifies if the operation might recover a conditional predecessor (Y or N). The default is N.		
DESC()	Optional	No	Operation descriptive text. If you do not specify DESC(), it defaults to: '* OCL added this oper *'		
DURATION()	Optional[1]	No	The operation estimated duration. EDUR() and DURATION are mutually exclusive.		HHMMSS
EDUR()	Optional[1]	No	The operation estimated duration. EDUR() and DURATION are mutually exclusive.		HHMM
EXPJCL	Optional	No	Expanded JCL used (Y or N). If not specified the default is N.		
EXTNAME()	Optional	No	A free-format name for the operation. It can include blanks and special characters for a maximum of 54 characters. Do not include parentheses in the extended name value.		
EXTSE()	Optional	No	The scheduling environment name. The maximum length can be 16 characters.		

Table 129. Keywords used in the Addop Instruction (continued)

Keyword	Requirement	List of Values Allowed	Description	Abbrevi-ation	Format
IADATE()	Optional[1]	No	The input arrival date of the application occurrence. IADATE() and IA() are mutually exclusive.	IAD	YYMMDD
IATIME()	Optional[1]	No	The input arrival time of the application occurrence. IATIME() and IA() are mutually exclusive.	IAT	HHMM
IA()	Optional[1]	No	The input arrival date and time of the application occurrence. It defaults to the earliest or latest input arrival date and time of the application occurrence occurrence found in the current plan, depending on the SORT parameter specified in PARMLIB or in the INIT instruction. IA() cannot be used with IADATE() or IATIME().		YYMMDDHHMM
JOBCRT()	Optional	No	Specifies whether the operation is critical or eligible for WLM assistance, if late. Possible values are: **P** Critical path. **W** Eligible for WLM assistance. **N** Not eligible for WLM assistance. This is the default.		

Table 129. Keywords used in the Addop Instruction (continued)

Keyword	Requirement	List of Values Allowed	Description	Abbrevi-ation	Format
JOBNAME()	Optional	No	The jobname. If the workstation is a job setup, printer, or computer workstation, the jobname is required.	JOB	
JOBPOL()	Optional	No	Specifies the WLM assistance policy to apply, if the job was defined as critical. Possible values are: **D** Deadline. The job is assisted if it has not completed at deadline time. **L** Long Duration. The job is assisted if it runs beyond the estimated duration. **S** Latest Start Time. The job is assisted if it was submitted before its latest start time. **C** Conditional. An algorithm is used to decide whether to apply the Deadline or		

Table 129. Keywords used in the Addop Instruction (continued)

Keyword	Requirement	List of Values Allowed	Description	Abbrevi-ation	Format
			Latest Start Time policy. **blank** The policy set in the OPCOPTS statement is applied.		
MONITOR	Optional	No	Specifies if the operation is monitored by an external product (Y or N).		
OPDL()	Optional	No	Operation deadline.		YYMMDDHHMM
OPIA()	Optional	No	Operation input arrival time.		YYMMDDHHMM
PREOPNO()	Optional	No	The internal predecessor of the added operation. It defaults to the default operation number specified in the DEFOPNO initialization parameter.	PO	
SECELEM	Optional	No	Security element for the system automation integration. It can be up to 8 characters.		
TIMEDEP()	Optional	No	Time-dependent option.		
USRSYS()	Optional	No	User sysout needed (Y or N). If not specified the default is N.		
WLMSCLS	Optional	No	The WLM service class.		
[1] Mutually exclusive with another keyword. [2] Required for automation workstations.					

Examples:

ADDOP APPL(TEST01) OP(30) PO(20) WS(CPU1) JOB(JOB11111)

Operation 020 will be added to the occurrence TEST01 with the latest or earliest input arrival date and time (depending on the RT parameter specified in PARMLIB or in the INIT instruction)

ADDOP APPL(TEST01) OP(30) WS(CPU1) JOB(JOB11111) TIMEDEP(Y), OPIA(&OYMD1.2000) OPDL(&OYMD1.2300)

Operation 30 will be added as a successor of the default operation number (for example, 001) and it will be scheduled at 08.00 p.m.

ADDOP APPL(TEST01) WS(FTW1) IA(0310160000) DESC(EXTNAME) PREOPNO(001) JOBNAME(TEST) OPNO(002) EXTNAME(DAILY PAYROLL JOB)

Operation 002 will be added to occurrence TEST01 with Extended Name DAILY PAYROLL JOB.

ADDOP APPL(TEST01) WS(FTW1) IA(0310160000) DESC(Extname Blank) PREOPNO(001) JOBNAME(TEST) OPNO(002) EXTNAME( )

Operation 002 will be added to occurrence TEST01 with Extended Name *blank* .

The EXTNAME keyword is mandatory when the following two conditions coexist:

- When you add an operation that is to be inserted into the Symphony file.
- When the TWSJOBNAME value is EXTNAME or EXTNOCC.

Note: You cannot add a Job setup operation.

The ADDOP instruction returns one of the following return codes:

RESULT = 0

Operation was successfully added to the occurrence.

RESULT = 4

Not possible to add the operation.

RESULT = 8

Invalid instruction or PIF problem. Refer to the error messages.

ADDPRED

The ADDPRED instruction adds a predecessor operation to an occurrence in the current plan or in the long-term plan. It modifies the current plan or long-term plan, depending on the date and time specified or defaulted to. If the date and time being used is after the end of the current plan, then the modification will be made to the long-term plan instead of the current plan.

Table 130: Keywords used in the Addpred Instruction on page 264 describes the keywords that can be used.

Table 130. Keywords used in the Addpred Instruction

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
APPL()	Required	No	The name of the application to be modified.		
OPNO()	Optional	No	The number of the operation to be modified. It defaults to the default operation number specified in the OCL program DEFOPNO.	OP	
IADATE()	Optional[1]	No	The input arrival date of the application occurrence. IADATE(=) assumes the current date. IADATE() and IA() are mutually exclusive.	IAD	YYMMDD
IATIME()	Optional[1]	No	The input arrival time of the application occurrence. If IADATE(=) is specified, IATIME() assumes the default input arrival time specified in the DEFIAT parameter of the EQQYRPRM member. IATIME() and IA() are mutually exclusive.	IAT	HHMM
IA()	Optional	No	The input arrival date and time of the application occurrence. IA(=) assumes the current date and time. It defaults to the earliest or latest input arrival date and time of the application occurrence. occurrence found in the current plan,		YYMMDDHHMM

Table 130. Keywords used in the Addpred Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
			depending on the SORT parameter specified in PARMLIB or in the INIT instruction. IA() cannot be used with IADATE() or IATIME().		
PREOPNO()	Optional	No	The predecessor operation number. It defaults to the default predecessor operation number specified in the DEFPREOPNO parameter of the EQQYRPRM member.	PO	
PREAPPL()	Optional	No	The predecessor application name. It defaults to the application name specified in the application occurrence.	PA	
PREIADATE()	Optional	No	The predecessor application input arrival date. It defaults to the date specified in IADATE().	PIAD	YYMMDD
PREIATIME()	Optional	No	The predecessor application time. It defaults to the time specified in IATIME().	PIAT	HHMM

Note: [1] Mutually exclusive with another keyword.

Examples:

ADDPRED APPL(TEST01) OP(40) PA(TEST00)

The default predecessor operation, DEFPREOPNO, of the application TEST00 will be added as a predecessor of the operation 40 of the application TEST01.

ADDPRED APPL(TEST01) IAD(970708) OP(40) PO(90), PA(TEST00)

The operation 90 of the application TEST00 will be added as a predecessor of the operation 40 of application TEST01 with the input arrival date 970708 and the input arrival time that corresponds to the default IA time, DEFIAT.

ADDPRED APPL(TEST01)

The default predecessor operation, DEFPREOPNO, of the application TEST01 will be added as an external predecessor of the default operation number, DEFOPNO, of the same application.

The ADDPRED instruction returns one of the following return codes:

RESULT = 0

Predecessor dependency definition added.

RESULT = 4

Operation not found in the current plan.

RESULT = 8

Invalid instruction or PIF problem. Refer to the error messages.

ADDRES

The ADDRES instruction adds a special resource to an operation in the current plan.

Table 131: Keywords used in the Addres Instruction on page 266 describes the keywords that can be used.

Table 131. Keywords used in the Addres Instruction

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
APPL()	Required	No	The name of the application to be modified.		
OPNO()	Optional	No	The number of the operation to be modified. It defaults to the default operation number specified in the DEFOPNO parameter of the EQQYRPRM member.	OP	
IADATE()	Optional[1]	No	The input arrival date of the application occurrence. IADATE=()	IAD	YYMMDD

Table 131. Keywords used in the Addres Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
			means the current date. IADATE() and IA() are mutually exclusive.		
IATIME()	Optional[1]	No	The input arrival time of the application occurrence. If IADATE(=) is specified, IATIME() assumes the default input arrival time that is defined in the DEFIAT parameter of the EQQYRPRM member. IATIME() and IA() are mutually exclusive.	IAT	HHMM
IA()	Optional[1]	No	The input arrival date and time of the application occurrence. IA(=) assumes the current date and time. It defaults to the earliest or latest input arrival date and time of the application occurrences found in the current plan, depending on the SORT parameter specified in the EQQYRPRM member in the INIT instruction. IA() cannot be used with IADATE() or IATIME().		YYMMDDHHMM

Table 131. Keywords used in the Addres Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
ONERROR	Optional	No	Keep on error: YES, NO, or blank	OE	
ONCOMPL	Optional	No	On complete action. Change resource availability: YES, NO, RESET, or blank	ONC	
RESNAME()	Required	No	The special resource to be added.	RN	
RESUSAGE()	Required	No	The special resource usage: **S** Shared **X** Exclusive	RU	
ONERROR()	Required	No	The ONERROR option: YES to keep the special resource, NO ot free it, in case of operation ended in error.		
Note: [1] Mutually exclusive with another keyword.					

Examples:

ADDRES APPL(TEST01) RN(prova.sr1) RU(X) OE (N)

Adds the special resource PROVA.SR1 with exclusive use to the default operation number, DEFOPNO, of the occurrence TEST01 and KEEP-ON-ERROR set to no.

ADDRES APPL(TEST02) IAD(&OYMD1) RN(prova.sr2) RU(X) OP(10), OE(Y)

Adds the special resource PROVA.SR2, with exclusive use and keep-on-error set to YES, to operation 10 of the occurrence TEST01 with the default input arrival time, DEFIAT

The ADDRES instruction returns one of the following return codes:

RESULT = 0

Special resource added.

RESULT = 4

Occurrence or operation not found in the current plan.

RESULT = 8

Invalid instruction or PIF problem. Refer to the error messages.

ADDSIMP

The ADDSIMP instruction adds condition dependencies to an operation in the current plan. It modifies the current plan, depending on the date and time specified or defaulted to.

Table 132: Keywords used in the Addsimp Instruction on page 269 describes the keywords that can be used.

Table 132. Keywords used in the Addsimp Instruction

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
APPL()	Required	No	The name of the application to be modified.		
OPNO()	Optional	No	The number of the operation to be modified. It defaults to the default operation number specified in the OCL program DEFOPNO.	OP	
IADATE()	Optional[1]	No	The input arrival date of the application occurrence. IADATE(=) assumes the current date. IADATE() and IA() are mutually exclusive.	IAD	YYMMDD
IATIME()	Optional[1]	No	The input arrival time of the application occurrence. If IADATE(=) is specified, IATIME() assumes the default input arrival time specified in the DEFIAT parameter of the EQQYRPRM member.	IAT	HHMM

Table 132. Keywords used in the Addsimp Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
			IATIME() and IA() are mutually exclusive.		
IA()	Optional	No	The input arrival date and time of the application occurrence. IA(=) assumes the current date and time. It defaults to the earliest or latest input arrival date and time of the application occurrence. occurrence found in the current plan, depending on the SORT parameter specified in PARMLIB or in the INIT instruction. IA() cannot be used with IADATE() or IATIME().		YYMMDDHHMM
PREOPNO()	Optional	No	The predecessor operation number. It defaults to the default predecessor operation number specified in the DEFPREOPNO parameter of the EQQYRPRM member.	PO	
PREAPPL()	Optional	No	The predecessor application name. It defaults to the application name specified in the application occurrence.	PA	
PREIADATE()	Optional	No	The predecessor application input arrival date. It defaults to	PIAD	YYMMDD

Table 132. Keywords used in the Addsimp Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
			the date specified in IADATE().		
PREIATIME()	Optional	No	The predecessor application time. It defaults to the time specified in IATIME().	PIAT	HHMM
CONDID()	Optional	No	The number of the condition to be inserted. It defaults to the default condition id specified in the OCL program DEFCONDID.		
CHKTYPE()	Optional	No	Check type. Possible values are: **RC** Return code **ST** Status The default is ST.		
LOG()	Optional	No	Logical operator: GE = Greater than or equal to. Valid only for RC check type. GT = Greater than. Valid only for RC check type. LE = Less than or equal to. Valid only for RC check type.		

Table 132. Keywords used in the Addsimp Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
			LT = Less than. Valid only for RC check type. EQ = Equal to. NE = Not equal to. Use it to specify conditions on final statuses only. RG = Range. The default is EQ.		
VALRC()	Optional	No	Return code, valid only for RC check type. The default is 0000.		
VALRC2()	Optional	No	Return code, valid only for RC check type, as second boundary in a range expressed by the RG logical operator. The default is 9999.		
VALST()	Optional	No	Condition status, Valid only for ST check type. The default is C.		
Note: [1] Mutually exclusive with another keyword.					

The ADDSIMP instruction returns one of the following return codes:

RESULT = 0

Predecessor dependency definition added.

RESULT = 4

Predecessor dependency not found in the current plan.

RESULT = 8

Invalid instruction or PIF problem. Refer to the error messages.

CALL

The CALL instruction invokes a routine. A result returned by the routine is stored in the variable RESULT. The meaning of the RESULT variable value depends on the routine.

Syntax

CALL *routineparameters*

Examples:

```
CALL CHKDATA &DATACONT.
CALL test1
IF RESULT > 0 THEN EXIT 99
```

CHGEXTNAME

The CHGEXTNAME instruction defines or modifies the extended job information associated to the operations of an application occurrence. You can delete the extended name by specifying `EXTNAME()` and `EXTSE()`.

For a description of the keywords that can be used, see Table 133: Keywords used in the Chgextname Instructions on page 273.

Table 133. Keywords used in the Chgextname Instructions

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
APPL()	Required	No	The name of the application to be modified.		
OPNO()	Required	No	The number of the operation to be modified.	OP	
IADATE()	Optional[1]	No	The input arrival date of the application occurrence. IADATE(=) assumes the current date. IADATE() and IA() are mutually exclusive.	IAD	YYMMDD
IATIME	Optional[1]	No	The input arrival time of the application occurrence. If IADATE(=) is specified, IATIME() assumes that the default input arrival	IAT	HHMM

Table 133. Keywords used in the Chgextname Instructions (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
			time is specified in the DEFIAT parameter of the EQQYRPRM member. IADATE() and IA() are mutually exclusive.		
IA()	Optional[1]	No	The input arrival date and time of the application occurrence. IA(=) assumes the current date and time. It defaults to the earliest or latest input arrival date and time of the application occurrences found in the current plan, depending on the SORT parameter. IA() cannot be used together with IADATE() or IATIME().		YYMMDDHHMM
EXTNAME()	Required if EXTSE not specified	No	A free-format name for the operation. It can include blanks and special characters. Do not include parentheses in the extended name value.		
EXTSE()	Required if EXTNAME not specified	No	Scheduling Environment name.		

Note: [1] Mutually exclusive with another keyword.

Examples:

CHGEXTNAME APPL(TEST01) EXTNAME('DAILY PAYROLL JOB')

This example sets the extended name DAILY PAYROLL JOB for the operations of TEST01 application.

CHGEXTNAME APPL(TEST01) EXTNAME()

This example deletes the extended name DAILY PAYROLL JOB for the operations of TEST01 application.

Do not use CHGEXTNAME to add the extended name to an operation that is to be included in the Symphony file, when the value of TWSJOBNAME is either EXTNAME or EXTNOCC. Use the ADDOP instruction with the EXTNAME keyword instead.

The CHGEXTNAME instruction returns one of the following return codes:

RESULT = 0

Occurrence modified.

RESULT = 4

Occurrence or operations not found in the current plan.

RESULT = 8

Invalid instruction or PIF problem. Refer to the error messages.

CHGJOB

The CHGJOB instruction changes the job name associated with the operations of an application occurrence.

This instruction applies only to jobs running in z/OS® environments.

For a description of the keywords that can be used, see Table 134: Keywords used in the Chgjob Instructions on page 275.

Table 134. Keywords used in the Chgjob Instructions

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
APPL()	Required	No	The name of the application to be modified.		
OPNO()	Optional	Yes	The number of the operation to be modified.	OP	
IADATE()	Optional[1]	No	The input arrival date of the application occurrence. IADATE(=) assumes the current date. IADATE() and IA() are mutually exclusive.	IAD	YYMMDD
IATIME	Optional[1]	No	The input arrival time of the application occurrence. If IADATE(=) is specified, IATIME()	IAT	HHMM

Table 134. Keywords used in the Chgjob Instructions (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
			assumes that the default input arrival time is specified in the DEFIAT parameter of the EQQYRPRM member. IADATE() and IA() are mutually exclusive.		
IA()	Optional[1]	No	The input arrival date and time of the application occurrence. IA(=) assumes the current date and time. It defaults to the earliest or latest input arrival date and time of the application occurrences found in the current plan, depending on the SORT parameter. IA() cannot be used together with IADATE() or IATIME().		YYMMDDHHMM
OLDJOB()	Optional	No	The old jobname	OLDJ	
NEWJOB()	Required	No	The new jobname	NEWJ	
WSNAME()	Optional	No	The operation workstation name	WS	
Note: [1] Mutually exclusive with another keyword.					

Note: If you omit the selection parameters OPNO(), OLDJOB(), or WSNAME(), OCL will change the jobname of all the operations in the occurrence. You cannot change the jobname of a job setup operation without changing the jobname of the dependent computer operation. In this case, do not use the OPNO() or WSNAME() keywords for applications that contain job setup operations.

Examples:

CHGJOB APPL(TEST01) OLDJ(job11111) NEWJ(job22222)

Changes the jobname to all the operations with JOB11111

CHGJOB APPL(TEST01) IAD(&OYMD1), NEWJ(job22222) OP(20)

Changes the jobname to operation 20

CHGJOB APPL(TEST01) IA(=) NEWJ(job22222) WS(CPU1)

Changes the jobname to all the operations at the workstation CPU1

CHGJOB APPL(TEST01) IAD(970708) IAT(1800), NEWJ(job22222) WS(CPU1) OP(20)

Changes the jobname to operation 20 at the workstation CPU1

The CHGJOB instruction returns one of the following return codes:

RESULT = 0

Occurrence modified.

RESULT = 4

Occurrence or operations not found in the current plan.

RESULT = 8

Invalid instruction or PIF problem. Refer to the error messages.

CHGOPSAI

The CHGOPSAI instruction defines or modifies the system automation information associated with the operations of an application occurrence. The COMMTEXT keyword always must be specified; you can disable AUTFUNC, COMPINFO, and SECELEM by setting them to blank.

For a description of the keywords that can be used, see Table 135: Keywords used in the Chgopsai Instructions on page 277.

Table 135. Keywords used in the Chgopsai Instructions

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
APPL()	Required	No	The name of the application to be modified.		
OPNO()	Required	No	The number of the operation to be modified.	OP	
AUTFUNC	Optional	No	Automated function for the system automation		

Table 135. Keywords used in the Chgopsai Instructions (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
			integration. It can be up to 8 characters.		
COMMTEXT	Optional[2]	No	Command text for the system automation integration. It can be up to 255 characters.		
COMPINFO	Optional	No	Completion information for the system automation integration. It can be up to 64 characters.		
IA()	Optional[1]	No	The input arrival date and time of the application occurrence. IA(=) assumes the current date and time. It defaults to the earliest or latest input arrival date and time of the application occurrences found in the current plan, depending on the SORT parameter. IA() cannot be used together with IADATE() or IATIME().		YYMMDDHHMM
IADATE()	Optional[1]	No	The input arrival date of the application occurrence. IADATE(=) assumes the current date. IADATE() and IA() are mutually exclusive.	IAD	YYMMDD
IATIME	Optional[1]	No	The input arrival time of the application occurrence. If IADATE(=) is specified, IATIME()	IAT	HHMM

Table 135. Keywords used in the Chgopsai Instructions (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
			assumes that the default input arrival time is specified in the DEFIAT parameter of the EQQYRPRM member. IADATE() and IA() are mutually exclusive.		
SECELEM	Optional	No	Security element for the system automation integration. It can be up to 8 characters.		
[1] Mutually exclusive with another keyword. [2] Required, if not previously specified.					

Examples:

CHGOPSAI APPL(TEST01) COMMTEXT(DAILY PAYROLL JOB) SECELEM(AAA)

Sets the command text and security element for the operations of application TEST01

CHGOPSAI APPL(TEST01) AUTFUNC()

Deletes the automated function for the operations of application TEST01

The CHGOPSAI instruction returns one of the following return codes:

RESULT = 0

Occurrence modified.

RESULT = 4

Occurrence or operations not found in the current plan.

RESULT = 8

Invalid instruction or PIF problem. Refer to the error messages.

CHKAPPL

Syntax

CHKAPPL *keyword*

The CHKAPPL instruction verifies the presence of application occurrences in the current plan. Use the STATUS keyword to verifies the application occurrences in a specific status.

If you specify the STATUS keyword as E (Error), you can send a message reporting the operation error code to TSO users.

For a description of the keywords that can be used, see Table 136: Keywords used in the Chkappl Instruction on page 280.

Table 136. Keywords used in the Chkappl Instruction

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
APPL()	Required	Yes	The name of the application to be checked. Wild characters are accepted, for example: APPL(TEST*) or APPL(T%%T*)		
IADATE()	Optional (1)	No	The input arrival date of the application occurrence. IADATE() means the current date. IADATE() and IA() are mutually exclusive.	IAD	YYMMDD
IATIME()	Optional (1)	No	The input arrival time of the application occurrence. If IADATE(=) is specified, IATIME() assumes the default input arrival time that is defined in the DEFIAT parameter. IATIME() and IA() are mutually exclusive.	IAT	HHMM
IA()	Optional (1)	No	The input arrival date time of the application occurrence. IA(=) assumes the current date and time. IA() cannot be used with IADATE() or IATIME().		YYMMDDHHMM

Table 136. Keywords used in the Chkappl Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
STATUS()	Optional	No	The occurrence status can be: **C** Complete **E** An operation has ended in error **D** Deleted **P** A pending predecessor exists for the occurrence **S** Started **U** Undecided **W** No operations in the occurrence have started	ST	
OPNO()	Optional	Yes	The operation number. This keyword has effect only with STATUS(E).	OP	
ALERT()	Optional	No	The alert option (YES or NO):		

Table 136. Keywords used in the Chkappl Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
			ALERT(YES) A message will be sent to the TSO users specified in the USER() keyword. YES is the default. **ALERT(NO)** No message will be sent. This keyword has effect only with STATUS(E).		
USER()	Optional	Yes	The users who are to receive the alert message. This keyword is required only with ALERT(YES).		
ERRCODE()	Optional	Yes	The operation code can be: CAN CCUN FAIL JCCE JCL JCLI MCP OF*xx* (*xx* is the status and extended status) OAUT OJCV	ERR	

Table 136. Keywords used in the Chkappl Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
			OSEQ OSUB OSUF OSUP OS*xx* (*xx* is the status and extended status) PCAN *nnnn* (step return code) S*xxx* (system abend code) U*xxx* (user abend code in hexadecimal notation) *xxxx* (user-defined error code) *yyyy* (distributed agent error code This keyword has effect only with STATUS(E)		
Note: 1 Mutually exclusive with another keyword.					

Examples:

CHKAPPL APPL(TEST01)

All the occurrences of application TEST01

CHKAPPL IAD(&OYMD1) APPL(TEST02)

The occurrences of application TEST02 with the input arrival date of the occurrence that executes the OCL instructions and with input arrival time corresponding to the current time

CHKAPPL IA(=) APPL(TEST02)

The occurrences of application TEST02 with the input arrival date and time corresponding to the current date and time

CHKAPPL APPL(TEST02) IAT(1830)

All the occurrences of application TEST02 with the input arrival time 18.30 and with the input arrival date corresponding to the current date

CHKAPPL APPL(TEST01) STATUS(E) ALERT(YES) USER(ibm001,ibm002)

All the occurrences of application TEST02 in Error status; a message will be sent to TSO users IBM001 and IBM002

CHKAPPL APPL(TEST01) STATUS(E) USER(ibm001,ibm002), ERR(JCL,JCLI) OP(10,15,20)

Checks whether operation 10, 15, or 20 ended with error code JCL or JCLI; if any did, a message will be sent to users IBM001 and IBM002

Note: It is not possible to specify a variable (which is not an IBM Z Workload Scheduler variable) in the IA, IAT and IAD parameters of the OCL CHAPPL instruction, as in the following example:

```
SET VAR4=9912241830
CHKAPPL APPL(TEST01) IA(&VAR4)
```

CHKAPPL returns one of the following return codes:

RESULT = 0

Occurrences found in the current plan.

RESULT = 4

Occurrences not found or no operations that ended in error were found in the current plan.

RESULT = 8

Invalid instruction or PIF problem. Refer to the error messages.

CHKDATE

The CHKDATE instruction verifies the validity of an input date and derives variables from the input date. It accepts dates in the following formats:

Keyword

Formats

DATE1()

YYYYMMDD, YYMMDD, YYYY/MM/DD, and YY/MM/DD

DATE2()

DDMMYYYY, DDMMYY, DD/MM/YYYY, and DD/MM/YY

If the date is valid, the CHKDATE instruction provides the value of the following variables that are based on the input date. The value reported in the table is calculated with an input date equal to 1997/09/28.

Table 137. Chkdate Instruction Variables

Variable	Description	Value
XYMD	Date in YYYYMMDD format	19970928
XYMD1	Date in YYMMDD format	970928
XYMD2	Date in YY/MM/DD format	97/09/28
XYMD3	Date in YYYY/MM/DD format	1997/09/28
XDMY	Date in DDMMYYYY format	28091997
XDMY1	Date in DDMMYY format	280997
XDMY2	Date in DD/MM/YY format	28/09/97
XDMY3	Date in DD/MM/YYYY format	28/09/1997
XDAY	Day of the week: 1–7; 1 = Monday, 2 = Tuesday…	7
XDAYD	Day of the week (MON, TUE, WED, THU, FRI, SAT, or SUN)	SUNDAY
XDD	Day of the month in DD format	28
XDDD	Day of the year in DDD format (Julian day)	271
XDDDYY	Date as a Julian date in DDYYYY format	27197
XDDDYYYY	Date as a Julian date in DDDYYYY format	2711997
XDE®	Ten-day period in the year	28
XFDAY	First day of the month: 1 = Monday, 2 = Tuesday…	1
XFDAYD	First day of the month	MONDAY
XFDAYJ	First day of the month in DDD format (Julian day)	244
XFDAYJ1	First day of the month in YYYYDDD format	1997244
XFF	Half of the month: 1 = first half, 2 = second half	2
XFIRSTDE	First day of a 10-day period in a month	19970921
XFIRSTQ	First day of the quarter in YYYYMMDD format	19970901
XLASTDD	Last day of the month in DD format	30
XLASTDDN	Last day of the next month	31
XLASTDDP	Last day of the previous month	31
XLASTDD1	Last day of the month in YYYYMMDD format	19970930
XLASTDD2	Last day of the month in YYMMDD format	970930
XLASTDD3	Last day of the month in DDMMYYYY format	30091997

Table 137. Chkdate Instruction Variables (continued)

Variable	Description	Value
XLASTDD4	Last day of the month in DDMMYY format	300997
XLSTDDJ	Last day of the month in DDD format (Julian day)	273
XLSTDDJ1	Last day of the month in YYYYDDD format	1997273
XMM	Month in MM format	09
XMMYY	Month and year in MMYY format	0997
XMY	Date in MMYYYY format	091997
XMMNAME	Name of the month	SEPTEMBER
XMMP1	Previous month in MM format	08
XMMN1	Next month in MM format	10
XQUARTER	Quarter of the year	3
XTOTWWM	Number of weeks in the month: 1–6	5
XWW[1]	Week of the year	39
XWWD	Week of the year and day of the week	397
XWWLAST	Last week of the month: Y or N	N
XWWMONTH	Week of the month: 1–6	4
XYM	Date in YYYMM format	199709
XYYDDD	Date as a Julian date in YYDDD format	97271
XYYYYDDD	Date as a Julian date YYYYDDD format	1997271
XYY	Year in YY format	97
XYYYY	Year in YYYY format	1997
XYYMM	Year and month in YYMM format	9709
XYYP	Previous year in YY format	96
XYYYYP	Previous year in YYYY format	1996
XYYMMP	Previous month in YYMM format	9708
XYYYYMMP	Previous month in YYYYMM format	199708
XYYN	Next year YY format	98
XYYYYN	Next year in YYYY format	1998
XYYMMN	Next month in YYMM format	9710

Table 137. Chkdate Instruction Variables (continued)

Variable	Description	Value
XYYYYMMN	Next month in YYYYMM format	199710
X1MOND	First Monday of the month in YYYYMMDD format	19970901
X1MONDJ	First Monday of the month in DDD format (Julian day)	244
X1MONDJ1	First Monday of the month in YYYYDDD format (Julian day)	1997244
X2MOND	Second Monday of the month in YYYYMMDD format	19970908
X2MONDJ	Second Monday of the month in DDD format (Julian day)	251
X2MONDJ1	Second Monday of the month in YYYYDDD format (Julian day)	1997251
X3MOND	Third Monday of the month in YYYYMMDD format	19970915
X3MONDJ	Third Monday of the month in DDD format (Julian day)	258
X3MONDJ1	Third Monday of the month in YYYYDDD format (Julian day)	1997258
X4MOND	Fourth Monday of the month in YYYYMMDD format	19970922
X4MONDJ	Fourth Monday of the month in DDD format (Julian day)	265
X4MONDJ1	Fourth Monday of the month in YYYYDDD format (Julian day)	1997265
X5MOND	Fifth Monday of the month in YYYYMMDD format	19970929
X5MONDJ	Fifth Monday of the month in DDD format (Julian day)	272
X5MONDJ1	Fifth Monday of the month in YYYYDDD format (Julian day)	1997272
XFREEDAY	F = free day, W = work day	F
XYMDN1	Calendar day + 1 in YYYYMMDD format	19970929
XDDN1	Calendar day + 1 in DD format	29
XYMDN1	Calendar day + 1 in DDMMYYYY format	29091997
XYMDN2	Calendar day + 2 in YYYYMMDD format	19970930
XYMDN3	Calendar day + 3 in YYYYMMDD format	19971001
XYMDN4	Calendar day + 4 in YYYYMMDD format	19971002
XYMDN5	Calendar day + 5 in YYYYMMDD format	19971003
XYMDN6	Calendar day + 6 in YYYYMMDD format	19971004
XYMDN7	Calendar day + 7 in YYYYMMDD format	19971005
XYMDN8	Calendar day + 8 in YYYYMMDD format	19971006
XYMDN9	Calendar day + 9 in YYYYMMDD format	19971007

Table 137. Chkdate Instruction Variables (continued)

Variable	Description	Value
XYMDN10	Calendar day + 10 in YYYYMMDD format	19971008
XYMDN11	Calendar day + 11 in YYYYMMDD format	19971009
XYMDN12	Calendar day + 12 in YYYYMMDD format	19971010
XYMDN13	Calendar day + 13 in YYYYMMDD format	19971011
XYMDN14	Calendar day + 14 in YYYYMMDD format	19971012
XYMDN15	Calendar day + 15 in YYYYMMDD format	19971013
XYMDN16	Calendar day + 16 in YYYYMMDD format	19971014
XYMDN17	Calendar day + 17 in YYYYMMDD format	19971015
XYMDN18	Calendar day + 18 in YYYYMMDD format	19971016
XYMDN19	Calendar day + 19 in YYYYMMDD format	19971017
XYMDN20	Calendar day + 20 in YYYYMMDD format	19971018
XYMDN21	Calendar day + 21 in YYYYMMDD format	19971019
XYMDN22	Calendar day + 22 in YYYYMMDD format	19971020
XYMDN23	Calendar day + 23 in YYYYMMDD format	19971021
XYMDN24	Calendar day + 24 in YYYYMMDD format	19971022
XYMDN25	Calendar day + 25 in YYYYMMDD format	19971023
XYMDN26	Calendar day + 26 in YYYYMMDD format	19971024
XYMDN27	Calendar day + 27 in YYYYMMDD format	19971025
XYMDN28	Calendar day + 28 in YYYYMMDD format	19971026
XYMDN29	Calendar day + 29 in YYYYMMDD format	19971027
XYMDN30	Calendar day + 30 in YYYYMMDD format	19971028
XWDDN1	Work day + 1 in DD format	29
XWDMYN1	Work day + 1 in DDMMYYYY format	29091997
XWYMDN1	Work day + 1 in YYYYMMDD format	19970929
XWYMDN2	Work day + 2 in YYYYMMDD format	19970930
XWYMDN3	Work day + 3 in YYYYMMDD format	19971001
XWYMDN4	Work day + 4 in YYYYMMDD format	19971002
XWYMDN5	Work day + 5 in YYYYMMDD format	19971003

Table 137. Chkdate Instruction Variables (continued)

Variable	Description	Value
XWYMDN6	Work day + 6 in YYYYMMDD format	19971006
XWYMDN7	Work day + 7 in YYYYMMDD format	19971007
XWYMDN8	Work day + 8 in YYYYMMDD format	19971008
XWYMDN9	Work day + 9 in YYYYMMDD format	19971009
XWYMDN10	Work day + 10 in YYYYMMDD format	19971010
XWYMDN11	Work day + 11 in YYYYMMDD format	19971013
XWYMDN12	Work day + 12 in YYYYMMDD format	19971014
XWYMDN13	Work day + 13 in YYYYMMDD format	19971015
XWYMDN14	Work day + 14 in YYYYMMDD format	19971016
XWYMDN15	Work day + 15 in YYYYMMDD format	19971017
XWYMDN16	Work day + 16 in YYYYMMDD format	19971020
XWYMDN17	Work day + 17 in YYYYMMDD format	19971021
XWYMDN18	Work day + 18 in YYYYMMDD format	19971022
XWYMDN19	Work day + 19 in YYYYMMDD format	19971023
XWYMDN20	Work day + 20 in YYYYMMDD format	19971042
XWYMDN21	Work day + 21 in YYYYMMDD format	19971027
XWYMDN22	Work day + 22 in YYYYMMDD format	19971028
XWYMDN23	Work day + 23 in YYYYMMDD format	19971029
XWYMDN24	Work day + 24 in YYYYMMDD format	19971030
XWYMDN25	Work day + 25 in YYYYMMDD format	19971031
XWYMDN26	Work day + 26 in YYYYMMDD format	19971103
XWYMDN27	Work day + 27 in YYYYMMDD format	19971104
XWYMDN28	Work day + 28 in YYYYMMDD format	19971105
XWYMDN29	Work day + 29 in YYYYMMDD format	19971106
XWYMDN30	Work day + 30 in YYYYMMDD format	19971107
XYDDP1	Calendar day - 1 in DD format	27
XYMDP1	Calendar day - 1 in YYYYMMDD format	19970927
XYMDP2	Calendar day - 2 in YYYYMMDD format	19970926

Table 137. Chkdate Instruction Variables (continued)

Variable	Description	Value
XYMDP3	Calendar day - 3 in YYYYMMDD format	19970925
XYMDP4	Calendar day - 4 in YYYYMMDD format	19970924
XYMDP5	Calendar day - 5 in YYYYMMDD format	19970923
XYMDP6	Calendar day - 6 in YYYYMMDD format	19970922
XYMDP7	Calendar day - 7 in YYYYMMDD format	19970921
XYMDP8	Calendar day - 8 in YYYYMMDD format	19970920
XYMDP9	Calendar day - 9 in YYYYMMDD format	19970919
XYMDP10	Calendar day - 10 in YYYYMMDD format	19970918
XYMDP11	Calendar day - 11 in YYYYMMDD format	19970917
XYMDP12	Calendar day - 12 in YYYYMMDD format	19970916
XYMDP13	Calendar day - 13 in YYYYMMDD format	19970915
XYMDP14	Calendar day - 14 in YYYYMMDD format	19970914
XYMDP15	Calendar day - 15 in YYYYMMDD format	19970913
XYMDP16	Calendar day - 16 in YYYYMMDD format	19970912
XYMDP11	Calendar day - 17 in YYYYMMDD format	19970916
XYMDP18	Calendar day - 18 in YYYYMMDD format	19970910
XYMDP19	Calendar day - 19 in YYYYMMDD format	19970909
XYMDP20	Calendar day - 20 in YYYYMMDD format	19970908
XYMDP21	Calendar day - 21 in YYYYMMDD format	19970907
XYMDP22	Calendar day - 22 in YYYYMMDD format	19970906
XYMDP23	Calendar day - 23 in YYYYMMDD format	19970905
XYMDP24	Calendar day - 24 in YYYYMMDD format	19970904
XYMDP25	Calendar day - 25 in YYYYMMDD format	19970903
XYMDP26	Calendar day - 26 in YYYYMMDD format	19970902
XYMDP27	Calendar day - 27 in YYYYMMDD format	19970901
XYMDP28	Calendar day - 28 in YYYYMMDD format	19970831
XYMDP29	Calendar day - 29 in YYYYMMDD format	19970830
XYMDP30	Calendar day - 30 in YYYYMMDD format	19970829

Table 137. Chkdate Instruction Variables (continued)

Variable	Description	Value
XWDDP1	Work day - 1 in DD format	26
XWDMYP1	Work day - 1 in DDMMYYYY format	26091997
XWYMDP1	Work day - 1 in YYYYMMDD format	19970926
XWYMDP2	Work day - 2 in YYYYMMDD format	19970925
XWYMDP3	Work day - 3 in YYYYMMDD format	19970924
XWYMDP4	Work day - 4 in YYYYMMDD format	19970923
XWYMDP5	Work day - 5 in YYYYMMDD format	19970922
XWYMDP6	Work day - 6 in YYYYMMDD format	19970919
XWYMDP7	Work day - 7 in YYYYMMDD format	19970918
XWYMDP8	Work day - 8 in YYYYMMDD format	19970917
XWYMDP9	Work day - 9 in YYYYMMDD format	19970916
XWYMDP10	Work day - 10 in YYYYMMDD format	19970915
XWYMDP11	Work day - 11 in YYYYMMDD format	19970912
XWYMDP12	Work day - 12 in YYYYMMDD format	19970911
XWYMDP13	Work day - 13 in YYYYMMDD format	19970910
XWYMDP14	Work day - 14 in YYYYMMDD format	19970909
XWYMDP15	Work day - 15 in YYYYMMDD format	19970908
XWYMDP16	Work day - 16 in YYYYMMDD format	19970905
XWYMDP17	Work day - 17 in YYYYMMDD format	19970904
XWYMDP18	Work day - 18 in YYYYMMDD format	19970903
XWYMDP19	Work day - 19 in YYYYMMDD format	19970902
XWYMDP20	Work day - 20 in YYYYMMDD format	19970901
XWYMDP21	Work day - 21 in YYYYMMDD format	19970829
XWYMDP22	Work day - 22 in YYYYMMDD format	19970828
XWYMDP23	Work day - 23 in YYYYMMDD format	19970827
XWYMDP24	Work day - 24 in YYYYMMDD format	19970826
XWYMDP25	Work day - 25 in YYYYMMDD format	19970825
XWYMDP26	Work day - 26 in YYYYMMDD format	19970822

Table 137. Chkdate Instruction Variables (continued)

Variable	Description	Value
XWYMDP27	Work day - 27 in YYYYMMDD format	19970821
XWYMDP28	Work day - 28 in YYYYMMDD format	19970820
XWYMDP29	Work day - 29 in YYYYMMDD format	19970819
XWDYMD30	Work day - 30 in YYYYMMDD format	19970818

[1] XWW is the variable updated by CHKDATE to show the week of the year, in terms of week number. It is a fixed calculation that starts the first day of the year and has a fixed length of seven days. It is assumed that week number 1 begins on January 1st, regardless of what day of the week it is.

The IBM Z Workload Scheduler-supplied variable are calculated on the occurrence input arrival date and time, whereas the OCL variables are calculated on an input date, such as a date provided with a promptable variable.

The following table describes the keywords that can be used.

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
DATE1()	Required[1]	No	The input date in the following formats: YYYYMMDD, YYYY/MM/DD, YYMMDD, or YY/MM/DD. DATE1() and DATE2() are mutually exclusive.		YYMMDD
DATE2()	Required[1]	No	The input date in the following formats: DDMMYYYY, DD/MM/YYYY, DDMMYY, or DD/MM/YY. DATE2() and DATE1() are mutually exclusive.		DDMMYY
CALNAME()	Optional	No	The scheduler default calendar ID. It defaults to DEFAULT.	CAL	

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
SUBSYS()	Optional	No	The scheduler subsystem name. It overrides the subsystem ID specified in a previous INIT instruction or in the EQQYRPRM member, except when you specify the SUBSYS() keyword in the EQQYPARM DD card.		
MSG()	Optional	No	It can be YES or NO. It specifies whether the program is to display the description and the values of the calculated variables in the OCL SYSOUT. It defaults to YES.		
INCVALUE()	Optional	No	This value indicates the decreasing value for the variables calculation. OCL will add days to, or subtract days from, an input date up to the *incvalue*. (OCL will calculate the new date +/- *n* work days or calendar days, where *n* is a value in the range 1–*incvalue*.) *incvalue* can be a value in the range 1–30 and overlays the value specified in PARMLIB.	INC	
Note: Mutually exclusive with another keyword.					

Examples:

```
CHKDATE  DATE2(&datac) MSG(NO) INC(15) CAL(calend01)
CHKDATE  DATE2(201097) MSG(NO)
```

The CHKDATE instruction can be an extension to the scheduler SETVAR directive. For example, you can use it to perform a double arithmetic calculation on an input date. If you need to calculate a date that corresponds to the occurrence input arrival date plus 2 workdays and plus 20 calendar days, you can use the following OCL instructions:

```
CHKDATE  DATE1(&OYMD1) INC(2) MSG(NO)
CHKDATE  DATE1(XWYMDN2) INC(20)
SETUPD var1 = XYMDN20
```

The first CHKDATE instruction calculates the occurrence input arrival date plus 2 workdays. The result is stored in the variable XWYMDN2. The second CHKDATE instruction uses the variable XWYMDN2 as the input date to calculate the new date: XWYMDN2 + 20 calendar days. The result is stored in variable XYMDN20, which you can use, for example, to update the default value of a variable in a variable table.

The CHKDATE instruction returns one of the following return codes:

RESULT = 0

Valid date.

RESULT = 8

Invalid date. Refer to the error messages.

Note: When using the CHKDATE instruction, the variable name must be put between single quotation marks. For example:

```
      CHKDATE  DATE1('XWYMDN2') INC(20)
```

COMPL

The COMPL instruction completes occurrences or operations within an occurrence in the current plan.

For a description of the keywords that can be used, see Table 138: Keywords used in the Compl Instruction on page 294.

Table 138. Keywords used in the Compl Instruction

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
APPL()	Required	No	The name of the application to be completed.		
OPNO()	Optional	Yes	The number of the operation to be completed. If OPNO() is specified then COMPL will complete	OP	

Table 138. Keywords used in the Compl Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
			the operations and not the occurrence. If the operation is the last one in the occurrence, then the occurrence will be completed.		
IADATE()	Optional[1]	No	The input arrival date of the application occurrence. IADATE=() means the current date. IADATE() and IA() are mutually exclusive.	IAD	YYMMDD
IATIME()	Optional[1]	No	The input arrival time of the application occurrence. If IADATE(=) is specified, IATIME() assumes the default input arrival time that is defined in the DEFIAT parameter of the EQQYRPRM member. IATIME() and IA() are mutually exclusive.	IAT	HHMM
IA()	Optional[1]	N	The input arrival date and time of the application occurrence. IA(=) assumes the current date and time. It defaults to the earliest or latest input arrival date and time of the application		YYMMDDHHMM

Table 138. Keywords used in the Compl Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
			occurrences found in the current plan, depending on the SORT parameter specified in PARMLIB or in the INIT instruction. IA() cannot be used with IADATE() or IATIME().		
Note: [1] Mutually exclusive with another keyword.					

Note: if you do not specify IADATE(), IATIME(), or IA(), OCL considers only the application occurrences in the current plan that do not have Complete status.

Examples:

COMPL APPL(TEST01,TEST02,TEST03)

Completes the occurrences TEST01, TEST02, and TEST03 with the earliest or latest input arrival date and time, depending on the SORT parameter

COMPL APPL(TEST01) IAD(=)

Completes the occurrence TEST01 with the input arrival date corresponding to the current date and with the default input arrival time

COMPL APPL(TEST01,TEST02,TEST03) IA(=)

Completes the occurrences TEST01, TEST02 and TEST03 with the input arrival date and time corresponding to the current date and time

COMPL APPL(TEST02) IAD(970709) OP(10,40)

Completes operations 10 and 40 of the occurrence TEST02 with the input arrival date 970709 and with the default input arrival time

The COMPL instruction returns one of the following return codes:

RESULT = 0

Occurrences or operations completed

RESULT = 4

Occurrences or operations not found

RESULT = 8

Invalid instruction or PIF problem. See the error messages.

DEL

The DEL instruction deletes an occurrence or an operation from the current plan or from the long-term plan. It modifies the current plan or long-term plan, depending on the date and time specified or defaulted to. If the date and time being used is after the end of the current plan, the modification will be made to the long-term plan instead of the current plan.

Table 139: Keywords used in the Del Instruction on page 297 describes the keywords that can be used.

Table 139. Keywords used in the Del Instruction

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
APPL()	Required	No	The name of the application to be deleted.		
OPNO()	Optional	Yes	The number of the operation to be deleted. If OPNO() is specified, DEL will delete the operations and not the occurrence.	OP	
IADATE()	Optional[1]	No	The input arrival date of the application occurrence. IADATE=() means the current date. IADATE() and IA() are mutually exclusive.	IAD	YYMMDD
IATIME()	Optional[1]	No	The input arrival time of the application occurrence. If IADATE(=) is specified, IATIME() assumes the default input arrival time that is defined in the DEFIAT parameter of the EQQYRPRM	IAT	HHMM

Table 139. Keywords used in the Del Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
			member. IATIME() and IA() are mutually exclusive.		
IA()	Optional[1]	No	The input arrival date and time of the application occurrence. IA(=) assumes the current date and time. It defaults to the earliest or latest input arrival date and time of the application occurrences found in the current plan, depending on the SORT parameter specified in PARMLIB or in the INIT instruction. IA() cannot be used with IADATE() or IATIME().		YYMMDDHHMM

Note: [1] Mutually exclusive with another keyword.

Examples:

DEL APPL(TEST01)

Deletes the occurrence with the earliest or latest input arrival date and time of the application TEST01 (according to the SORT parameter or the INIT instruction)

DEL APPL(TEST01) IAD(=) IAT(1700)

Deletes the occurrence of application TEST01 with the input arrival date corresponding to the current date and with the input arrival time 1700

DEL APPL(TEST01) IAD(970708)

Deletes the occurrence TEST01 with the input arrival date 970708 and the input arrival time corresponding to the default IA time, DEF

DEL APPL(TEST01) IA(=)

Deletes the occurrence TEST01 with the input arrival date and time corresponding to the current date and time

DEL APPL(TEST01) IA(9707081800)

Deletes the occurrence TEST01 with the input arrival date 970708 and time 1800

DEL APPL(TEST01) OP(10,30)

Deletes operations 10 and 30 of the occurrence TEST01 with the earliest or latest input arrival date and time, depending on the SORT parameter

Note: If the occurrence is associated with an application group, DEL will remove the group definition from the occurrence before deleting the occurrence.

The DEL instruction returns one of the following return codes:

RESULT = 0

Occurrence deleted.

RESULT = 4

Occurrence found but operations not found in the current plan.

RESULT = 8

Occurrence not found in the current plan. Invalid instruction or PIF problem. Refer to the error messages.

DELCOND

The DELCOND instruction adds a condition to an operation in the current plan. It modifies the current plan, depending on the date and time specified or defaulted to.

Table 140: Keywords used in the Delcond Instruction on page 299 describes the keywords that can be used.

Table 140. Keywords used in the Delcond Instruction

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
APPL()	Required	No	The name of the application to be modified.		
OPNO()	Optional	No	The number of the operation to be modified. It defaults to the default operation number specified in the OCL program DEFOPNO.	OP	

Table 140. Keywords used in the Delcond Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
IADATE()	Optional[1]	No	The input arrival date of the application occurrence. IADATE(=) assumes the current date. IADATE() and IA() are mutually exclusive.	IAD	YYMMDD
IATIME()	Optional[1]	No	The input arrival time of the application occurrence. If IADATE(=) is specified, IATIME() assumes the default input arrival time specified in the DEFIAT parameter of the EQQYRPRM member. IATIME() and IA() are mutually exclusive.	IAT	HHMM
IA()	Optional	No	The input arrival date and time of the application occurrence. IA(=) assumes the current date and time. It defaults to the earliest or latest input arrival date and time of the application occurrence. occurrence found in the current plan, depending on the SORT parameter specified in PARMLIB or in the INIT instruction. IA() cannot be used with IADATE() or IATIME().		YYMMDDHHMM
CONDID()	Optional	No	The number of the condition to be inserted. It defaults to the default		

Table 140. Keywords used in the Delcond Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
			condition id specified in the OCL program DEFCONDID.		

Note: [1] Mutually exclusive with another keyword.

The DELCOND instruction returns one of the following return codes:

RESULT = 0

Predecessor dependency definition added.

RESULT = 4

Predecessor dependency not found in the current plan.

RESULT = 8

Invalid instruction or PIF problem. Refer to the error messages.

DELPRED

The DELPRED instruction deletes the external predecessors of an occurrence in the current plan or in the long-term plan. It modifies the current plan or long-term plan, depending on the date and time specified or defaulted to. If the date and time being used is after the end of the current plan, the modification will be made to the long-term plan instead of the current plan.

Table 141: Keywords used in the Delpred Instruction on page 301 describes the keywords that can be used.

Table 141. Keywords used in the Delpred Instruction

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
APPL()	Required	No	The name of the application to be modified.		
OPNO()	Optional	Yes	The number of the operation to be modified.	OP	
IADATE()	Optional[1]	No	The input arrival date of the application occurrence. IADATE=() means the current date.	IAD	YYMMDD

Table 141. Keywords used in the Delpred Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
			IADATE() and IA() are mutually exclusive.		
IATIME()	Optional[1]	No	The input arrival time of the application occurrence. If IADATE(=) is specified, IATIME() assumes the default input arrival time that is defined in the DEFIAT parameter of the EQQYRPRM member. IATIME() and IA() are mutually exclusive.	IAT	HHMM
IA()	Optional[1]	No	The input arrival date and time of the application occurrence. IA(=) assumes the current date and time. It defaults to the earliest or latest input arrival date and time of the application occurrences found in the current plan, depending on the SORT parameter specified in PARMLIB or in the INIT instruction. IA() cannot be used with IADATE() or IATIME().		YYMMDDHHMM
EXCLOPNO()	Optional	Yes	The operation number that must not be modified.	EOP	

Table 141. Keywords used in the Delpred Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
EXCLPRE()	Optional	Yes	The predecessor application name that must not be deleted.	EPRE	
Note: [1] Mutually exclusive with another keyword.					

Examples:

DELPRED APPL(TEST01)

All the predecessor dependencies will be deleted from occurrence TEST01 with the latest or earliest input arrival date and time, depending on the SORT parameter specified in PARMLIB or in the INIT instruction.

DELPRED APPL(TEST01) IAD(970708) OP(40,50)

All the predecessor dependencies will be deleted from operations 40 and 50 of the occurrence TEST01 with the input arrival date 970708 and the input arrival time that corresponds to the current time.

DELPRED APPL(TEST01) IA(=) EXCLOP(20)

All the predecessor dependencies will be deleted from the operations other than operation 20 of the occurrence TEST01 with the input arrival date and time that corresponds to the current date and time.

DELPRED APPL(TEST01) IA(=) EPRE(TEST00)

All the predecessor dependencies other than occurrence TEST00 will be deleted from the occurrence EST01 with the input arrival date and time that corresponds to the current date and time.

The DELPRED instruction returns one of the following return codes:

RESULT = 0

Predecessor dependency definition deleted.

RESULT = 4

Occurrence or operation not found in the current plan or in the long-term plan.

RESULT = 8

Invalid instruction or PIF problem. Refer to the error messages.

DELRES

The DELRES instruction deletes one or more special resources from an operation in the current plan.

Table 142: Keywords used in the Delres Instruction on page 304e describes the keywords that can be used.

Table 142. Keywords used in the Delres Instruction

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
APPL()	Required	No	The name of the application to be modified.		
OPNO()	Optional	Yes	The number of the operation to be modified. It defaults to the default operation number, DEFOPNO, specified in the OCL program.	OP	
IADATE()	Optional[1]	No	The input arrival date of the application occurrence. IADATE=() means the current date. IADATE() and IA() are mutually exclusive.	IAD	YYMMDD
IATIME()	Optional[1]	No	The input arrival time of the application occurrence. If IADATE(=) is specified, IATIME() assumes the default input arrival time that is defined in the DEFIAT parameter of the EQQYRPRM member. IATIME() and IA() are mutually exclusive.	IAT	HHMM
IA()	Optional[1]	No	The input arrival date and time of the application occurrence. IA(=) assumes the current date and time. It defaults to the earliest		YYMMDDHHMM

Table 142. Keywords used in the Delres Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
			or latest input arrival date and time of the application occurrences found in the current plan, depending on the SORT parameter specified in the EQQYRPRM member or in the INIT instruction. IA() cannot be used with IADATE() or IATIME().		
EXCLRES()	Optional	Yes	The special resource that must not be deleted.	ERES	
RESNAME()	Optional	No	The special resource to be deleted. Do not include parentheses in the resource name.	RN	

Note: [1] Mutually exclusive with another keyword.

Examples:

DELRES APPL(TEST01)

Deletes all the special resources from the operations of the occurrence TEST01 with the latest or earliest input arrival date and time, depending on the SORT parameter specified in the EQQYRPRM member or in the INIT instruction

DELRES APPL(TEST01) IAD(=) OP(10)

Deletes all the special resources from operation 10 of the occurrence TEST01 with the input arrival date corresponding to the current date and with the input arrival time corresponding to the default input arrival time, DEFIAT

DELRES APPL(TEST01) IA(=) OP(10) ERES(PROVA.SR1)

Deletes all the special resources, except for special resource PROVA.SR1, from operation 10 of the occurrence TEST01, with the input arrival date and time corresponding to the current date and time.

The DELRES instruction returns one of the following return codes:

RESULT = 0

Special resource deleted.

RESULT = 4

Operation not found in the current plan.

RESULT = 8

Invalid instruction or PIF problem. Refer to the error messages.

DELSIMP

The DELSIMP instruction deletes condition dependencies of an operation in the current plan. It modifies the current plan, depending on the date and time specified or defaulted to.

Table 143: Keywords used in the Delsimp Instruction on page 306 describes the keywords that can be used.

Table 143. Keywords used in the Delsimp Instruction

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
APPL()	Required	No	The name of the application to be modified.		
OPNO()	Optional	No	The number of the operation to be modified. It defaults to the default operation number specified in the OCL program DEFOPNO.	OP	
IADATE()	Optional[1]	No	The input arrival date of the application occurrence. IADATE(=) assumes the current date. IADATE() and IA() are mutually exclusive.	IAD	YYMMDD
IATIME()	Optional[1]	No	The input arrival time of the application occurrence. If IADATE(=) is specified, IATIME() assumes the default input arrival time	IAT	HHMM

Table 143. Keywords used in the Delsimp Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
			specified in the DEFIAT parameter of the EQQYRPRM member. IATIME() and IA() are mutually exclusive.		
IA()	Optional	No	The input arrival date and time of the application occurrence. IA(=) assumes the current date and time. It defaults to the earliest or latest input arrival date and time of the application occurrence. occurrence found in the current plan, depending on the SORT parameter specified in PARMLIB or in the INIT instruction. IA() cannot be used with IADATE() or IATIME().		YYMMDDHHMM
PREOPNO()	Optional	No	The predecessor operation number. It defaults to the default predecessor operation number specified in the DEFPREOPNO parameter of the EQQYRPRM member.	PO	
PREAPPL()	Optional	No	The predecessor application name. It defaults to the application name specified in the application occurrence.	PA	

Table 143. Keywords used in the Delsimp Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
PREIADATE()	Optional	No	The predecessor application input arrival date. It defaults to the date specified in IADATE().	PIAD	YYMMDD
PREIATIME()	Optional	No	The predecessor application time. It defaults to the time specified in IATIME().	PIAT	HHMM
CONDID()	Optional	No	The number of the condition to be inserted. It defaults to the default condition id specified in the OCL program DEFCONDID.		
CHKTYPE()	Optional	No	Check type. Possible values are: **RC** Return code **ST** Status The default is ST.		
LOG()	Optional	No	Logical operator: GE = Greater than or equal to. Valid only for RC check type. GT = Greater than. Valid only for RC check type. LE = Less than or equal to. Valid		

Table 143. Keywords used in the Delsimp Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
			only for RC check type. LT = Less than. Valid only for RC check type. EQ = Equal to. NE = Not equal to. Use it to specify conditions on final statuses only. RG = Range. The default is EQ.		
VALRC()	Optional	No	Return code, valid only for RC check type. The default is 0000.		
VALRC2()	Optional	No	Return code, valid only for RC check type, as second boundary in a range expressed by the RG logical operator. The default is 9999.		
VALST()	Optional	No	Condition status, Valid only for ST check type. The default is C.		

Note: [1] Mutually exclusive with another keyword.

The DELSIMP instruction returns one of the following return codes:

RESULT = 0

Predecessor dependency definition added.

RESULT = 4

Predecessor dependency not found in the current plan.

RESULT = 8

Invalid instruction or PIF problem. Refer to the error messages.

EXIT

Syntax

EXIT *result*| *return_code*

The EXIT instruction terminates the program with a specified return code, which you can specify in the instruction. The value can be a number or RESULT. You can use this instruction to set a return code, which you can then test, using the JCL COND or IF parameters, to determine which job steps to execute.

Examples:

```
//SYSIN  DD  *
* OCL instructions
IF VAR2 = 'KO' THEN EXIT 99
EXIT RESULT
⋮
//*
//TESTIF   IF (RC = 99) THEN
//STEP002 EXEC PGM=MYPROG
//…
//TESTIF  ENDIF
```

FORCE

The FORCE instruction forces the execution of an occurrence or of an operation within an occurrence in the current plan.

Table 144: Keywords used in the Force Instruction on page 310 describes the keywords that can be used.

Table 144. Keywords used in the Force Instruction

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
APPL()	Required	No	The name of the application to be started.		
OPNO()	Optional	Yes	The number of the operation to be started. It defaults to the default operation number, DEFOPNO, specified in the OCL program.[2]	OP	

Table 144. Keywords used in the Force Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
IADATE()	Optional[1]	No	The input arrival date of the application occurrence. IADATE=() means the current date. IADATE() and IA() are mutually exclusive.	IAD	YYMMDD
IATIME()	Optional[1]	No	The input arrival time of the application occurrence. If IADATE (=) is specified, IATIME() assumes the default input arrival time that is defined in the DEFIAT parameter of the EQQYRPRM member. IATIME() and IA() are mutually exclusive.	IAT	HHMM
IA()	Optional[1]	N	The input arrival date and time of the application occurrence. IA(=) assumes the current date and time. It defaults to the earliest or latest input arrival date and time of the application occurrences found in the current plan, depending on the SORT parameter specified in PARMLIB or in the INIT instruction. IA()		YYMMDDHHMM

Table 144. Keywords used in the Force Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
			cannot be used with IADATE() or IATIME().		

Notes:

1. Mutually exclusive with another keyword.
2. In order to start the occurrence from a specific operation number, the program performs the following actions:
   a. Deletes all the special resources associated to the operation
   b. Deletes all the external predecessors
   c. Changes the operation status to R (ready), and changes the following automatic options of the operation:
      - Job submission to YES
      - Time dependent to NO
      - Operation descriptive text to '* OCL forced this oper *'
      - Parallel server to 1
      - Resource R1 to 0
      - Resource R2 to 0

      If the change is successful, the return code is 0. (The return code is stored in the RESULT variable.) Otherwise it does the following:
      i. Deletes the external predecessors of all the internal predecessors of the specified operation
      ii. Changes again the operation status to R (ready) and the operation automatic options
   d. Checks whether the occurrence is started.

Examples:

FORCE APPL(TEST01)

Forces the default operation, DEFOPNO, to start

FORCE APPL(TEST01) OP(50)

Forces operation 50 to start

FORCE APPL(TEST01) IA(=)

Forces operation 50 to start, using the current date and time as the input arrival date and time

FORCE APPL(TEST01) IA(9707081800) OP(70)

Forces operation 70 to start, using the input arrival date 970708 and the input arrival time 18.00

FORCE APPL(TEST01) IAD(=)

Forces the default operation, DEFOPNO, to start, using the current date as the input arrival date and the default input arrival time

FORCE APPL(TEST01) IAD(&OYMD1)

Forces the default operation, DEFOPNO, to start, using the input arrival date and time value of variable &OYMD1

FORCE APPL(TEST01) IAD(970708) IAT(&OHHMM) OP(60)

Forces operation 60 to start, using the input arrival date 970708 and the input arrival time value of variable &OHHMM

The FORCE instruction returns one of the following return codes:

RESULT = 0

Occurrence or operations status is started.

RESULT = 4

Occurrence found but operations not found in the current plan.

RESULT = 8

Occurrence not found in the current plan. Invalid instruction or PIF problem. Refer to the error messages.

GOTO

The GOTO instruction transfers control to a point specified by a LABEL instruction within the OCL program.

Syntax

GOTO *label_name*

Example:

```
IF &var1 = 'OK' then GOTO esci
⋮
LABEL esci
⋮
```

HOLD

The HOLD instruction holds an operation or all the operations of an occurrence in the current plan.

Table 145: Keywords used in the Hold Instruction on page 314 describes the keywords that can be used.

Table 145. Keywords used in the Hold Instruction

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
APPL()	Required	Yes	The name of the application to be modified.		
OPNO()	Optional	Yes	The number of the operation to be modified. If OPNO() is not specified, the HOLD command, MH, will be issued to all the occurrence operations.	OP	
IADATE()	Optional (1)	No	The input arrival date of the application occurrence. IADATE(=) assumes the current date. IADATE() and IA() are mutually exclusive.	IAD	YYMMDD
IATIME	Optional (1)	No	The input arrival time of the application occurrence. If IADATE(=) is specified, IATIME() assumes that the default input arrival time is specified in the DEFIAT parameter of the EQQYRPRM member. IADATE() and IA() are mutually exclusive.	IAT	HHMM
IA()	Optional (1)	No	The input arrival date and time of the application occurrence. IA(=) assumes the		YYMMDDHHMM

Table 145. Keywords used in the Hold Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
			current date and time. It defaults to the earliest or latest input arrival date and time of the application occurrence occurrences found in the current plan, depending on the SORT parameter. IA() cannot be used together with IADATE() or IATIME().		
Note: 1 Mutually exclusive with another keyword.					

Example control statements:

HOLD APPL(TEST01)

Sets to HOLD status all operations of occurrence TEST01

HOLD APPL(TEST01) OP(10,30,50)

Sets to hold status operations 10, 30, and 50 of occurrence TEST01

HOLD APPL(TEST01) IAD(=)

Sets to HOLD status all operations of occurrence TEST01 for the current input arrival date and the default input arrival time

HOLD APPL(TEST01) IAD(=) IAT(1800)

Sets to HOLD status all operations of occurrence TEST01 for the current input arrival date and input arrival time 18.00

HOLD APPL(TEST01) IAD(970708)

Sets to HOLD status all operations of occurrence TEST01 for input arrival date 970708 and the default input arrival time

HOLD APPL(TEST01) IA(=)

Sets to HOLD status all operations of occurrence TEST01 for the current input arrival date and time

HOLD APPL(TEST01) IA(9707081801)

Sets to HOLD status all operations of occurrence TEST01 for input arrival date 970708 and input arrival time 18.01

The HOLD instruction returns one of the following return codes:

RESULT = 0

Operations status modified.

RESULT = 4

Occurrence found but operations not found in the current plan.

RESULT = 8

Occurrence not found in the current plan. Invalid instruction or PIF problem. Refer to the error messages.

IF-THEN-ELSE

Syntax

IF *expression* THEN *instruction1*; ELSE *instruction2*

IF-THEN-ELSE conditionally processes an instruction depending on the evaluation of the expression. The expression is evaluated and results in 0 (false) or 1 (true). The instruction after the THEN clause is processed only if the result is 1 (true). If you specify an ELSE clause, the instruction after the ELSE clause is processed only if the result of the evaluation is 0 (false).

Example:

```
IF result > 0 then exit 99; else ADD(TEST01)
```

Note:

1. If the ELSE clause is on the same line as the last instruction of the THEN clause, it must be preceded by a semicolon (;) followed immediately by a blank character.
2. If the ELSE clause is on the line following the last instruction of the THEN clause, that last THEN clause line must end with a semicolon followed immediately by a comma.

Example:

```
if substr(&OYMD1,5,2) = 30 then GOTO DAY30;,
else EXIT 50
```

The ELSE clause binds to the nearest IF instruction at the same level.

The expression can contain concatenation, comparison, and logical operators. An operator represents an operation, such as addition, to be performed on one or two terms.

Comparison operations

About this task

The comparison operators compare two terms and return the value 1 if the result of the comparison is true, or 0 if the result is false.

The comparison operators and operations are:

=

True if the terms are equal (numerically or when padded, and so forth)

\=, ¬=, /=

True if the terms are not equal (inverse of =)

>

Greater than

<

Less than

><

Greater than or less than (the same as not equal)

<>

Greater than or less than (the same as not equal)

>=

Greater than or equal to

\<, ¬<

Not less than

<=

Less than or equal to

\>, ¬>

Not greater than

==

True if the terms are strictly equal (identical)

\==, ¬==, /==

True if the terms are not strictly equal (this is the inverse of ==)

>>

Strictly greater than

<<

Strictly less than

\<<

Strictly not less than

\>>, ¬>>

Strictly not greater than

A character string has the value false if it is 0, and true if it is 1. The logical operators take one or two such values (values other than 0 or 1 are not allowed) and return 0 or 1 as appropriate:

&	AND	Returns 1 if both terms are true
|	OR	Returns 1 if either term is true

The instruction can be one of the following:

ADD	ADDOP	ADDPRED	ADDRES	CALL
CHGEXTNAME	CHGJOB	CHGOPSAI	CHKAPPL	CHKDATE
COMPL	DEL	DELPRED	DELRES	EXIT
FORCE	GOTO	HOLD	IF-THEN-ELSE	INIT
JSUACT	LABEL	MODOP	NOP	OPSTAT
RELEASE	RELOP	RELSUCC	SET	SETUPD
SRSTAT	UNNOP	UPD	WSSTAT	WTO

Here are some examples:

```
IF RESULT = 8 THEN EXIT 70; ELSE COMPL APPL(TEST01)
IF LEFT(&VAR1,1) = 1 THEN ADD APPL(TEST01)
IF substr(VAR1,5,2) = 'XX' then COMPL APPL(TEST02)
IF VAR4 = VAR1||VAR2||VAR3 THEN UPD VAR4
IF VAR5 > VAR4 THEN SET VAR6 = VAR3 + 2
IF substr(&OYMD1,3,4) = '0612' THEN goto GIU12
IF &OHHMM > '1830' THEN GOTO T1830
```

The IF instruction returns one of the following return codes:

CC = 0

Instruction correctly processed

CC = 8

Syntax error

INIT

Syntax

INIT VARTAB() | SUBSYS() | SORT()

The INIT instruction overrides the IBM Z Workload Scheduler subsystem name and the SORT parameter specified in the EQQYRPRM member, and specifies the variable table name used by the UPD or SETUPD instructions.

Table 146: Keywords used in the Init Instruction on page 319 describes the keywords that you can use:

Table 146. Keywords used in the Init Instruction

Keyword	Requirement	Multiple Values Allowed	Description
VARTAB()	Optional	No	The scheduler variable table to be updated. This keyword is required only by the UPD and SETUPD instructions.
SUBSYS()	Optional	No	The scheduler subsystem name. It overrides the subsystem name specified in the EQQYRPRM member, unless you specify the SUBSYS() keyword in the EQQYPARM DD.
SORT()	Optional	No	This keyword overrides the SORT parameter specified in the EQQYRPRM member. It determines the application occurrence in the current plan to be selected if you do not specify the input arrival date and time in the OCL instruction. SORT(MAX) determines the occurrence with the latest input arrival date and time. SORT(MIN) determines the occurrence with the earliest input arrival date and time.

Examples:

```
INIT VARTAB(&OADID) SUBSYS(OPCC)
INIT SUBSYS(OPCC)
INIT SORT(MAX)
```

INIT returns one of the following return codes:

CC = 0

Instruction correctly processed

CC = 8

Invalid instruction. Refer to the error message.

JSUACT

The JSUACT instruction invokes the TSO JSUACT command, which you can use to activate or inactivate the job submission function in the z/OS® environment, in the distributed environment, or in both environments.

For a description of the keywords that you can use, see Table 147: Keywords used in the JSUACT Instruction on page 320.

Table 147. Keywords used in the JSUACT Instruction

Keyword	Requirement	Multiple Values Allowed	Description
ACT()	Required	No	If you want to activate the job submission function specify YES, otherwise NO.
SUBSYS()	Optional	No	The name of the tracker the JSUACT is directed to. This parameter can be four characters in length. The first character must be alphabetic; the remaining characters must be alphanumeric. All lowercase characters are converted to uppercase. If you specify MSTR, the JSUACT command is directed to all scheduler subsystems on the z/OS® system where the JSUACT command was issued. Default is OPCA.
TRACE()	Optional	No	Event tracing indicator. When a nonzero positive number is specified, a trace entry is created for each event generated by the JSUACT command. 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.
TYPE()	Optional		Indicates whether the job submission must be deactivated. Possible values are: **H** To deactivate job submission in the z/OS® environment. It is the default value. **F** To deactivate job submission in the distributed environment. **B** To deactivate job submission in both environments.

Example:

```
JSUACT ACT(YES) SUBSYS(OPCB) TYPE(H)
```

In this example the JSUACT instruction is used to activate the job submission function in the z/OS® environment.

The JSUACT instruction returns one of the following return codes:

RESULT = 0

Job submission function changed

RESULT = 8

Invalid instruction

KILLJOB

The KILLJOB instruction stops a job that is already running. Applies only to operations running on IBM Z Workload Scheduler Agents or on distributed workstations that are directly connected to the end-to-end server (OPCMASTER).

This action can be taken only on STARTED jobs that are in the EXECUTING status, so that their operation number is known. The application number is required.

For a description of the keywords that you can use, see Table 148: Keywords used in the Killjob Instruction on page 321.

Table 148. Keywords used in the Killjob Instruction

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
APPL()	Required	Yes	The name of the application to be modified.		
OPNO()	Optional	Yes	The number of the operation to be modified. If OPNO() is not specified, the command will be issued to all the occurrence operations.	OP	
IADATE()	Optional[1]	No	The input arrival date of the application occurrence. IADATE(=) assumes the current date.	IAD	YYMMDD

Table 148. Keywords used in the Killjob Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
			IADATE() and IA() are mutually exclusive.		
IATIME	Optional[1]	No	The input arrival time of the application occurrence. If IADATE(=) is specified, IATIME() assumes that the default input arrival time is specified in the DEFIAT parameter of the EQQYRPRM member. IADATE() and IA() are mutually exclusive.	IAT	HHMM
IA()	Optional[1]	No	The input arrival date and time of the application occurrence. IA(=) assumes the current date and time. It defaults to the earliest or latest input arrival date and time of the application occurrence occurrences found in the current plan, depending on the SORT parameter. IA() cannot be used together with IADATE() or IATIME().		YYMMDDHHMM

Table 148. Keywords used in the Killjob Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
Note: [1] Mutually exclusive with another keyword.					

Example control statements:

KILLJOB APPL(TEST01)

Kills all operations of occurrence TEST01

KILLJOB APPL(TEST01) OP(10,30,50)

Kills operations 10, 30, and 50 of occurrence TEST01

KILLJOB APPL(TEST01) IAD(=)

Kills all operations of occurrence TEST01 for the current input arrival date and the default input arrival time

KILLJOB APPL(TEST01) IAD(=) IAT(1800)

Kills all operations of occurrence TEST01 for the current input arrival date and input arrival time 18.00

KILLJOB APPL(TEST01) IAD(090709)

Kills all operations of occurrence TEST01 for input arrival date 090709 and the default input arrival time

KILLJOB APPL(TEST01) IA(=)

Kills all operations of occurrence TEST01 for the current input arrival date and time

KILLJOB APPL(TEST01) IA(0907081801)

Kills all operations of occurrence TEST01 for input arrival date 970708 and input arrival time 18.01

The KILLJOB instruction returns one of the following return codes:

RESULT = 0

Operations killed.

RESULT = 4

Occurrence found but operations not found in the current plan.

RESULT = 8

Occurrence not found in the current plan. Invalid instruction or PIF problem. Refer to the error messages.

KILLREC

The KILLREC instruction stops a recovery job that is already running. Applies only to operations running on distributed workstations that are directly connected to the end-to-end server (OPCMASTER).

This action can be taken only on recovery jobs that are in the EXECUTING status, so that their operation number is known. The application number is required.

For a description of the keywords that you can use, see Table 149: Keywords used in the Killrec Instruction on page 324.

Table 149. Keywords used in the Killrec Instruction

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
APPL()	Required	Yes	The name of the application to be modified.		
OPNO()	Optional	Yes	The number of the operation to be modified. If OPNO() is not specified, the command will be issued to all the occurrence operations.	OP	
IADATE()	Optional[1]	No	The input arrival date of the application occurrence. IADATE(=) assumes the current date. IADATE() and IA() are mutually exclusive.	IAD	YYMMDD
IATIME	Optional[1]	No	The input arrival time of the application occurrence. If IADATE(=) is specified, IATIME() assumes that the default input arrival time is specified in the DEFIAT parameter of the EQQYRPRM member. IADATE() and IA() are mutually exclusive.	IAT	HHMM
IA()	Optional[1]	No	The input arrival date and time of the application		YYMMDDHHMM

Table 149. Keywords used in the Killrec Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
			occurrence. IA(=) assumes the current date and time. It defaults to the earliest or latest input arrival date and time of the application occurrence occurrences found in the current plan, depending on the SORT parameter. IA() cannot be used together with IADATE() or IATIME().		
Note: [1] Mutually exclusive with another keyword.					

Example control statements:

KILLREC APPL(TEST01)

Kills all recovery jobs of occurrence TEST01

KILLREC APPL(TEST01) OP(312)

Kills recovery job 312 of occurrence TEST01

KILLREC APPL(TEST01) IAD(=) IAT(1800)

Kills all recovery jobs of occurrence TEST01 for the current input arrival date and input arrival time 18.00

KILLREC APPL(TEST01) IAD(090708)

Kills all recovery jobs of occurrence TEST01 for input arrival date 090708 and the default input arrival time

KILLREC APPL(TEST01) IA(=)

Kills all recovery jobs of occurrence TEST01 for the current input arrival date and time

The KILLREC instruction returns one of the following return codes:

RESULT = 0

Operations killed.

RESULT = 4

Occurrence found but operations not found in the current plan.

RESULT = 8

Occurrence not found in the current plan. Invalid instruction or PIF problem. Refer to the error messages.

LABEL

The LABEL instruction defines a label within the OCL program that is the target of a GOTO instruction.

Syntax

LABEL *label_name*

Example:

```
IF &var1 = 'OK' then GOTO esci
⋮
LABEL esci
⋮
LABEL esci2
```

Note: If OCL processes a LABEL instruction without branching to it in response to a GOTO instruction, the return code is the highest set by any of the OCL instructions.

The LABEL instruction returns one of the following return codes:

LASTRC

Highest return code returned by the program routines if OCL processed the label without branching to it

CC = 8

Label name not specified

MODCOND

The MODCOND instruction adds a condition to an operation in the current plan. It modifies the current plan, depending on the date and time specified or defaulted to.

Table 150: Keywords used in the Modcond Instruction on page 326 describes the keywords that can be used.

Table 150. Keywords used in the Modcond Instruction

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
APPL()	Required	No	The name of the application to be modified.		

Table 150. Keywords used in the Modcond Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
OPNO()	Optional	No	The number of the operation to be modified. It defaults to the default operation number specified in the OCL program DEFOPNO.	OP	
IADATE()	Optional[1]	No	The input arrival date of the application occurrence. IADATE(=) assumes the current date. IADATE() and IA() are mutually exclusive.	IAD	YYMMDD
IATIME()	Optional[1]	No	The input arrival time of the application occurrence. If IADATE(=) is specified, IATIME() assumes the default input arrival time specified in the DEFIAT parameter of the EQQYRPRM member. IATIME() and IA() are mutually exclusive.	IAT	HHMM
IA()	Optional	No	The input arrival date and time of the application occurrence. IA(=) assumes the current date and time. It defaults to the earliest or latest input arrival date and time of the application occurrence. occurrence found in the current plan, depending on the SORT parameter specified in PARMLIB or in the INIT		YYMMDDHHMM

Table 150. Keywords used in the Modcond Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
			instruction. IA() cannot be used with IADATE() or IATIME().		
CONDID()	Optional	No	The number of the condition to be inserted. It defaults to the default condition id specified in the OCL program DEFCONDID.		
COUNT()	Optional	No	Condition counter. Use it to define the rule type: 0 = All the condition dependencies in this condition must be true *n>0* = At least *n* out of the condition dependencies in this condition must be true The default is the current value.		
DESC()	Optional	No	Descriptive text.		

Note: [1] Mutually exclusive with another keyword.

The MODCOND instruction returns one of the following return codes:

RESULT = 0

Predecessor dependency definition added.

RESULT = 4

Predecessor dependency not found in the current plan.

RESULT = 8

Invalid instruction or PIF problem. Refer to the error messages.

MODOP

The MODOP instruction changes the operation's details.

Table 151: Keywords used in the Modop Instructions on page 329 describes the keywords that can be used to select the operations to be modified.

Table 151. Keywords used in the Modop Instructions

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
APPL()	Required	Yes	The name of the application to be modified.		
CLNTYPE()	Optional	No	Cleanup type: **A** Automatic **I** Immediate **M** Manual **N** None If not specified, the default is N.		
CONDRJOB()	Optional	No	Specifies if the operation might recover a conditional predecessor (Y or N). The default is N.		
EXPJCL	Optional	No	Expanded JCL used (Y or N). If not specified the default is N.		
OPNO()	Optional	Yes	The number of the operation to be modified. If you omit this parameter, the process changes	OP	

Table 151. Keywords used in the Modop Instructions (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
			all the operations in the selected occurrence, regardless of any default operation number, DEFOPNO, specified in the OCL program.		
IADATE()	Optional[1]	No	The input arrival date of the application occurrence. IADATE(=) assumes the current date. IADATE() and IA() are mutually exclusive.	IAD	YYMMDD
IATIME	Optional[1]	No	The input arrival time of the application occurrence. If IADATE(=) is specified, IATIME() assumes the default input arrival time specified in the DEFIAT parameter of the EQQYRPRM member. IADATE() and IA() are mutually exclusive.	IAT	HHMM
IA()	Optional[1]	No	The input arrival date and time of the application occurrence. IA(=) assumes the current date and time. It defaults to the earliest or latest input arrival date and time of the application occurrences found in the current plan, depending on the SORT parameter. IA() cannot be used together with IADATE() or IATIME().		YYMMDDHHMM

Table 151. Keywords used in the Modop Instructions (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
JOBCRT()	Optional	No	Specifies whether the operation is critical or eligible for WLM assistance, if late. Possible values are: **P** Critical path. **W** Eligible for WLM assistance. **N** Not eligible for WLM assistance. This is the default.		
JOBPOL()	Optional	No	Specifies the WLM assistance policy to apply, if the job was defined as critical. Possible values are: **D** Deadline. The job is assisted if it has not completed at deadline time. **L** Long Duration. The job is		

Table 151. Keywords used in the Modop Instructions (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
			assisted if it runs beyond the estimated duration. **S** Latest Start Time. The job is assisted if it was submitted before its latest start time. **C** Conditional. An algorithm is used to decide whether to apply the Deadline or Latest Start Time policy. **blank** The policy set in the OPCOPTS statement is applied.		
MONITOR	Optional	No	Specifies if the operation is monitored by an external product (Y or N).		

Table 151. Keywords used in the Modop Instructions (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
USRSYS()	Optional	No	User sysout needed (Y or N). If not specified the default is N.		
WLMSCLS	Optional	No	The WLM service class.		

Note: [1] Mutually exclusive with another keyword.

Table 152: Operations Details that can be modified on page 333 describes the operation details that can be modified.

Table 152. Operations Details that can be modified

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
JOBNAME()	Optional	No	The job name	JOB	
WSNAME()	Optional	No	The workstation name	WS	
DESC()	Optional	No	The operation descriptive text		
EDUR()	Optional[1]	No	The operation estimated duration. EDUR() and DURATION() are mutually exclusive.		HHMM
ASUB()	Optional	No	The automatic job submission option (Y or N)		
AJR()	Optional	No	The hold/release option (Y or N)		
TIMEDEP()	Optional	No	The time dependent option (Y or N)		
CLATE()	Optional	No	The cancel-if-late option (Y or N)		
OPIA()	Optional	No	The operation input arrival date and time		YYMMDDHHMM

Table 152. Operations Details that can be modified (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
OPDL()	Optional	No	The operation deadline		YYMMDDHHMM
PSUSE()	Optional	No	The number of parallel servers required by the operation		
R1USE()	Optional	No	A value in the range 0–99, indicating the maximum capacity of workstation resource 1		
R2USE()	Optional	No	A value in the range 0–99, indicating the maximum capacity of workstation resource 2		
DURATION()	Optional[1]	No	The operation estimated duration. EDUR() and DURATION() are mutually exclusive.		HHMMSS

Note: [1] Mutually exclusive with another keyword.

Examples:

MODOP APPL(TEST01) OP(10) TIMEDEP(N)

Changes the time dependent option for operation 10

MODOP APPL(TEST01) IAD(&OYMD1), JOB(job22222) OP(20) OPIA(&OYMD1.1600)

Changes the operation input arrival time

The MODOP instruction returns one of the following return codes:

RESULT = 0

Occurrence modified.

RESULT = 4

Occurrence found but operations not found in the current plan, or the operation input arrival or deadline are not valid, but are accepted. Refer to the warning message.

RESULT = 8

Occurrence not found in the current plan. Invalid instruction or PIF problem. Refer to the error messages.

Note: Using the MODOP instruction as follows:

```
MODOP APPL(APPLNAME) IADATE(IADATE) IATIME(IATIME) OPNO(OPNO)
```

one of the following parameters must be specified:

- AJR
- ASUB
- CLATE
- CLNTYPE
- CONDRJOB
- DESC
- DURATION
- EDUR
- EXPJCL
- JOBCRT
- JOBNAME
- JOBPOL
- MONITOR
- OPDL
- OPIA
- PSUSE
- R1USE
- R2USE
- TIMEDEP
- USRSYS
- WLMSCLS
- WSNAME

Otherwise the message, EQQCL4VW NO PARAMETER WAS ISSUED, message is issued.

NOP

The NOP instruction removes an operation that is already in the current plan. When a NOP operation is ready to be started, IBM Z Workload Scheduler immediately sets it to C status. The operation is not submitted and successor operations are eligible to start. The NOP instruction can be issued for any operation that has status A, R, *, W, or, for computer workstations

with automatic reporting only, C. NOP operations are identified by the N extended status code. If you want to restore the operation, use the UNNOP instruction.

Table 153: Keywords used in the Nop Instruction on page 336 describes the keywords that can be used.

Table 153. Keywords used in the Nop Instruction

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
APPL()	Required	Yes	The name of the application to be modified.		
OPNO()	Optional	Yes	The number of the operation to be modified. If OPNO() is not specified, the NOP command, NP, will be issued to all the occurrence operations.	OP	
IADATE()	Optional[1]	No	The input arrival date of the application occurrence. IADATE(=) assumes the current date. IADATE() and IA() are mutually exclusive.	IAD	YYMMDD
IATIME	Optional[1]	No	The input arrival time of the application occurrence. If IADATE(=) is specified, IATIME() assumes that the default input arrival time is specified in the DEFIAT parameter of the EQQYRPRM member. IADATE() and IA() are mutually exclusive.	IAT	HHMM
IA()	Optional[1]	No	The input arrival date and time of the application occurrence. IA(=) assumes the current date and time. It defaults to the earliest or latest input arrival date and time of the application occurrences found in the current plan, depending on		YYMMDDHHMM

Table 153. Keywords used in the Nop Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
			the SORT parameter. IA() cannot be used together with IADATE() or IATIME().		
Note: [1] Mutually exclusive with another keyword.					

Example control statements:

NOP APPL(TEST01)

Sets all the operations of occurrence TEST01 in NOP status

NOP APPL(TEST01) OP(10,30,50)

Sets operations 10, 30, and 50 of occurrence TEST01 in NOP status

NOP instruction returns one of the following return codes:

RESULT = 0

Operations status modified.

RESULT = 4

Occurrence found but operations not found in the current plan.

RESULT = 8

Occurrence not found in the current plan. Invalid instruction or PIF problem. Refer to the error messages.

OPSTAT

The OPSTAT instruction invokes the TSO command OPSTAT, which you can use to change the status of an operation at any workstation, except workstations that have the nonreporting attribute. Events generated by OPSTAT are matched against operations on the ready list. Events received for operations in waiting, W, or complete, C, status are ignored. Jobs and tasks that are running are always allowed to finish. If there is more than one operation at the workstation, you can optionally specify the ADID, IA, OPNUM, or JOBNAME parameters to identify the particular operation whose status is to be changed.

Table 154: Keywords used in the Opstat Instruction on page 337 shows the keywords that can be specified.

Table 154. Keywords used in the Opstat Instruction

Keyword	Requirement	Description	Abbreviation	Format
ADID()	Required	The application identifier of the operation whose status you want to change.	A	

Table 154. Keywords used in the Opstat Instruction (continued)

Keyword	Requirement	Description	Abbreviation	Format
CLASS()	Optional	For a printer workstation, specifies the printer SYSOUT class of the operation whose status you want to change.		
DURATION()	Optional	If you are specifying STATUS(C) to set the operation status to complete, you can optionally specify a duration for the completed operation. You specify the duration in hours and minutes, in the format hhmm.		HHMM
FORM()		For a printer workstation, specifies the printer FORM name of the operation whose status you want to change.		
OPNUM()	Optional	The operation number of the operation whose status you want to change.		
ERRORCODE()	Optional	If you are specifying STATUS(E) to set the operation status to *ended-in-error*, it is required that you specify an error code for the operation. The error code can be any 4 characters.		
EVDATE()	Optional	The date of this operation status event. You can use the EVDATE parameter to indicate to the scheduler that the operation changed status at a time other than the current date. If you do not specify this parameter, the operation is considered to have changed status on the date		YYMMDD

Table 154. Keywords used in the Opstat Instruction (continued)

Keyword	Requirement	Description	Abbreviation	Format
		the scheduler processed the OPSTAT command.		
EVTIME()	Optional	The time of this operation status event. You can use the EVTIME parameter to indicate to the scheduler that the operation changed status at a time other than the current time. If you do not specify this parameter, the operation is considered to have changed status at the time the scheduler processed the OPSTAT command.		HHMM
IA()	Optional	The input arrival date and time of the occurrence that contains the operation whose status you want to change. It defaults to the current date and current time or to the current date and default input arrival time, if the default is specified in the DEFIAT parameter of the EQQYRPRM member.		YYMMDDHHMM
JOBNAME()	Optional	The job name associated with the operation whose status you want to change.	J	
NUMJOB()	Optional	Use this optional parameter to specify a job number for an operation. Specify a number in the range 0–99999. The scheduler builds a job number in the format USR*nnnnn*, padding the number with zeros on the left if you specify fewer than 5 digits.		

Table 154. Keywords used in the Opstat Instruction (continued)

Keyword	Requirement	Description	Abbreviation	Format
STATUS	Optional	The operation status that you want to set. You can change the operation status to **C** (operation completed successfully). **Note:** You cannot use the OPSTAT command to change an operation from status W to status C, because predecessor jobs might not have completed.	ST	
SUBSYS()	Optional	The name of the tracker subsystem to which the OPSTAT instruction is directed. It defaults to the subsystem name specified in the OCL OPTRK initialization parameter.		
TOKEN()	Optional	The token assigned for the operation whose status you want to change. A token is automatically assigned for operations started on workstations that specify a user-defined destination ID. The token can be used to uniquely identify the operation.		
TRACE()	Optional	Event tracing indicator. When a positive number is specified, a trace entry is created for each event generated by the OPSTAT command. The trace record is written to the message log file identified by ddname EQQMLOG. The record identifies the name of each receiving subsystem. The		

Table 154. Keywords used in the Opstat Instruction (continued)

Keyword	Requirement	Description	Abbreviation	Format
		default value 0 will not generate trace records.		
WSNAME()	Optional	The name of the workstation for which you are reporting the status of an operation.	W	

The SUBSYS keyword defaults to the IBM Z Workload Scheduler subsystem name (tracker) specified in the OCL SUBSYS initialization parameter

Example:

```
OPSTAT W(BDEC) ST(C) J(DNCD3000) A(ACLMSDLY)
```

In this example an operation for application ACLMSDLY at workstation BDEC is reported as completed.

The OPSTAT instruction returns one of the following return codes:

RESULT = 0

Operation status modified

RESULT = 8

Invalid instruction

PROMPTN

The PROMPTN instruction specifies that NO is the reply to a recovery prompt issued for an abended operation.

Table 155: Keywords used in the PROMPTN Instruction on page 341 describes the keywords that can be used.

Table 155. Keywords used in the PROMPTN Instruction

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
APPL()	Required	No	The name of the application to be modified.		
OPNO()	Optional	Yes	The number of the operation to be modified.	OP	
IADATE()	Optional (1)	No	The input arrival date of the application occurrence. IADATE(=)	IAD	YYMMDD

Table 155. Keywords used in the PROMPTN Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
			assumes the current date. IADATE() and IA() are mutually exclusive.		
IATIME	Optional (1)	No	The input arrival time of the application occurrence. If IADATE(=) is specified, IATIME() assumes that the default input arrival time is specified in the DEFIAT parameter of the EQQYRPRM member. IADATE() and IA() are mutually exclusive.	IAT	HHMM
IA()	Optional (1)	No	The input arrival date and time of the application occurrence. IA(=) assumes the current date and time. It defaults to the earliest or latest input arrival date and time of the application occurrence occurrences found in the current plan, depending on the SORT parameter. IA() cannot be used together with IADATE() or IATIME().		YYMMDDHHMM
Note: 1 Mutually exclusive with another keyword.					

Examples:

PROMPTN APPL(TEST01)

Specifies that NO is the reply to a recovery prompt issued for any of the operations of application occurrence TEST01

PROMPTN APPL(TEST01) OP(10,30,50)

Specifies that NO is the reply to a recovery prompt issued for operations 10, 30, and 50 of application occurrence TEST01

PROMPTN APPL(TEST01) IAD(=)

Specifies that NO is the reply to a recovery prompt issued for any of the operations of application occurrence TEST01 for the current input arrival date and the default input arrival time

PROMPTN APPL(TEST01) IAD(=) IAT(1800)

Specifies that NO is the reply to a recovery prompt issued for any of the operations of application occurrence TEST01 for the current input arrival date and input arrival time 18.00

PROMPTN APPL(TEST01) IAD(970708)

Specifies that NO is the reply to a recovery prompt issued for any of the operations of application occurrence TEST01 for the input arrival date 970708 and the default input arrival time

PROMPTN APPL(TEST01) IA(=) OP(30)

Specifies that that NO is the reply to a recovery prompt issued for operation 30 of application occurrence TEST01 for the current input arrival date and time

PROMPTN APPL(TEST01) IA(9707081801)

Specifies that NO is the reply to a recovery prompt issued for any of the operations of application occurrence TEST01 for the input arrival date 970708 and the input arrival time 18.01

The PROMPTN instruction returns one of the following return codes:

RESULT = 0

Operations status modified.

RESULT = 4

Occurrence or operations not found in the current plan.

RESULT = 8

Invalid instruction or PIF problem. Refer to the error messages.

PROMPTY

The PROMPTY instruction specifies that YES is the reply to a recovery prompt issued for an abended operation.

Table 156: Keywords used in the PROMPTY Instruction on page 344 describes the keywords that can be used.

Table 156. Keywords used in the PROMPTY Instruction

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
APPL()	Required	No	The name of the application to be modified.		
OPNO()	Optional	Yes	The number of the operation to be modified.	OP	
IADATE()	Optional (1)	No	The input arrival date of the application occurrence. IADATE(=) assumes the current date. IADATE() and IA() are mutually exclusive.	IAD	YYMMDD
IATIME	Optional (1)	No	The input arrival time of the application occurrence. If IADATE(=) is specified, IATIME() assumes that the default input arrival time is specified in the DEFIAT parameter of the EQQYRPRM member. IADATE() and IA() are mutually exclusive.	IAT	HHMM
IA()	Optional (1)	No	The input arrival date and time of the application occurrence. IA(=) assumes the current date and		YYMMDDHHMM

Table 156. Keywords used in the PROMPTY Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
			time. It defaults to the earliest or latest input arrival date and time of the application occurrence occurrences found in the current plan, depending on the SORT parameter. IA() cannot be used together with IADATE() or IATIME().		
Note: 1 Mutually exclusive with another keyword.					

Examples:

PROMPTY APPL(TEST01)

Specifies that YES is the reply to a recovery prompt issued for any of the operations of application occurrence TEST01 that ended in error with return code different from FAIL.

PROMPTY APPL(TEST01) OP(10,30,50)

Specifies that YES is the reply to a recovery prompt issued for operations 10, 30, and 50 of application occurrence TEST01 if they ended in error with return code different from FAIL.

PROMPTY APPL(TEST01) IAD(=)

Specifies that YES is the reply to a recovery prompt issued for any operation of application occurrence TEST01, for the current input arrival date and the default input arrival time, that ended in error with return code different from FAIL.

PROMPTY APPL(TEST01) IAD(=) IAT(1800)

Specifies that YES is the reply to a recovery prompt issued for any operation of application occurrence TEST01, for the current input arrival date and input arrival time 18.00, that ended in error with return code different from FAIL.

PROMPTY APPL(TEST01) IAD(970708)

Specifies that YES is the reply to a recovery prompt issued for any operation of application occurrence TEST01, for the input arrival date 970708 and the default input arrival time, that ended in error with return code different from FAIL.

PROMPTY APPL(TEST01) IA(=) OP(30)

Specifies that YES is the reply to a recovery prompt issued for operation 30 of application occurrence TEST01, for the current input arrival date and time, if it ended in error with return code different from FAIL.

PROMPTY APPL(TEST01) IA(9707081801)

Specifies that YES is the reply to a recovery prompt issued for any operation of application occurrence TEST01, for the input arrival date 970708 and the input arrival time 18.01, that ended in error with return code different from FAIL.

The PROMPTY instruction returns one of the following return codes:

RESULT = 0

Operations status modified.

RESULT = 4

Occurrence or operations not found in the current plan.

RESULT = 8

Invalid instruction or PIF problem. Refer to the error messages.

RELEASE

The RELEASE instruction releases an operation or all the operations of an occurrence that are in HOLD status in the current plan.

Table 157: Keyword used in the Release Instruction on page 346 describes the keywords that can be used.

Table 157. Keyword used in the Release Instruction

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
APPL()	Required	Yes	The name of the application to be modified.		
OPNO()	Optional	Yes	The number of the operation to be modified. If OPNO() is not specified, the RELEASE command, MR, will be issued	OP	

Table 157. Keyword used in the Release Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
			to all the occurrence operations.		
IADATE()	Optional (1)	No	The input arrival date of the application occurrence. IADATE(=) assumes the current date. IADATE() and IA() are mutually exclusive.	IAD	YYMMDD
IATIME	Optional (1)	No	The input arrival time of the application occurrence. If IADATE(=) is specified, IATIME() assumes that the default input arrival time is specified in the DEFIAT parameter of the EQQYRPRM member. IADATE() and IA() are mutually exclusive.	IAT	HHMM
IA()	Optional (1)	No	The input arrival date and time of the application occurrence. IA(=) assumes the current date and time. It defaults to the earliest or latest input arrival date and time of the application occurrence occurrences found in the current plan, depending on the		YYMMDDHHMM

Table 157. Keyword used in the Release Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
			SORT parameter. IA() cannot be used together with IADATE() or IATIME().		
Note: 1 Mutually exclusive with another keyword.					

Examples:

RELEASE APPL(TEST01)

Releases from HOLD status all operations of application occurrence TEST01

RELEASE APPL(TEST01) OP(10,30,50)

Releases from HOLD status operations 10, 30, and 50 of application occurrence TEST01

RELEASE APPL(TEST01) IAD(=)

Releases from HOLD status all operations of application occurrence TEST01 for the current input arrival date and the default input arrival time

RELEASE APPL(TEST01) IAD(=) IAT(1800)

Releases from HOLD status all operations of application occurrence TEST01 for the current input arrival date and input arrival time 18.00

RELEASE APPL(TEST01) IAD(970708)

Releases from HOLD status all operations of application occurrence TEST01 for the input arrival date 970708 and the default input arrival time

RELEASE APPL(TEST01) IA(=) OP(30)

Releases from HOLD status operation 30 of application occurrence TEST01 for the current input arrival date and time

RELEASE APPL(TEST01) IA(9707081801)

Releases from HOLD status all operations of application occurrence TEST01 for the input arrival date 970708 and the input arrival time 18.01

The RELEASE instruction returns one of the following return codes:

RESULT = 0

Operations status modified.

RESULT = 4

Occurrence found but operations not found in the current plan.

RESULT = 8

Occurrence not found in the current plan. Invalid instruction or PIF problem. Refer to the error messages.

RELOP

The RELOP instruction releases an internal successors of an operation within an occurrence in the current plan.

Table 158: Keywords used in the Relop Instruction on page 349 describes the keywords that can be used.

Table 158. Keywords used in the Relop Instruction

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
APPL()	Required	No	The name of the application to be modified.		
OPNO()	Optional	Yes	The number of the operation to be modified.	OP	
IADATE()	Optional[1]	No	The input arrival date of the application occurrence. IADATE=() means the current date. IADATE() and IA() are mutually exclusive.	IAD	YYMMDD
IATIME()	Optional[1]	No	The input arrival time of the application occurrence. If IADATE(=) is specified, IATIME() assumes the default input arrival time that is defined in the DEFIAT parameter of the EQQYRPRM member. IATIME() and IA() are mutually exclusive.	IAT	HHMM
IA()	Optional[1]	No	The input arrival date and time of the		YYMMDDHHMM

Table 158. Keywords used in the Relop Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
			application occurrence. IA(=) assumes the current date and time. It defaults to the earliest or latest input arrival date and time of the application occurrences found in the current plan, depending on the SORT parameter specified in PARMLIB or in the INIT instruction. IA() cannot be used with IADATE() or IATIME().		
EXCLISUC()	Optional	Yes	The operation successor that must not be released.	EISUC	
Note: [1] Mutually exclusive with another keyword.					

Note: Before deleting the dependency between the OPNO() and internal successor operation, the program links the successor operation to the default first operation within the application, which is specified in the DEOPNO variable.

Examples:

RELOP APPL(TEST01) OP(30)

All the internal successor dependencies will be deleted from operation 30 of the occurrence TEST01 with the earliest or latest input arrival date and time, depending on the SORT parameter.

RELOP OP(30) APPL(TEST01) EISUC(60)

All the internal successor dependencies other than operation 60 will be deleted from operation 30 of the occurrence TEST01 with the earliest or latest input arrival date and time, depending on the SORT parameter.

RELOP APPL(TEST01) OP(30) IAD(970708) IAT(1800)

All the internal successor dependencies will be deleted from operation 30 of occurrence TEST01 with the input arrival date 970708 and time 1800.

RELOP APPL(TEST01) EISUC(50,60) OP(30)

All the internal successor dependencies other than 50 and 60 will be deleted from operation 30 of occurrence TEST01 with the earliest or latest input arrival date and time depending on the SORT parameter.

The RELOP instruction returns one of the following return codes:

RESULT = 0

The internal successors have been released.

RESULT = 4

Internal successors not released.

RESULT = 8

Invalid instruction or PIF problem. Refer to the error messages.

RELSUCC

The RELSUCC instruction releases the external successors of an occurrence in the current plan.

Table 159: Keywords used in the Relsucc Instruction on page 351 describes the keywords that can be used.

Table 159. Keywords used in the Relsucc Instruction

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
APPL()	Required	No	The name of the application to be modified.		
OPNO()	Optional	Yes	The number of the operation to be modified. OPNO() and EXCLOP() are mutually exclusive.	OP	
IADATE()	Optional[1]	No	The input arrival date of the application occurrence. IADATE=() means the current date. IADATE() and IA() are mutually exclusive.	IAD	YYMMDD
IATIME()	Optional[1]	No	The input arrival time of the application occurrence. If IADATE(=) is specified,	IAT	HHMM

Table 159. Keywords used in the Relsucc Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
			IATIME() assumes the default input arrival time that is defined in the DEFIAT parameter of the EQQYRPRM member. IATIME() and IA() are mutually exclusive.		
IA()	Optional[1]	No	The input arrival date and time of the application occurrence. IA(=) assumes the current date and time. It defaults to the earliest or latest input arrival date and time of the application occurrences found in the current plan, depending on the SORT parameter specified in PARMLIB or in the INIT instruction. IA() cannot be used with IADATE() or IATIME().		YYMMDDHHMM
EXCLSUC()	Optional	Yes	The application successor that must not be released.	ESUC	
EXCLOP()	Optional[1]	Yes	The operation that must not be modified. EXCLOP() and OPNO() are mutually exclusive.	EOP	

Table 159. Keywords used in the Relsucc Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
Note: [1] Mutually exclusive with another keyword.					

Examples:

RELSUCC APPL(TEST01)

All the successor dependencies will be deleted from occurrence TEST01 with the earliest or latest input arrival date and time, depending on the SORT parameter

RELSUCC APPL(TEST01) IAD(970708) IAT(1800)

All the successor dependencies will be deleted from occurrence TEST01 with the input arrival date 970708 and time 1800

RELSUCC APPL(TEST01) IAD(&OYMD1) EOP(30,35)

All the successor dependencies will be deleted from all the operations, other than 30 and 35, of occurrence TEST01 with the occurrence input arrival date and with the input arrival time corresponding to the default IA time, DEFIAT

RELSUCC APPL(TEST01) IAD(970708) OP(40,50,60,70,,80) ESUC(TEST03,TEST04)

All the successor dependencies other than TEST03 and TEST04 will be deleted from operations 40, 50, 60, 70, and 80 of the occurrence TEST01 with the input arrival date 970708 and with the default input arrival time, DEFIAT

The RELSUCC instruction returns one of the following return codes:

RESULT = 0

The successors have been released.

RESULT = 4

Successors not released.

RESULT = 8

Invalid instruction or PIF problem. Refer to the error messages.

SET

SET assigns a value to a variable, which can be used in the OCL program.

Syntax

SET variable = expression

The expression can contain arithmetic and concatenation operators and built-in functions.

Concatenation operators

The concatenation operators combine two strings to form one string by appending the second string to the right-hand end of the first string. The concatenation might occur with or without an intervening blank. The concatenation operators are:

(blank)

Concatenate terms with one blank in between.

||

Concatenate without an intervening blank.

Built-In Functions

The expression can also contain REXX built-in functions: SUBSTR, LEFT, RIGHT, and OVERLAY. Here are some examples:

```
SET VAR1 = VAR1 !! VAR2 !! 'XX'
(!! is the concatenation character)
SET VAR2 = SUBSTR(&OYMD1,5,2) + 1
SET VAR3 = RIGHT(VAR1,2,'0')
SET VAR4 = LEFT(VAR4,3)
SET VAR5 = OVERLAY('X',VAR4,5)
```

Syntax of SUBSTR built-in function

SUBSTR(*string*,*n*,*length*,*pad*)

The SUBSTR built-in function returns the substring of string that begins at the *n*th character and is of length *length*, padded if necessary with the character *pad*. *n* must be a positive whole number. If *n* is greater than LENGTH(*string*), only padding characters are returned. If you omit *length*, the rest of the string is returned. The default padding character is a blank. Here are some examples:

```
SET VAR1 = SUBSTR('abc',2)          ->    VAR1 = 'bc'
SET VAR1 = SUBSTR('abc',2,4)        ->    VAR1 = 'bc  '
SET VAR1 = SUBSTR('abc',2,6,'.')    ->    VAR1 = 'bc....'
```

Syntax of RIGHT built-in function

RIGHT(*string*,*length*,*pad*)

The RIGHT built-in function returns a string of length *length*, containing the rightmost *length* characters of *string*. The string returned is padded with *pad* characters, or truncated, on the left, as necessary. The default padding character is a blank. *length* must be a positive whole number or zero.

Here are some examples:

```
SET VAR1 = RIGHT('abc  d',8)     ->    VAR1 = '  abc  d'
SET VAR1 = RIGHT('abc def',5)    ->    VAR1 = 'c def'
SET VAR1 = RIGHT('12',5,'0')     ->    VAR1 = '00012'
```

Syntax of LEFT built-in function

LEFT(string,length,pad)

The LEFT built-in function returns a string of length *length*, containing the leftmost *length* characters of *string*. The string returned is padded with *pad* characters, or truncated, on the right as necessary. The default padding character is a blank. *length* must be a positive whole number or zero. The LEFT function is exactly equivalent to SUBSTR(*string*,1,*length*,*pad*)

Here are some examples:

```
SET VAR1 = LEFT('abc d',8)        ->    VAR1 = 'abc d   '
SET VAR1 = LEFT('abc d',8,'.')    ->    VAR1 = 'abc d...'
SET VAR1 = LEFT('abc  def',7)     ->    VAR1 = 'abc  de'
```

Syntax of OVERLAY built-in function

OVERLAY(*new*,*target*,*n*,*length*,*pad*)

The OVERLAY built-in function returns the string *target*, which, starting at the *n*th character, is overlaid with the string *new*, padded or truncated to length *length*. (The overlay might extend beyond the end of the original target string.) If you specify *length*, it must be a positive whole number or zero. The default value for *length* is the length of *new*. If *n* is greater than the length of the target string, padding is added to the left of the *new* string. The default padding character is a blank, and the default value for *n* is 1. If you specify *n*, it must be a positive whole number.

Here are some examples. Assume that VAR1 value is 'ABCDEFGH'.

```
SET VAR1 = OVERLAY(' ',VAR1,3)         -> VAR1 = 'AB DEFGH'
SET VAR1 = OVERLAY('.',VAR1,3,2)       -> VAR1 = 'AB. EFGH'
SET VAR1 = OVERLAY('qq',VAR1)          -> VAR1 = 'qqCDEFGH'
SET VAR1 = OVERLAY('qq',VAR1,4)        -> VAR1 = 'ABCqqFGH'
SET VAR1 = OVERLAY('123',VAR1,5,6,'+') -> VAR1 = 'ABCD123+++'
```

The SET instruction returns one of the following return codes:

CC = 0

Instruction correctly processed

CC = 8

Invalid instruction. See the error message.

SETUPD

Syntax

SETUPD expression

The SETUPD instruction sets the value of a user variable and updates its default value in a variable table. The new default value can be used jobs in an application occurrence that use that variable.

The variable table must be specified by the INIT instruction.

SETUPD is equivalent to both the SET and UPD instructions.

The SETUPD instruction uses the EQQPIFT program, which is in the EQQPIFJV member of the scheduler sample library. You need to specify the CARDIN DD card in the OCL procedure EQQYRPRC.

Examples:

```
SETUPD VAR1 = '00000'
SETUPD VAR1 = SUBSTR(&OYMD1,5,2) + 1
```

According to the previous example, enclose within single quotes any input string or character when specified as value of a user variable.

The SETUPD instruction returns one of the following return codes:

CC = 0

Instruction correctly processed

CC = 4

Variable does not exist in the variable table; it is added

CC = 8

Invalid instruction. See the error messages.

SRSTAT

The SRSTAT instruction invokes the TSO command SRSTAT, which you can use to change the overriding (global) availability, quantity, and deviation of a special resource. You can use it to prevent operations from allocating a particular resource, or to request the ETT function to add an occurrence to the current plan.

Table 160: Keywords used in the Srstat Instruction on page 356 shows the keywords that can be specified.

Table 160. Keywords used in the Srstat Instruction

Keyword	Requirement	Description	Abbreviation
'resource name'	Required	The name of the resource whose availability you want to change. This parameter must be contained within single quotation marks and can be up to 44 characters in length.	
AVAIL()	Optional	It can be one of the following: **YES** Indicates that the availability status of the resource is to be set to YES. **NO** Indicates that the availability status of the resource is to be set to NO.	

Table 160. Keywords used in the Srstat Instruction (continued)

Keyword	Requirement	Description	Abbreviation
		RESET Sets the overriding availability to blank, so that the interval or default value is used. **KEEP** The default, does not change the availability status.	
CREATE()	Optional	It can be one of the following: **NO** If NO is specified, the resource is not added to the current plan of the receiving subsystem if it does not exist in the database. CREATE(NO) does not have any effect. If the resource does not exist in the database, a new resource is created. **YES** If YES is specified or defaulted, and the DYNAMICADD keyword of the RESOPTS initialization statement is set to YES or EVENT, IBM Z Workload Scheduler adds the resource to the current plan of the receiving IBM Z Workload Scheduler subsystem if the resource is not in the database. It uses these values: **Text** Blank. **Specres group ID** Blank. **Hiperbatch** No. **Used for** Control. **On error** Blank. If an error occurs,IBM Z Workload Scheduler uses the	CRE

Table 160. Keywords used in the Srstat Instruction (continued)

Keyword	Requirement	Description	Abbreviation
		value specified in the operation details or, if that is also blank, the value of the ONERROR keyword of RESOPTS. **Overriding availability, quantity, and deviation** The value specified by SRSTAT, or blank. **Default quantity** 1. The default quantity is automatically increased if contention occurs. **Default availability** Yes. **Intervals** No intervals are created. **Workstations** * (All workstations can allocate the resource).	
DEVIATION()	Optional	It can be one of the following: ***amount*** If you want to change the deviation, specify a deviation, which is an amount to be added to (positive number) or subtracted from (negative number) the current quantity. A specified amount can be in the range of -999999 through +999999. **KEEP** The default, does not alter the deviation. **RESET** Sets the deviation to zero.	DEV
LIFESPAN()	Optional	(*interval*, *new_availability_value*)	

Table 160. Keywords used in the Srstat Instruction (continued)

Keyword	Requirement	Description	Abbreviation
		interval The interval of time, in minutes. It can be from 0 to 99999. ***new_availability_values*** The value to which the global availability is changed after interval expiration. It can be YES, NO, or RESET	
QUANTITY()	Optional	It can be one of the following: ***amount*** If you want to change the overriding (global) quantity, specify the amount in the range 1–999999. **RESET** Sets the overriding quantity to blank. so that the interval or default value is used. **KEEP** The default, does not alter the quantity.	Q
SUBSYS()	Optional	The name of the tracker subsystem to which the SRSTAT instruction is directed. It defaults to the subsystem name specified in the OCL OPTRK initialization parameter.	
TRACE()	Optional	Event tracing indicator. When a positive number is specified, a trace entry is created for each event generated by the SRSTAT command. 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 will not generate trace records.	

Example:

```
SRSTAT 'IMS.DATA.BASE' SUBSYS(OPCB) AVAIL(YES)
```

In this example the availability status of the resource IMS™.DATA.BASE is changed to YES.

The SRSTAT instruction returns one of the following return codes:

RESULT = 0

Special resource availability modified

RESULT = 8

Invalid instruction

UNNOP

The UNNOP instruction restores an operation from the NOP status.

Table 161: Keywords used in the Unnop Instruction on page 360 describes the keywords that can be used.

Table 161. Keywords used in the Unnop Instruction

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
APPL()	Required	Yes	The name of the application to be modified.		
OPNO()	Optional	Yes	The number of the operation to be modified. If OPNO() is not specified, the UNNOP command, UN, will be issued to all the occurrence operations.	OP	
IADATE()	Optional[1]	No	The input arrival date of the application occurrence. IADATE(=) assumes the current date. IADATE() and IA() are mutually exclusive.	IAD	YYMMDD
IATIME	Optional[1]	No	The input arrival time of the application occurrence. If IADATE(=) is specified, IATIME() assumes that the default input arrival time is specified in the DEFIAT parameter of the EQQYRPRM member.	IAT	HHMM

Table 161. Keywords used in the Unnop Instruction (continued)

Keyword	Requirement	Multiple Values Allowed	Description	Abbreviation	Format
			IADATE() and IA() are mutually exclusive.		
IA()	Optional[1]	No	The input arrival date and time of the application occurrence. IA(=) assumes the current date and time. It defaults to the earliest or latest input arrival date and time of the application occurrences found in the current plan, depending on the SORT parameter. IA() cannot be used together with IADATE() or IATIME().		YYMMDDHHMM

Note: [1] Mutually exclusive with another keyword.

Example control statements:

UNNOP APPL(TEST01)

Restores all the operations of occurrence TEST01 from the NOP status

UNNOP APPL(TEST01) OP(10,30,50)

Restore operations 10, 30, and 50 of occurrence TEST01 from the NOP status

The UNNOP instruction returns one of the following return codes:

RESULT = 0

Operations status modified.

RESULT = 4

Occurrence found but operations not found in the current plan.

RESULT = 8

Occurrence not found in the current plan. Invalid instruction or PIF problem. Refer to the error messages.

UPD

The UPD instruction changes the default value of a user variable in a variable table of the scheduler. The new default value can be used by jobs in an application occurrence that use that variable. The variable table must be identified by the INIT instruction.

Syntax

UPD *variable*

The UPD instruction uses the EQQPIFT program, which is supplied with IBM Z Workload Scheduler and is in the EQQPIFJV member of the scheduler sample library. You need to specify the CARDIN DD card in the OCL procedure EQQYRPRC.

If the variable exists in the variable table, the UPD instruction changes the default value; otherwise it adds the variable and the variable's default value to the variable table.

Example:

```
UPD VAR2
```

Enclose within single quotes any input string or character when specified as value of a user variable.

The UPD instruction returns one of the following return codes:

CC = 0

Instruction correctly processed

CC = 4

Variable did not exist in the variable table; it has been added

CC = 8

Invalid instruction. See the error message.

WSSTAT

The WSSTAT instruction invokes the TSO command WSSTAT, which you can use to change the status of a workstation in the current plan. You can also establish or close the connection of a workstation to the network. The status information is communicated to the Z controller to indicate a workstation as active, off-line, or failed. When you use the WSSTAT instruction to report a workstation status of offline or failed, you can optionally define restart and routing options for the workload defined on the workstation. You can also change the domain manager of a workstation.

Table 162: Keywords used in the Wsstat Instruction on page 363 shows the keywords that can be used.

Table 162. Keywords used in the Wsstat Instruction

Keyword	Requirement	Description
ALTWS()	Optional	The alternate workstation name. When the workstation status is set to offline or failed, you can specify the alternate workstation where reroutable operations are to be started.
CMD()	Optional	When you want to change the workstation status, you can specify one of the following for CMD(): **L** Link **P** Stop **S** Start **U** Unlink
MANAGES()	Optional	When you want to modify the domain manager of a workstation, you can specify the new domain manager with MANAGES().
REROUTE()	Optional	When the workstation status is set to offline or failed, you can specify one of the following for REROUTE(): **R** To reroute operations to the alternate workstation **L** To leave the operations at the inactive workstation
STARTOPR()	Optional	When the workstation status is set to offline or failed, you can specify what the scheduler is to do with operations that are currently in started status at the workstation: **R** Restart operations automatically on the alternate workstation. **E** Set all started operations to ended-in-error status **L** Leave the operations in started status.
STATUS()	Required	The status you want to report for the workstation:

Table 162. Keywords used in the Wsstat Instruction (continued)

Keyword	Requirement	Description
		A Active **O** Offline **F** Failed
TRACE()	Optional	Event tracing indicator. When a nonzero positive number is specified, a trace entry is created for each event generated by the WSSTAT command. 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 will not generate trace records.
WSNAME()	Required	The name of the workstation to be updated.

Examples:

```
WSSTAT WSNAME(AS4H) STATUS(O) START(R)
```

In this example, the status of workstation AS4H is set to offline. Started operations will be restarted on the alternate workstation.

Example

```
WSSTAT WSNAME(AS4H) SUBSYS(OPCC) MANAGES(DOMAIN1)
```

In this example, the workstation AS4H changes its domain manager to DOMAIN1

The WSSTAT instruction returns one of the following return codes:

RESULT = 0

Workstation status modified

RESULT = 8

Invalid instruction

WTO

The WTO instruction displays messages on the system console and waits for a reply.

Example:

```
WTO Reply YES, SI, OK or press enter to continue
```

Note: OCL uses the IPOWTO program, which is provided as a sample program with IBM Z Workload Scheduler. The message to be displayed by this program is written in member WTOIN in the EQQMISC data set. You therefore need not specify the WTOIN DD card in the OCL procedure.

The WTO instruction returns one of the following return codes:

CC = 0

Valid reply

CC = 8

Reply is not YES, SI, OK, or enter

Requirements

The requirements for OCL are as follows:

- IBM Z Workload Scheduler
- IBM® Library for SAA® REXX/370 Version 1 Release 3
- IPOWTO program provided in the EQQOCWTO member of the IBM Z Workload Scheduler sample library.
- EQQPIFT program provided in the EQQPIFJV member of the scheduler sample library

Note: All libraries referred to in the STEPLIB DD card in the OCL procedure must be APF-authorized.

Sample job and procedure

This section provides an example of a job, EQQYRJCL, and an example of a procedure, EQQYRPRC.

EQQYRJCL sample job

Example

```
//A JOB CARD ACCORDING TO YOUR INSTALLATION STANDARDS IS REQUIRED
//*
//* THIS JOB RUNS THE EQQOCL PROGRAM.
//*
//* THERE IS ONE STEP IN THIS JOB:
//*
//* EQQOCL:    INVOKES THE EQQOCL REXX COMPILED PROGRAM
//*
//* IN ORDER TO USE THIS JOB SUCCESSFULLY YOU SHOULD MODIFY IT
//* AS FOLLOWS:
//* 1. REPLACE THE JOBCARD WITH A VALID JOBCARD FOR YOUR
//*    INSTALLATION
//* 2. LOCATE DATA SET NAMES BEGINNING WITH OPCA.INST AND REPLACE
//*    WITH DATA SET NAMES VALID FOR YOUR INSTALLATION.
//* 3. WRITE YOUR OCL INSTRUCTIONS IN THE SYSIN CARD
//*
//* MOREOVER YOU SHOULD CONSIDER THE FOLLOWING INSTRUCTIONS:
//* 4. UPDATE THE EQQYRPRC PROCEDURE IN THE SCHEDULER SAMPLE LIBRARY
```

```
//*    ACCORDING TO THE INSTRUCTIONS GIVEN IN ITS PROLOG
//* 5. THE SAMPLE PROGRAM EQQRXSTG DELIVERED WITH THE SCHEDULER MUST BE
//*    AVAILABLE
//* 6. IF THE WTO INSTRUCTION IS USED, THE IPOWTO SAMPLE PROGRAM
//*    DELIVERED WITH IBM Z Workload Scheduler MUST BE AVAILABLE
//* 7. IF THE UPD INSTRUCTION IS USED, THE EQQPIFT SAMPLE PROGRAM
//*    DELIVERED WITH THE SCHEDULER IN THE EQQPIFJV MEMBER OF SAMPLE LIBRARY
//*    MUST BE AVAILABLE
//*
//* NOTE THAT THIS JOB ASSUMES THAT THE SCHEDULER HAS BEEN INSTALLED
//* AND THAT THE SMP ACCEPT FUNCTION HAS BEEN PERFORMED.
//*
//MYJCLLIB JCLLIB ORDER=OPCA.INST.SEQQSAMP
//*
//EQQOCL   EXEC  EQQYRPRC
//SYSPRINT DD SYSOUT=*,DCB=(RECFM=FB,LRECL=133,BLKSIZE=1330)
//SYSTSPRT DD SYSOUT=*
//EQQOCL.SYSIN    DD *
* SPECIFY YOUR OCL INSTRUCTIONS
* …
```

EQQYRPRC sample procedure

Example

```
//EQQYRPRC PROC
//*********************************************************************
//*
//* THIS PROCEDURE IS USED BY EQQYRJCL SAMPLE THAT RUNS THE SCHEDULER
//* CONTROL LANGUAGE
//*
//* IN ORDER TO USE THIS JOB SUCCESSFULLY YOU SHOULD MODIFY IT
//* AS FOLLOWS:
//* 1. LOCATE DATA SET NAMES BEGINNING WITH OPCA.INST AND REPLACE WITH
//*    DATA SET NAMES VALID FOR YOUR INSTALLATION.
//*
//* MOREOVER YOU SHOULD CONSIDER THE FOLLOWING INSTRUCTIONS:
//* 3. MAKE SURE THAT THE REXX/370 RUNTIME LIBRARIES ARE AVAILABLE TO
//*    TSO/E (IRXCMPTM TABLE IS DEFINED)
//* 4. THE SAMPLE PROGRAM EQQRXSTG DELIVERED WITH THE SCHEDULER MUST BE
//*    AVAILABLE
//* 5. IF THE WTO INSTRUCTION IS USED, MAKE SURE THAT THE
//*    SAMPLE PROGRAM EQQOCWTO DELIVERED WITH IBM Z Workload Scheduler
//*    IS AVAILABLE
//* 6. IF THE UPD INSTRUCTION IS USED, MAKE SURE THAT THE PL/I RUNTIME
//*    LIBRARIES ARE AVAILABLE, AND THAT THE EQQPIFT SAMPLE PROGRAM
//*    (CONTAINED IN EQQPIFJV SAMPLE) IS AVAILABLE.
//*
//* REQUIRED DATA SETS:
//*
//*   STEPLIB  = THE SCHEDULER LOAD LIBRARY
//*   SYSEXEC  = OCL MODULE
//*   OCLPARM  = OCL PARAMETER LIBRARY
//*   OCLMLIB  = OCL MESSAGE LIBRARY
//*   OCLLOG   = OCL LOG FILE (IF NEEDED); IT MUST BE LRECL 133 AND
//*              ALLOCATED WITH DISP=MOD
//*   EQQMLIB  = THE SCHEDULER MESSAGE LIBRARY
//*   EQQMLOG  = THE SCHEDULER MESSAGE LOG
```

```
//*   EQQYPARM = THE SCHEDULER PARAMETERS FOR PIF REQUESTS
//*
//*****************************************************************
//*
//EQQOCL   EXEC  PGM=IKJEFT01,PARM='EQQOCL'
//STEPLIB  DD DISP=SHR,DSN=OPCA.INST.SEQQLMD0                 <== CHANGE
//         DD DISP=SHR,DSN=PLI_RUNTIME_LIBRARY                <== CHANGE
//OCLLOG   DD DISP=MOD,DSN=OCL_LOG_FILE                       <== CHANGE
//OCLPARM  DD DISP=SHR,DSN=OPCA.INST.SEQQSAMP(EQQYRPRM)       <== CHANGE
//OCLMLIB  DD DISP=SHR,DSN=OPCA.INST.SEQQSAMP(EQQYRMSG)       <== CHANGE
//SYSEXEC  DD DISP=SHR,DSN=OPCA.INST.SEQQMISC                 <== CHANGE
//SYSPRINT DD SYSOUT=*,DCB=(RECFM=FB,LRECL=133,BLKSIZE=1330)
//CARDIN   DD UNIT=SYSDA,SPACE=(TRK,(20,200)),                <== CHANGE
//            DCB=(RECFM=FB,LRECL=80,BLKSIZE=3120)
//SYSTSPRT DD SYSOUT=*
//SYSTSIN  DD DUMMY
//EQQMLIB  DD DISP=SHR,DSN=OPCA.INST.SEQQMSG0                 <== CHANGE
//EQQYPARM DD DISP=SHR,DSN=OPCA.INST.PARM(INIT)               <== CHANGE
//EQQMLOG  DD SYSOUT=*
//EQQDUMP  DD SYSOUT=*
```

Messages

Message format

OCL messages have the format: EQQCL*xxCtext* where:

EQQCL

The message prefix.

xx

The message identifier, which is an alphanumeric value in the range 0–9, A–Z: 00,01,02–09, 0A, 0B–0Y, 10, 11–19, 1A…

C

The message severity code, which can have one of the following values:

I

Information message. Processing continues and, in almost all cases, no action is required of the user.

W

Warning message. Processing continues and, in most cases, no action in required of the user.

E

Error message. Processing terminates; action is required of the user.

text

The message text, which can vary in length. The text can contain message variables that are substituted at run time or when the message is issued. In this book, message variables are shown in italic text.

The OCL messages are written to the OCL output data set, SYSTSPRT, and in the OCL message log data set, if it is allocated in the OCL procedure. The message log data set name is specified in the OCLLOG DD card. The format of an OCL message printed in the message log data set is:

```
mm/dd HH.MM.SS message
```

where:

mm/dd

The current month (*mm*) and day (*dd*).

HH.MM.SS

The current time of day:

HH

The number of hours, starting from midnight. It is a number in the range 00–23.

MM

The number of minutes in the current hour. It is a number in the range 00–59.

SS

The number of seconds in the current minute. It is a number in the range 00–59.

message

The message identifier and text (see the description of OCL messages earlier in this section).

Here is an example of messages in the OCL message log data set:

```
07/30 13:41:33 EQQCL00I Instruction : INIT VARTAB(&OADID)
07/30 13:41:34 EQQCL02I INIT    instruction executed : RC=0
07/30 13:41:34 EQQCL00I Instruction : SETUPD VAR1 = 'PIPPO'
07/30 13:41:34 EQQCL02I SETUPD  instruction executed : RC=0
07/30 13:41:34 EQQCL00I Instruction : COMPL APPL(TEST01)
07/30 13:41:35 EQQCL02E COMPL   instruction executed : RC=8
07/30 13:41:35 EQQCL00I Instruction : CHKAPPL APPL(TEST02)
07/30 13:41:35 EQQCL02W CHKAPPL instruction executed : RC=4
07/30 13:41:35 EQQCL00I Instruction : IF RESULT = 8 THEN NOP
```

Here is an example of the messages written in the SYSTSPRT DD card:

```
EQQCL01I ================================================================
EQQCL00I Processing: CHKAPPL APPL(TEST01) STATUS(S)
EQQCL0JI Searching for the occurrence TEST01 in CP
EQQCL0OI Occurrence found: APPL(TEST01) IA(9709080800) STATUS(S)
EQQCL0OI Occurrence found: APPL(TEST01) IA(9709080930) STATUS(S)
EQQCL0KI Total n. of matching occurrences: 2
EQQCL01I ================================================================
EQQCL00I Processing: IF RESULT=0 THEN ADD APPL(TEST02)
EQQCL03I True condition: IF RESULT = 0
EQQCL01I ================================================================
EQQCL00I Processing: ADD APPL(TEST02)
EQQCL0AI The occurrence was successfully added: APPL(TEST02) IA(9709091657)
```

For explanations of all OCL messages, refer to *"EQQCLnnn Messages"* in *Messages and Codes*.

Chapter 5. Driving IBM® Z Workload Scheduler with REST API

IBM® Z Workload Scheduler provides a set of fully functional APIs that are implemented based on Representational State Transfer (REST) services. The REST API helps you easily integrate workload scheduling capabilities with external products and solutions. The same product functionality covered by the existing J2EE API is available with the REST API. The REST API is programming language independent and favors easier network configuration and firewall traversal. With the APIs, you can exploit heterogeneous environments and provide new automation opportunities with direct impact on productivity. The following are some examples or scenarios where the APIs can be implemented:

- Create your own graphical interface to create and modify scheduling definitions and update objects in the plan.
- Update definitions or plan objects within a script for integration or automation.
- When a specific event occurs within an external product, you can automatically submit a batch workload through IBM® Z Workload Scheduler.
- In a managed file transfer solution, when a specific file arrives, you can submit one or more job flows that elaborate the file, closing the loop on your business process, whether it be bank transactions, a payroll process, or report generation. Your external managed file transfer product starts the business process and IBM® Z Workload Scheduler takes care of the processing, assuring that it be monitored with the rest of the processes from a single point of control and eventually linked with other processes.

The IBM® Z Workload Scheduler REST API provides several services to administer workload modelling and plan.

After installing the Z connector, you can access the available REST API services by connecting to the following URL:

```
https://hostname:port_number/twsz
```

where:

hostname

Host name of the Z connector.

port_number

HTTPS port number of the Z connector. The default is **9443**.

Click **List Operations** to view the operations available with the service, and then click **Expand Operations** to view details such as, the implementation notes, parameters, and response messages for each method. At the end of the details you can find a **Try it out!** button to see the operation in action.

Appendix A. Program interface record format

This appendix describes the fields of the data records handled by the program interface communication routine, EQQYCOM.

These formats are used when information is retrieved by EQQYCOM and provided to the user-written program, and when information is provided by the user program to EQQYCOM to be written to IBM Z Workload Scheduler databases or data sets.

Note: For a correct interpretation of the fields described as "Tod clock at last update", see TOD fields on page 370.

TOD fields

All field in TOD format contain the time-of-day clock value and are set automatically from the system when a replace or insert request is issued. This data is represented in a binary counter corresponding to a 64-bits unsigned integer and its value is implemented every 2-12 microseconds (clock unit) starting from 1st January 1900 with the cycle of the clock of approximately 143 years. In order to understand better the content of this field, please refer to the two tables below:

Table 163. Clock value setting at the start of different years

YEAR	CLOCK SETTNG (HEX NOTATION)			
1900	0000	0000	0000	0000
1976	8853	BAF0	B400	0000
1980	8F80	9FD3	2200	0000
1984	96AD	84B5	9000	0000
1988	9DDA	6997	FE00	0000
1992	A507	4E7A	6C00	0000
1996	AC34	335C	DA00	0000
2000	B361	183F	4800	0000

Table 164. Clock value setting at different time interval

INTERVAL	CLOCK UNIT (HEX ROTATION)			
1 microsec.				1000
1 millisec.			3E	8000
1 second			F424	0000
1 minute		39	3870	0000
1 hour		D69	3A40	0000
1 day	1	41DD	7600	0000

Table 164. Clock value setting at different time interval (continued)

INTERVAL	CLOCK UNIT (HEX ROTATION)			
365 days	1CA	E8C1	3E00	0000
366 days	1CC	2A9E	B400	0000
1.461 days (*)	72C	E4E2	6E00	1000

Application description (resource codes AD, ADCOM)

An application description record can contain these segments:

ADCOM

Common segment. Only one common segment must appear as the first segment in each record.

ADAPD

Application dependency segment.

ADCIV

Interval definition for conditional external predecessor segment.

ADDEP

Dependency segment.

ADCNC

Condition segment.

ADCNS

Condition dependency segment.

ADEXT

Extended name segment.

ADKEY

Key segment.

ADLAT

Operation user-defined late information segment.

ADOP

Operation segment.

ADRE

Remote job information segment.

ADRUN

Run cycle segment.

ADSAI

Operation system automation information segment.

ADSR

Special resource segment.

ADUSF

User field segment.

ADVDD

Operation variable values segment.

ADXIV

Interval definition for external predecessor.

Note: For a correct interpretation of the fields described as "TOD clock at last update", see TOD fields on page 370.

ADAPD - Application dependency segment

The application dependency part of an Application Description.

Table 165. ADAPD Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE		ADAPD	APPLICATION DEPENDENCY SECTION OF AD
0	(0)	CHARACTER	16	ADAPDADID	APPLICATION PREDECESSOR | 'BLANK'
16	(10)	CHARACTER	4	ADAPDWSID	WORKSTATION NAME
20	(14)	SIGNED	4	ADAPDOPNO	OPERATION NUMBER
24	(18)	CHARACTER	4	*	FREE
28	(1C)	CHARACTER	50	ADAPDDESC	DESCRIPTION
78	(4E)	CHARACTER	1	ADAPDLTP	LTP REPORT PRINT OPTION A | C
79	(4F)	CHARACTER	1	ADAPDVERS	RECORD VERSION NUMBER = 1
80	(50)	UNSIGNED	1	ADAPDFLAG	FLAGS
81	(51)	CHARACTER	1	ADAPDCSEL	RESOLUTION CRITERIA C|S|R|A
82	(52)	CHARACTER	1	*	FREE
83	(53)	CHARACTER	1	ADAPDIVTYPE	INTERVAL TYPE R|A (RELATIVE|ABSOLUTE)
84	(54)	CHARACTER	1	ADAPDIVFWHE	FROM WHEN B|A (BEFORE|AFTER)

Table 165. ADAPD Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
85	(55)	CHARACTER	3	ADAPDIVFHHH	FROM HOURS HHH (ONLY RELATIVE INTERVAL)
88	(56)	CHARACTER	2	ADAPDIVFHH	FROM HOURS HHH (ONLY ABSOLUTE INTERVAL)
90	(5A)	CHARACTER	2	ADAPDIVFMM	FROM MINUTES MM
92	(5C)	CHARACTER	1	ADAPDIVFD	FROM DAYS (ONLY ABSOLUTE INTERVAL)
93	(5D)	CHARACTER	1	ADAPDIVTWHE	TO WHEN B|A (BEFORE|AFTER)
94	(5E)	CHARACTER	3	ADAPDIVTHHH	TO HOURS HHH (ONLY RELATIVE INTERVAL)
97	(61)	CHARACTER	2	ADAPDIVTHH	TO HOURS HHH (ONLY ABSOLUTE INTERVAL)
99	(63)	CHARACTER	2	ADAPDIVTMM	TO MINUTES MM
101	(65)	CHARACTER	1	ADAPDIVTD	TO DAYS (ONLY ABSOLUTE INTERVAL)
102	(66)	CHARACTER	2	*	FREE

ADCIV - Interval definition for conditional external predecessor segment

The interval definition for a conditional external predecessor. Used when ADCNS ADCNSCCSEL has value R or A (only one ADCIV per ADCNS can be used, but the same ADCIV can be used by more ADCNS segments if they refer to the same external predecessor application and operation).

Table 166. ADCIV Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	47	ADCIV	CONDITIONAL EXTERNAL PREDECESSOR INTERVAL
0	(0)	CHARACTER	16	ADCIVADID	PREDECESSOR APPLICATION NAME
16	(10)	SIGNED	4	ADCIVCID	PREDECESSOR CONDITION ID
20	(14)	SIGNED	4	ADCIVOPNO	PREDECESSOR OPERATION NUMBER
24	(18)	SIGNED	4	ADCIVOWNOP	OWNING OPERATION
28	(1C)	CHARACTER	1	ADCIVTYPE	INTERVAL TYPE R/A (RELATIVE/ABSOLUTE)
29	(1D)	CHARACTER	1	ADCIVFWHE	FROM WHEN B/A (BEFORE/AFTER)
30	(1E)	CHARACTER	3	ADCIVFHHH	FROM HOURS HHH (ONLY RELATIVE INTERVAL)
33	(21)	CHARACTER	2	ADCIVFHH	FROM HOURS HH (ONLY ABSOLUTE INTERVAL)

Table 166. ADCIV Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
35	(23)	CHARACTER	2	ADCIVFMM	FROM MINUTES MM
37	(25)	CHARACTER	1	ADCIVFD	FROM DAYS (ONLY ABSOLUTE INTERVAL)
38	(26)	CHARACTER	1	ADCIVTWHE	TO WHEN B/A (BEFORE/AFTER)
39	(27)	CHARACTER	3	ADCIVTHHH	TO HOURS HHH (ONLY RELATIVE INTERVAL)
42	(2A)	CHARACTER	2	ADCIVTHH	TO HOURS HH (ONLY ABSOLUTE INTERVAL)
44	(2C)	CHARACTER	2	ADCIVTMM	TO MINUTES MM
46	(2E)	CHARACTER	1	ADCIVTD	TO DAYS (ONLY ABSOLUTE INTERVAL)

ADCOM - Common segment

The common part of an application description.

The reserved fields marked by an * in the name column should be treated as record data. Their value should be preserved when a record is updated and set to zero when a new segment is created.

Table 167. ADCOM Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	192	ADCOM	COMMON SECTION OF AD
0	(0)	CHARACTER	23	ADKEY	KEY
0	(0)	CHARACTER	16	ADID	APPLICATION ID
16	(10)	CHARACTER	1	ADSTAT	APPLICATION STATUS A = ACTIVE, P = PENDING
17	(11)	CHARACTER	6	ADTO	VALID-TO DATE
23	(17)	CHARACTER	1	*	RESERVED
24	(18)	CHARACTER	1	ADTYPE	APPLICATION TYPE A = APPLICATION, G = GROUP DEF.
25	(19)	CHARACTER	1	ADMONITOR	MONITOR AD
26	(1A)	CHARACTER	6	ADFROM	VALID-FROM DATE
32	(20)	CHARACTER	24	ADDESC	DESCRIPTIVE TEXT
56	(38)	CHARACTER	8	ADGROUP	AUTHORITY GROUP NAME

Table 167. ADCOM Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
64	(40)	CHARACTER	16	ADOWNER	OWNER ID
80	(50)	CHARACTER	24	ADODESC	OWNER DESCRIPTION
104	(68)	SIGNED	4	ADPRIOR	PRIORITY
108	(6C)	CHARACTER	16	ADCAL	CALENDAR
124	(7C)	CHARACTER	6	ADLDATE	DATE LAST UPDATED
130	(82)	CHARACTER	4	ADLTIME	TIME LAST UPDATED
134	(86)	CHARACTER	8	ADLUSER	USERID OF LAST UPDATER
142	(8E)	UNSIGNED	1	ADCOMVERS	RECORD VERSION NUMBER
143	(8F)	CHARACTER	16	ADGROUPID	GROUP DEFINITION ID
159	(9F)	CHARACTER	1	*	RESERVED
160	(A0)	CHARACTER	8	ADLUTS	TOD CLOCK AT LAST UPDATE
168	(A8)	SIGNED	4	ADDSM	DEADLINE SMOOTHING FACTOR
172	(AC)	SIGNED	4	ADDLIM	DEADLINE FEEDBACK LIMIT
176	(B0)	CHARACTER	16	*	RESERVED

ADDEP - Dependency segment

The dependency part of an application description.

Table 168. ADDEP Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	96	ADDEP	DEPENDENCY SECTION OF AD
0	(0)	CHARACTER	16	ADDEPADID	EXTERNAL PREDECESSOR | 'BLANK'
16	(10)	CHARACTER	4	ADDEPWSID	WORKSTATION NAME
20	(14)	SIGNED	4	ADDEPOPNO	OPERATION NUMBER
24	(18)	SIGNED	4	ADDEPOWNOP	OWNING OP (THE SUCCESSOR)
28	(1C)	SIGNED	4	ADDEPTPT	TRANSPORT TIME IN MINUTES
32	(20)	CHARACTER	50	ADDEPDESC	DESCRIPTION

Table 168. ADDEP Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
82	(52)	CHARACTER	1	ADDEPLTP	LTP REPORT PRINT OPTION A|C
83	(53)	UNSIGNED	1	ADDEPVERS	RECORD VERSION NUMBER=1
84	(54)	CHARACTER	8	ADDEPJOBN	JOBNAME (NOT ALWAYS SET)
92	(5C)	CHARACTER	1	ADDEPFLAG	FLAGS
93	(5D)	CHARACTER	1	ADDEPCSEL	RESOLUTION CRITERIA C/S/R/A
94	(5E)	CHARACTER	1	ADDEPXMAND	IS MANDATORY N/P/C
95	(5F)	CHARACTER	1	*	FREE

ADCNC - Condition segment

An operation condition.

Example

```
Offsets      Type     Length  Name          Description
    0    (0) STRUCTURE    56  ADCNC         AD OPERATION CONDITION
    0    (0) SIGNED        4  ADCNCOWNID    OWNING AD OPERATION
    4    (4) SIGNED        4  ADCNCID       CONDITION ID
    8    (8) SIGNED        4  ADCNCSIMPNO   NUMBER OF CONDITION DEPENDENCIES
   12    (C) CHARACTER     1  *             NOT USED
   13    (D) UNSIGNED      1  ADCNCVERS     VERSION
   14    (E) CHARACTER     2  *             FREE
   16   (10) SIGNED        4  ADCNCCOUNT    RULE TYPE:
                                            0 = ALL
                                            N>0 = AT LEAST N OF
   20   (14) CHARACTER    24  ADCNCDESC     OPERATION DESCRIPTION
   44   (2C) CHARACTER    12  *             FREE
```

ADCNS - Condition dependency segment

An operation condition dependency.

Example

```
Offsets      Type     Length  Name          Description
    0    (0) STRUCTURE    74  ADCNS         AD OPERATION CONDITION DEPENDENCY
    0    (0) SIGNED        4  ADCNSOWNID    OWNING AD OPERATION
    4    (4) SIGNED        4  ADCNSID       CONDITION ID
    8    (8) CHARACTER    24  ADCNSPREDID   PREDECESSOR ID:
    8    (8) CHARACTER    16  ADCNSPREAD
   24   (18) CHARACTER     8  ADCNSPREOP
   24   (18) CHARACTER     4  ADCNSPREWSID
   28   (1C) SIGNED        4  ADCNSPREOPNO
   32   (20) CHARACTER     1  ADCNSDEPTYP   DEPENDENCY TYPE:
                                            I: INTERNAL
```

```
                                        E: EXTERNAL
33   (21) CHARACTER     2  ADCNSPRETYP   CHECK TYPE:
                                        RC: RETURN CODE
                                        ST: STATUS
35   (23) CHARACTER     2  ADCNSPRELOG   LOGICAL OPERATOR TYPE:
                                        GE: >= GREATER EQUAL
                                        GT: > GREATER
                                        LE: >= LESS EQUAL
                                        LT: > LESS
                                        EQ: = EQUAL
                                        RG: = RANGE
37   (25) CHARACTER     4  ADCNSVALRC    RC VALUE
41   (29) CHARACTER     4  ADCNSVALRC2   RC2 VALUE (FOR RANGE)
45   (2D) CHARACTER     1  ADCNSVALST    ST VALUE:
                                        S: STARTED
                                        C: COMPLETED
                                        X: SUPPRESSED BY CONDITION
                                        E: ERROR
46   (2E) CHARACTER     8  ADCNSPROC     STEP NAME
54   (36) CHARACTER     8  ADCNSSTEP     PROCEDURE INVOCATION STEP NAME
62   (3E) UNSIGNED      1  ADCNSVERS     VERSION
63   (3F) CHARACTER     1  ADCNSCCSEL    RESOLUTION CRITERIA C/S/R/A
64   (40) CHARACTER     1  *             RESERVED
65   (41) CHARACTER     9  *             FREE
```

ADEXT - Extended name segment

The extended name of an operation.

Table 169. ADEXT Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	100	ADEXT	EXTENDED INFORMATION OF AD OPERATION
0	(0)	CHARACTER	54	ADEXTNAME	EXTENDED NAME
54	(36)	UNSIGNED	1	ADEXTVERS	RECORD VERSION NUMBER = 2
55	(37)	CHARACTER	1	*	RESERVED
56	(38)	SIGNED	4	ADEXTOWNOP	OWNING OP NUMBER
60	(3C)	CHARACTER	16	ADEXTSENAME	SCHEDULING ENVIRONMENT NAME
76	(4C)	CHARACTER	24	*	RESERVED

ADKEY - Key segment

The program interface LIST request with the ADKEY resource code lets you get a short version of the ADCOM segment consisting of only the application description key fields. The name of this segment is ADKEY and it contains only the first three fields of the ADCOM segment: ADID, ADSTAT, and ADTO.

ADLAT - Operation user-defined late segment

Late information in operation.

Table 170. ADLAT Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	27	ADLAT	AD OPERATION LATE
0	(0)	SIGNED	4	ADLATOWNID	OWNING AD OPERATION
4	(4)	CHARACTER	1	ADLATVERS	VERSION
5	(5)	CHARACTER	1	ADLAT1BASE	BASEDATE 'F'
6	(6)	CHARACTER	1	ADLAT1DIR	DIRECTION 'A'
7	(7)	CHARACTER	1	*	RESERVED
8	(8)	SIGNED	4	ADLAT1DD	DAY OFFSET FOR THE NOT STARTED ALERT
12	(C)	CHARACTER	4	ADLAT1DT	TIME FOR THE NOT STARTED ALERT
16	(10)	CHARACTER	1	ADLAT2BASE	BASEDATE 'F'
17	(11)	CHARACTER	1	ADLAT2DIR	DIRECTION 'A'
18	(12)	CHARACTER	1	ADLAT2AC	NOT STARTED ACTION A = Only an alert message is issued. C = The operation is set to Complete, if its status allows it. Otherwise it is NOPed. E = The operation is set to Error with OLAT, if its status allows it. Otherwise, this setting is postponed at the time when the status allows it. N = The operation and all its internal successors are NOPed, if their status allows NOPing. Otherwise, it is ignored.
19	(13)	CHARACTER	1	*	RESERVED
20	(14)	SIGNED	4	ADLAT2DD	DAY OFFSET FOR THE NOT STARTED ACTION
24	(18)	CHARACTER	4	ADLAT2DT	TIME FOR THE NOT STARTED ACTION

ADOP - Operation segment

The operation part of an application description.

Note: Certain values are used to show a default or that the field has no value:

ADOPSM = -1

The default should be used.

ADOPLIM = -1

The default should be used.

ADOPHRC = -1

The field is not set.

ADOPHRC = -1

The field is not set.

Table 171. ADOP Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	160	ADOP	OPERATION OF AN AD
0	(0)	CHARACTER	4	ADOPWSID	WORKSTATION
4	(4)	SIGNED	4	ADOPNO	OPERATION NUMBER
8	(8)	CHARACTER	8	ADOPJN	JOBNAME
16	(10)	CHARACTER	24	ADOPDESC	OPERATION DESCRIPTION
40	(28)	SIGNED	4	ADOPDUR	DURATION IN MINUTES
44	(2C)	SIGNED	4	ADOPSM	SMOOTHING FACTOR (OR -1)
48	(30)	SIGNED	4	ADOPLIM	LIMIT FOR FEEDBACK (OR -1)
52	(34)	SIGNED	4	ADOPHRC	HIGHEST OK RC (OR -1)
56	(38)	SIGNED	4	ADOPSTD	RELATIVE DAY INPUT ARRIVAL
60	(3C)	CHARACTER	4	ADOPSTT	INPUT ARRIVAL TIME
64	(40)	SIGNED	4	ADOPDD	RELATIVE DAY DEADLINE
68	(44)	CHARACTER	4	ADOPDT	DEADLINE TIME
72	(48)	SIGNED	4	ADOP#R1	NUMBER OF R1 RESOURCES REQUIRED
76	(4C)	SIGNED	4	ADOP#R2	NUMBER OF R2 RESOURCES REQUIRED
80	(50)	SIGNED	4	ADOP#PS	NUMBER OF SERVERS USED
84	(54)	CHARACTER	1	ADOPJCL	JOB CLASS
85	(55)	CHARACTER	1	ADOPPCL	PRINT CLASS

Table 171. ADOP Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
86	(56)	CHARACTER	8	ADOPFOR	FORM NUMBER
94	(5E)	CHARACTER	1	ADOPSUB	AUTOMATIC SUBMIT Y|N
95	(5F)	CHARACTER	1	ADOPAJR	AUTOMATIC CPU RELEASE Y|N
96	(60)	CHARACTER	1	ADOPCAN	CANCEL IF LATE TIME Y|N
97	(61)	CHARACTER	1	ADOPTIM	SUBMIT JOB ON TIME Y|N
98	(62)	CHARACTER	1	ADOPAEC	AUTOMATIC ERROR COMPL Y|N
99	(63)	UNSIGNED	1	ADOPVERS	RECORD VERSION NUMBER = 2
100	(64)	CHARACTER	1	ADOPWTO	DEADLINE WTO Y|N
101	(65)	CHARACTER	1	ADOPRES	RESTARTABLE Y|N|BLANK
102	(66)	CHARACTER	1	ADOPRER	REROUTEABLE Y|N|BLANK
103	(67)	CHARACTER	1	ADOPCM	RESTART AND CLEANUP A=AUTOMATIC I=IMMEDIATE M=MANUAL N=NONE
104	(68)	CHARACTER	8	ADOPWSINFO	WORKSTATION INFO
104	(68)	CHARACTER	1	ADOPWSISET	INFO AVAILABLE Y|N
105	(69)	CHARACTER	1	ADOPWSTYPE	TYPE G|C|P
106	(6A)	CHARACTER	1	ADOPWSREP	REPORTING ATTRIBUTE A|S|C|N
107	(6B)	CHARACTER	1	ADOPWSSUBT	SUBTYPE JCL, STC, WTO, none J|S|W|blank
108	(6C)	CHARACTER	4	*	RESERVED
112	(70)	CHARACTER	1	ADOPJCRT	(WLM) CRITICAL JOB
113	(71)	CHARACTER	1	ADOPJPOL	(WLM) LATE JOB POLICY
114	(72)	CHARACTER	1	ADOPUSRSYS	USER SYSOUT NEEDED
115	(73)	CHARACTER	1	ADOPEXPJCL	EXPANDED JCL NEEDED
116	(74)	SIGNED	4	ADOPDURI	DURATION IN 100TH OF SEC
120	(78)	CHARACTER	1	ADOPMON	OPERATION MONITORED

Table 171. ADOP Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
121	(79)	CHARACTER	1	*	RESERVED
122	(7A)	CHARACTER	1	ADOPUSEEXT	USE ADEXTNAME FIELD
123	(7B)	CHARACTER	1	ADOPUSESE	USE ADEXTSE FIELD
124	(7C)	CHARACTER	1	ADOPUSESA	USE SYSTEM AUTOMATION Y|N
125	(7D)	CHARACTER	8	ADOPWLMCLASS	WLM SERVICE CLASS
133	(85)	CHARACTER	1	ADOPCONDRJOB	CONDITIONAL RECOVERY JOB
134	(86)	CHARACTER	1	ADOPNOP	NOP JOB
135	(87)	CHARACTER	1	ADOPMH	MANUALLY HOLD JOB
136	(88)	CHARACTER	1	*	RESERVED
137	(89)	CHARACTER	1	ADOPDLACT	DEADLINE ACTION ' ' (blank) = Default. No action is taken. A = Only an alert message is issued. C = The operation is set to Complete, if its status allows it. Otherwise, it is NOPed. E = The operation is set to Error with ODEA, if its status allows it. Otherwise, this setting is postponed at the time when the status allows it. N = The operation and all its internal successors are NOPed, if their status allows NOPing. Otherwise, it is ignored.
138	(8A)	CHARACTER	1	ADOPOLDDUR	OLD DURATION VALUE, Y|N. When an application is modified: Y = The original value for the duration is kept. N = The value for the duration is modified with the new value you set.
139	(8B)	CHARACTER	21	*	RESERVED

ADRE - Remote job information segment

A segment containing the remote job information.

Table 172. ADRE Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	105	ADRE	
0	(0)	CHARACTER	16	ADRE_JSNAME	ADID OR JOB STREAM NAME
16	(10)	UNSIGNED	1	ADRE_VERS	RECORD VERSION NUMBER = 1
17	(11)	CHARACTER	1	ADRE_COMPL	COMPLETE ON FAILED BIND
18	(12)	CHARACTER	2	*	RESERVED
20	(14)	SIGNED	4	ADRE_OPNO	OPERATION NUMBER
24	(18)	CHARACTER	16	ADRE_JSWS	JOB STREAM WORKSTATION
40	(28)	CHARACTER	40	ADRE_JOBNAME	JOB NAME
80	(50)	SIGNED	4	ADRE_OWNOP	OWNING OP NUMBER
84	(54)	CHARACTER	1	*	RESERVED
85	(55)	CHARACTER	1	ADRE_MATCHTYPE	MATCHING CRITERIA: C|S|R|A
86	(56)	CHARACTER	1	*	RESERVED
87	(57)	CHARACTER	1	ADRE_FWHE	FROM WHEN: B=BEFORE IA | A=AFTER IA
88	(58)	CHARACTER	3	ADRE_FHHH	FROM HOURS HHH (ONLY REL)
91	(5B)	CHARACTER	2	ADRE_FHH	FROM HOURS HH (ONLY ABS)
93	(5D)	CHARACTER	2	ADRE_FMM	FROM MINUTES MM
95	(5F)	CHARACTER	1	ADRE_FD	FROM DAYS (ONLY ABS)
96	(60)	CHARACTER	1	ADRE_TWHE	TO WHEN: B=BEFORE IA | A=AFTER IA
97	(61)	CHARACTER	3	ADRE_THHH	TO HOURS HHH (ONLY REL)
100	(64)	CHARACTER	2	ADRE_THH	TO HOURS HH (ONLY ABS)
102	(66)	CHARACTER	2	ADRE_TMM	TO MINUTES MM
104	(68)	CHARACTER	1	ADRE_TD	TO DAYS (ONLY ABS)

ADRUN - Run cycle segment

The run cycle part of an application description. A run cycle is based either on offsets or on rules. The segment contains the fixed part plus either run cycle offsets or a rule definition.

Type

Required input.

For run cycles based on offsets, type is:

N

Normal run cycle that identifies times and days when the application runs.

X

Negative run cycle that identifies times and days when the application does *not* run. If you specify a particular day and time as a negative run cycle, no occurrences of the application are generated for that day and time, regardless of what is generated by a normal or regular run cycle. Run cycles are used in conjunction; negative run cycles are used to suppress run days generated by normal or regular run cycles.

For run cycles based on rules, type is:

R

Regular run cycle that identifies times and days when the application runs.

E

Exclusion run cycle that identifies times and days when the application does *not* run. If you specify a particular day and time as an exclusion run cycle, no occurrences of the application are generated for that day and time, regardless of what is generated by a regular or normal run cycle. Run cycles are used in conjunction; exclusion run cycles are used to suppress run days generated by regular or normal run cycles.

Free day rule

Required input for all run cycles, which indicates how run days are treated:

E

Free days excluded; only work days are taken into account

1

Free days included; run on the nearest day *before* the free day

2

Free days included; run on the nearest day *after* the free day

3

Free days included; run *on* the free day

4

Free days included; do *not* run at all.

Note: ADRIADALL is the start of either run cycle offsets or a rule. EQQPIFAD sample shows how to handle it.

Table 173. ADRUN Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	228	ADRUN	RUNCYCLE SECTION
0	(0)	CHARACTER	8	ADRPER	PERIOD NAME
8	(8)	CHARACTER	6	ADRVALF	RUN CYCLE VALID-FROM
14	(E)	CHARACTER	6	ADRVALT	RUN CYCLE VALID-TO
20	(14)	CHARACTER	50	ADRUNDESC	RUN CYCLE DESCRIPTION
70	(46)	CHARACTER	1	ADRUNRULE	RULE FOR WORK/FREE DAYS
71	(47)	CHARACTER	1	ADRTYPE	PERIOD BASED (N|X) | RULE BASED (R|E)
72	(48)	SIGNED	4	ADRIAD(24)	OFFSETS (START DAYS WITHIN PERIOD)
168	(A8)	CHARACTER	4	ADRIAT	INPUT ARRIVAL TIME
172	(AC)	SIGNED	4	ADRDD	DEADLINE DAY RELATIVE TO START
176	(B0)	CHARACTER	4	ADRDT	DEADLINE TIME
180	(B4)	UNSIGNED	1	ADRUNVERS	RECORD VERSION NUMBER=1
181	(B5)	CHARACTER	16	ADRJVTAB	JCL VARIABLE TABLE
197	(C5)	CHARACTER	1	ADRSHTYPE	SHIFT TYPE (W/D or blank)
198	(C6)	SIGNED	2	ADRINPOS	NUMBER OF POSITIVE RUN CYCLE OFFSETS
200	(C8)	SIGNED	2	ADRINNEG	NUMBER OF NEGATIVE RUN CYCLE OFFSETS
202	(CA)	SIGNED	2	ADRIRDLEN	RULE DEFINITION LENGTH
204	(CC)	CHARACTER	4	ADRREPEATEVRY	REPEAT EVERY
208	(D0)	CHARACTER	4	ADRREPEATENDT	REPEAT END TIME
212	(D4)	SIGNED	4	ADRSHIFT	SHIFT VALUE (-999 to 999)
216	(D8)	CHARACTER	12	*	RESERVED
228	(E4)	CHARACTER	*	ADRIADALL	START OF RUN CYCLE OFFSETS OR A RULE

Table 174. Run Cycle Offsets

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
228	(E4)	STRUCTURE	*	ADRIADALL	START OF RUN CYCLE OFFSETS
228	(E4)	SIGNED	4	ADRIAOFF	ARRAY OF RUN CYCLE OFFSETS (LENGTH=(ADRINPOS+ADRINNEG)*4)

Run cycle offsets are an array of positive fullwords. ADRINPOS and ADRINNEG identify the number of entries in the array. The positive offsets are first.

If the total number of offsets is 24 or less, the offsets are also found in the ADRIAD array. ADRIAD is an array of 24 integer values that specify the start days within the period. Each nonzero value defines a day that the run cycle selects; that is, when the application runs if ADRTYPE is N, or does not run if ADRTYPE is X. The first day of the period is specified by 1 and the last day by -1. The first zero value ends the array.

Table 175. Rule Definition

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
228	(E4)	STRUCTURE	*	ADRIADALL	RULE DEFINITION
228	(E4)	SIGNED	4	ADRULEL	RULE LENGTH (ADRULEL + ADRULET)
232	(E8)	CHARACTER	*	ADRULET	RULE TEXT

For a rule-based run cycle, ADRIRDLEN identifies the length of the rule definition. The ADRIADALL structure contains a fullword copy of ADRIRDLEN (ADRULEL), which is followed by the rule text. ADRULEL must specify the same length as ADRIRDLEN. You can insert comments or extra blanks when creating a rule, but these characters are not saved in the AD database. The syntax of the rule text is the same as for the ADRULE control statement used by the batch loader. For more detailed information, see *Managing the Workload*.

Here is an example of a rule definition, which selects the third day in each month:

```
ADRULEL 33 (X'21')
ADRULET 'ADRULE ONLY(3) DAY(DAY) MONTH'
```

Note: Note that the ADOP segment is enlarged by 32 characters. This will not affect current program interface applications until in a future release, when the reserved field becomes used for operation data.

ADSAI - Operation system automation information segment

System automation information.

Note: This segment exists for system automation operations only.

Table 176. ADSAI Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	352	ADSAI	SYSTEM AUTOMATION INFO FOR AD OPERATION
0	(0)	CHARACTER	256	ADSAICOMMTEXT	SYSTEM AUTOMATION OPERATION COMMAND TEXT
0	(0)	CHARACTER	64	ADSAICOMMTEX1	SYSTEM AUTOMATION OPERATION COMMAND TEXT, ROW 1
64	(40)	CHARACTER	64	ADSAICOMMTEX2	SYSTEM AUTOMATION OPERATION COMMAND TEXT, ROW 2
128	(80)	CHARACTER	64	ADSAICOMMTEX3	SYSTEM AUTOMATION OPERATION COMMAND TEXT, ROW 3
192	(C0)	CHARACTER	63	ADSAICOMMTEX4	SYSTEM AUTOMATION OPERATION COMMAND TEXT, ROW 4
255	(FF)	CHARACTER	1	ADSAIFILLER	RESERVED
256	(100)	CHARACTER	8	ADSAIAUTOOPER	SYSTEM AUTOMATED OPERATOR
264	(108)	CHARACTER	8	ADSAISECELEM	SYSTEM AUTOMATION SECURITY ELEMENT
272	(110)	CHARACTER	64	ADSAICOMPINFO	SYSTEM AUTOMATION COMPLETION INFORMATION
336	(150)	CHARACTER	4	*	RESERVED
340	(154)	SIGNED	4	ADSAIOWNOP	OWNING OPERATION NUMBER
344	(158)	CHARACTER	8	*	RESERVED

ADSR - Special resource segment

The special resource part of an application description.

Table 177. ADSR Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	64	ADSR	SPECIAL RESOURCE SECTION
0	(0)	CHARACTER	44	ADSRN	SPECIAL RESOURCE NAME

Table 177. ADSR Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
44	(2C)	SIGNED	4	ADSROWNOP	OWNING OPERATION NUMBER
48	(30)	CHARACTER	1	ADSRT	S = SHARED, X = EXCLUSIVE
49	(31)	UNSIGNED	1	ADSRVERS	RECORD VERSION NUMBER = 2
50	(32)	CHARACTER	1	ADSRONER	KEEP ON ERROR (Y|N|blank)
51	(33)	CHARACTER	1	*	FREE
52	(34)	SIGNED	4	ADSRAMNT	QUANTITY REQUIRED. THE VALUE 0 MEANS THE TOTAL QUANTITY OF SPECIAL RESOURCE.
56	(38)	CHARACTER	1	ADSRAVACO	ON COMPLETE (Y|N|R|blank)
57	(39)	CHARACTER	7	*	RESERVED

ADUSF - User field segment

An operation user field.

Table 178. ADUSF Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	84	ADUSF	AD OPERATION USER FIELD
0	(0)	SIGNED	4	ADUSFOWNID	OWNING AD OPERATION
4	(4)	CHARACTER	16	ADUSFNAME	USER FIELD NAME
20	(14)	CHARACTER	54	ADUSFVALUE	USER FIELD VALUE
74	(4A)	CHARACTER	2	*	NOT USED
76	(4C)	UNSIGNED	1	ADUSFVERS	VERSION
77	(4D)	CHARACTER	7	*	NOT USED

ADVDD - Operation variable values

The operation variable values, associated with a specific run cycle.

Table 179. ADVDD Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	27	ADDVDD	AD OPERATION VARIABLE VALUES
0	(0)	SIGNED	4	ADVDDOWNID	OWNING AD OPERATION
4	(4)	SIGNED	4	ADVDDDUR	VARIABLE DURATION
8	(8)	CHARACTER	8	ADVDDDEAD	VARIABLE DEADLINE
8	(8)	SIGNED	4	ADVDDDEADD	VARIABLE DEADLINE RELATIVE DAY
12	(C)	CHARACTER	4	ADVDDDEADT	VARIABLE DEADLINE TIME
16	(10)	CHARACTER	8	ADVDDRG	RUN CYCLE GROUP NAME
24	(18)	CHARACTER	1	ADVDNOP	NOP JOB (Y, N, or blank)
25	(19)	CHARACTER	1	ADVDDMH	MANUALLY HOLD JOB (Y, N, or blank)
26	(1A)	CHARACTER	1	ADVDDCRJ	CRITICAL JOB (N, P, W, or blank)
27	(1B)	CHARACTER	1	ADVDDDLACT	DEADLINE ACTION ' ' (blank) = Only an alert message is issued. A = Only an alert message is issued. C = The operation is set to Complete, if its status allows it. Otherwise it is NOPed. E = The operation is set to Error with OLAT, if its status allows it. Otherwise, this setting is postponed at the time when the status allows it. N = The operation and all its internal successors are NOPed, if their status allows NOPing. Otherwise, it is ignored.
28	(1C)	CHARACTER	12	ADVDDLATE1	NOT STARTED ALERT
28	(1C)	CHARACTER	1	ADVDDLATE1BAS	BASEDATE (ALWAYS 'F')
29	(1D)	CHARACTER	1	ADVDDLATE1DIR	DIRECTION (ALWAYS 'A')
30	(1E)	CHARACTER	2	*	RESERVED
32	(20)	SIGNED	4	ADVDDLATE1DD	DAY OFFSET FOR NOT STARTED ALERT
36	(24)	CHARACTER	4	ADVDDLATE1DT	TIME FOR NOT STARTED ALERT
40	(28)	CHARACTER	12	ADVDDLATE2	NOT STARTED ACTION

Table 179. ADVDD Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
40	(28)	CHARACTER	1	ADVDDLATE2BAS	BASEDATE (ALWAYS 'F')
41	(29)	CHARACTER	1	ADVDDLATE2DIR	DIRECTION (ALWAYS 'A')
42	(2A)	CHARACTER	1	ADVDDLATE2AC	NOT STARTED ACTION A = Only an alert message is issued. C = The operation is set to Complete, if its status allows it. Otherwise it is NOPed. E = The operation is set to Error with OLAT, if its status allows it. Otherwise, this setting is postponed at the time when the status allows it. N = The operation and all its internal successors are NOPed, if their status allows NOPing. Otherwise, it is ignored.
43	(2B)	CHARACTER	1	*	RESERVED
44	(2C)	SIGNED	4	ADVDDLATE2DD	DAY OFFSET FOR NOT STARTED ACTION
48	(30)	CHARACTER	4	ADVDDLATE2DT	TIME FOR NOT STARTED ACTION

ADXIV - Interval definition for external predecessor segment

The interval definition for an external predecessor. Used when ADDEP ADDEPCSEL has value R or A (only one ADXIV per ADDEP can be used).

Table 180. ADXIV Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	47	ADXIV	EXTERNAL PREDECESSOR INTERVAL
0	(0)	CHARACTER	16	ADXIVADID	PREDECESSOR APPLICATION NAME
16	(10)	CHARACTER	4	ADXIVWSID	PREDECESSOR WORKSTATION NAME
20	(14)	SIGNED	4	ADXIVOPNO	PREDECESSOR OPERATION NUMBER
24	(18)	SIGNED	4	ADXIVOWNOP	OWNING OPERATION NUMBER
28	(1C)	CHARACTER	1	ADXIVTYPE	INTERVAL TYPE R|A (RELATIVE | ABSOLUTE)

Table 180. ADXIV Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
29	(1D)	CHARACTER	1	ADXIVFWHE	FROM WHEN B|A (BEFORE|AFTER)
30	(1E)	CHARACTER	3	ADXIVFHHH	FROM HOURS HHH (ONLY RELATIVE INTERVAL)
33	(21)	CHARACTER	2	ADXIVFHH	FROM HOURS HH (ONLY ABSOLUTE INTERVAL)
35	(23)	CHARACTER	2	ADXIVFMM	FROM MINUTES MM
37	(25)	CHARACTER	1	ADXIVFD	FROM DAYS (ONLY ABSOLUTE INTERVAL)
38	(26)	CHARACTER	1	ADXIVTWHE	TO WHEN B/A (BEFORE|AFTER)
39	(27)	CHARACTER	3	ADXIVTHHH	TO HOURS HHH (ONLY RELATIVE INTERVAL)
42	(2A)	CHARACTER	2	ADXIVTHH	TO HOURS HH (ONLY ABSOLUTE INTERVAL)
44	(2C)	CHARACTER	2	ADXIVTMM	TO MINUTES MM
46	(2E)	CHARACTER	1	ADXIVTD	TO DAYS (ONLY ABSOLUTE INTERVAL)

All workstations closed (resource code AWSCL)

There is no common segment. One segment exists for each interval when all workstations are closed.

AWSCL - All workstations closed interval segment

Description of an interval when all workstations are closed.

Table 181. AWSCL Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	80	AWSCL	WS CLOSED INTERVAL
0	(0)	CHARACTER	6	AWCKEY	UNIQUE IDENTIFIER
0	(0)	CHARACTER	6	AWCDATE	DATE
6	(6)	CHARACTER	4	AWCFROM	FROM TIME
10	(A)	CHARACTER	4	AWCTO	TO TIME
14	(E)	CHARACTER	30	AWCDESC	DESCRIPTION CLOSED INTERVAL
44	(2C)	UNSIGNED	1	AWCVERS	VERSION OF RECORD=1
45	(2D)	CHARACTER	6	AWCLDATE	DATE LAST UPDATED

Table 181. AWSCL Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
51	(33)	CHARACTER	4	AWCLTIME	TIME LAST UPDATED
55	(37)	CHARACTER	8	AWCLUSER	USERID OF LAST UPDATER
63	(3F)	CHARACTER	1	*	RESERVED
64	(40)	CHARACTER	8	AWCLLUTS	TOD CLOCK AT LAST UPDATE
72	(48)	CHARACTER	8	*	RESERVED

Calendar (resource codes CL, CLCOM)

Each calendar record can contain these segments:

CLCOM

Common segment. Only one common segment must appear as the first segment in each record.

CLSD

Specific date segment.

CLWD

Specific day of week segment.

Note: For a correct interpretation of the fields described as "Tod clock at last update", see TOD fields on page 370.

CLCOM - Common segment

Common description of a calendar.

Table 182. CLCOM Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	96	CLCOM	
0	(0)	CHARACTER	16	CLKEY	UNIQUE IDENTIFIER
0	(0)	CHARACTER	16	CLNAME	CALENDER NAME
16	(10)	SIGNED	4	CLDAYS	NUMBER OF SPECIFIC AND WEEK DAYS
20	(14)	CHARACTER	4	CLSHIFT	END TIME OF A SHIFT
24	(18)	CHARACTER	30	CLDESC	DESCRIPTION

Table 182. CLCOM Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
54	(36)	UNSIGNED	1	CLVERS	VERSION OF RECORD=1
55	(37)	CHARACTER	6	CLLDATE	DATE LAST UPDATED
61	(3D)	CHARACTER	4	CLLTIME	TIME LAST UPDATED
65	(41)	CHARACTER	8	CLLUSER	USER ID OF LAST UPDATER
73	(49)	CHARACTER	7	*	RESERVED
80	(50)	CHARACTER	8	CLLUTS	TOD CLOCK AT LAST UPDATE
88	(58)	CHARACTER	8	*	RESERVED

CLSD - Specific date segment

Calendar description: a specific date.

Day status can be:

W

Work

F

Free

Table 183. CLSD Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	48	CLSD	
0	(0)	CHARACTER	6	CLSDDATE	SPECIFIC DATE
6	(6)	CHARACTER	2	*	RESERVED
8	(8)	CHARACTER	1	CLSDSTAT	STATUS, WORK OR FREE
9	(9)	CHARACTER	30	CLSDDESC	DESCRIPTION OF THE DATE
39	(27)	CHARACTER	9	*	RESERVED

CLWD - Weekday segment

Calendar description: a weekday.

A weekday can be:

MONDAY
TUESDAY
WEDNESDAY
THURSDAY
FRIDAY
SATURDAY
SUNDAY

Note: WEDNESDAY is actually stored as WEDNESDA.

Table 184. CLWD Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	48	CLWD	
0	(0)	CHARACTER	8	CLWDDAY	WEEK DAY
8	(8)	CHARACTER	1	CLWDSTAT	STATUS, WORK OR FREE
9	(9)	CHARACTER	30	CLWDDESC	DESCRIPTION OF THE DATE
39	(27)	CHARACTER	9	*	RESERVED

Current plan condition (resource codes CPCOND, CPCONDCO)

The current plan condition record can contain these segments:

CPCOND

Common segment. Only one CPCOND must be provided.

CPSIMP

Conditional dependency segment.

CPCOND - Condition segment

Current plan operation condition.

Example

```
Offsets      Type     Length  Name          Description
    0    (0) STRUCTURE   134  CPCONDCO      CURRENT PLAN OPERATION CONDITION -
                              KEY FIELDS: ---------------
    0    (0) CHARACTER    16  CPCOADI       APPLICATION ID
   16   (10) CHARACTER    10  CPCOIA        APPLICATION INPUT ARRIVAL
   16   (10) CHARACTER     6  CPCOIAD       MODIFIED IF IA IS MODIFIED
```

```
   22   (16) CHARACTER     4  CPCOIAT        ELSE ORIGINAL FROM PLAN
   26   (1A) SIGNED        4  CPCOOPNO       OPERATION NUMBER
   30   (1E) SIGNED        4  CPCOCID        CONDITION ID
                                            ----------------------------
   34   (22) CHARACTER    24  CPCODESC       CONDITION DESCRIPTION
   58   (3A) CHARACTER     1  *
   59   (3B) CHARACTER     1  *              FREE FOR ALIGNEMENT
   60   (3C) SIGNED        4  CPCO#SIMP      NUMBER OF CONDITION DEPENDENCIES
   64   (40) SIGNED        4  CPCOCOUNT      RULE TYPE:
                                            0 = ALL
                                            N>0 = AT LEAST N OF
   68   (44) CHARACTER     1  CPCOVALUE      FINAL CONDITION STATUS: U:
                                            UNDECIDED T: TRUE F: FALSE
   69   (45) UNSIGNED      1  CPCOVERS       VERSION
   70   (46) CHARACTER     1  CPCOXST        COND EXTENDED STATUS
   71   (47) CHARACTER    63  *              FREE
```

CPSIMP - Condition dependency segment

Current plan operation condition dependency.

Example

```
Offsets      Type     Length  Name           Description
    0    (0) STRUCTURE    85  CPSIMP         CURRENT PLAN OPERATION CONDITION DEPENDENCY
                                            KEY FIELDS: ---------------
    0    (0) CHARACTER    16  CPSIPREADI     APPLICATION ID
   16   (10) CHARACTER    10  CPSIPREIA      APPLICATION INPUT ARRIVAL
   16   (10) CHARACTER     6  CPSIPREIAD     MODIFIED IF IA IS MODIFIED
   22   (16) CHARACTER     4  CPSIPREIAT     ELSE ORIGINAL FROM PLAN
   26   (1A) SIGNED        4  CPSIPREOPNO    OPERATION NUMBER
   30   (1E) CHARACTER     2  CPSITYP        CHECK TYPE: RC OR ST
   32   (20) CHARACTER     2  CPSILOG        OPERATOR: GE, GT, LE, LT, EQ,
                                            NE, RG
   34   (22) CHARACTER     4  CPSIVALRC      RC VALUE
   38   (26) CHARACTER     4  CPSIVALRC2     RC2 VALUE
   42   (2A) CHARACTER     1  CPSIVALST      ST VALUE
                                            ----------------------------
   43   (2B) CHARACTER     1  CPSILVAL       CONDITION DEPENDENCY STATUS: U T F
   44   (2C) UNSIGNED      1  CPSIVERS       VERSION
   45   (2D) CHARACTER     1  CPSIREMOVED    CONDITION DEPENDENCY REMOVED: Y|N
   46   (2E) CHARACTER     1  CPSISTEPMISS   MISSING STEP END INFORMATION: Y|N
   47   (2F) CHARACTER     8  CPSISTEP       PROCEDURE INVOCATION STEP NAME
   55   (37) CHARACTER     8  CPSIPSTEP      STEP NAME
   63   (3F) CHARACTER     8  CPSIJOBNAME    JOB NAME
   71   (47) CHARACTER     4  CPSIWSNAME     WS NAME
   75   (4B) CHARACTER     1  CPSINEWSTAT    NEW STATUS: T F
   76   (4C) CHARACTER     9  *              FREE
```

Current plan occurrence (resource code CPOC, CPOCCOM)

The current plan occurrence record consists always of the following segment:

CPOC

Current plan occurrence common segment. Only one common segment must exist.

It can optionally consist of the following segments:

CPOCPRE

Occurrence predecessor segment.

CPOCSUC

Occurrence successor segment.

CPOC - Current plan occurrence segment

Current plan occurrence.

Note:

1. Minutes are the unit of duration.
2. Y and N are the indicator values.
3. Actual arrival, CPOCAA, for manually completed occurrences is blank, if no operations have started.

ADDING FUNCTION

Blank

The daily plan batch program

A

Automatic recovery

D

Dialog (Modify Current Plan dialog)

E

ETT, event-triggered tracking

P

PIF, program interface

Table 185. CPOC Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	452	CPOC	CURRENT PLAN OCCURRENCE
0	(0)	CHARACTER	16	CPOCADI	APPLICATION ID
16	(10)	CHARACTER	10	CPOCIA	INPUT ARRIVAL
16	(10)	CHARACTER	6	CPOCIAD	MODIFIED IF IA IS MODIFIED

Table 185. CPOC Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
22	(16)	CHARACTER	4	CPOCIAT	ELSE ORIGINAL FROM PLAN
26	(1A)	CHARACTER	8	CPOCGRP	AUTHORITY GROUP
34	(22)	CHARACTER	10	CPOCIAO	INPUT ARRIVAL FROM LTP
34	(22)	CHARACTER	6	CPOCIAOD	DATE
40	(28)	CHARACTER	4	CPOCIAOT	TIME
44	(2C)	CHARACTER	24	CPOCDESC	DESCRIPTIVE TEXT
68	(44)	CHARACTER	16	CPOCOID	OWNER ID
84	(54)	CHARACTER	24	CPOCODES	OWNER DESCRIPTION
108	(6C)	CHARACTER	10	CPOCDL	DEADLINE
108	(6C)	CHARACTER	6	CPOCDLD	DATE
114	(72)	CHARACTER	4	CPOCDLT	TIME
118	(76)	CHARACTER	10	CPOCAA	ACTUAL ARRIVAL
118	(76)	CHARACTER	6	CPOCAAD	IF ARRIVED
124	(7C)	CHARACTER	4	CPOCAAT	ELSE BLANKS
128	(80)	CHARACTER	10	CPOCAC	ACTUAL COMPLETION
128	(80)	CHARACTER	6	CPOCACD	IF COMPLETED
134	(86)	CHARACTER	4	CPOCACT	ELSE BLANKS
138	(8A)	CHARACTER	4	CPOCERR	OCCURRENCE ERROR CODE
142	(8E)	CHARACTER	1	CPOCST	OCCURRENCE STATUS
143	(8F)	CHARACTER	1	CPOCRER	RERUN REQUESTED (Y|N)
144	(90)	CHARACTER	1	CPOCADDED	ADDED TO CURRENT PLAN (Y|N)
145	(91)	CHARACTER	1	CPOCLATE	LATEST OUT PASSED (Y|N)
146	(92)	CHARACTER	1	CPOCADDF	ADDING FUNCTION (E|D|P|A|)
147	(93)	CHARACTER	1	CPOCMON	MONITORING FLAG
148	(94)	SIGNED	4	CPOCPRI	PRIORITY
152	(98)	SIGNED	4	CPOC#OP	NUMBER OF OPERATIONS IN OCCURRENCE
156	(9C)	SIGNED	4	CPOCOPC	NUMBER OF OPERATIONS COMPLETED

Table 185. CPOC Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
160	(A0)	SIGNED	4	CPOC#ER	NUMBER OF OPERATIONS ENDED IN ERROR
164	(A4)	SIGNED	4	CPOC#UN	NUMBER OF OPERATIONS UNDECIDED
168	(A8)	SIGNED	4	CPOC#ST	NUMBER OF OPERATIONS STARTED
172	(AC)	SIGNED	4	CPOCRDU	REMAINING DUR CRITICAL PATH
176	(B0)	SIGNED	4	CPOCROP	REMAINING OPS CRITICAL PATH
180	(B4)	CHARACTER	4	CPOCCWS	WSNAME OF 1ST CRITICAL OP
184	(B8)	SIGNED	4	CPOCCOP	OP NO. OF 1ST CRITICAL OP
188	(BC)	UNSIGNED	1	CPOCVERS	VERSION NUMBER=1
189	(BD)	CHARACTER	16	CPOCJVT	JCL VARIABLE TABLE
205	(CD)	CHARACTER	1	*	RESERVED NOT ADD
206	(CE)	CHARACTER	16	CPGROUPID	GROUP DEFINITION ID
222	(DE)	CHARACTER	16	CPOCCAL	CALENDAR NAME
238	(EE)	CHARACTER	2	*	RESERVED
240	(F0)	UNSIGNED	4	CPOCRDUI	REMAIN. DUR CRIT. PATH SEC
244	(F4)	CHARACTER	4	*	RESERVED
248	(F8)	CHARACTER	8	CPOCOCTO	OCCURRENCE TOKEN
256	(100)	CHARACTER	10	CPOCCLO	FIRST CRITICAL OP LATEST OUT
256	(100)	CHARACTER	6	CPOCCLOD	DATE
262	(106)	SIGNED	4	CPOCCLOT	TIME IN 100TH OF SEC.
266	(10A)	CHARACTER	44	CPOCETTCRIT	ETT CRITERIA
310	(136)	CHARACTER	1	CPOCETTTYP	ETT TYPE: J OR R
311	(137)	CHARACTER	8	CPOCETTJOB	ETT JOB NAME
319	(13F)	CHARACTER	8	CPOCETTJID	ETT JOB ID
327	(147)	CHARACTER	35	CPOCETTGROOT	ETT GDG ROOT
362	(16A)	CHARACTER	44	CPOCETTEVNAM	COMPLETE ETT EVENT NAME
406	(196)	CHARACTER	8	CPOCETTGGEN	ETT GDG GENERATION
414	(19E)	CHARACTER	6	*	RESERVED

Table 185. CPOC Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
420	(1A4)	CHARACTER	8	CPOCRUNC	RUN CYCLE THAT GENERATED THE OCCURRENCE
428	(1AC)	CHARACTER	24	*	RESERVED

CPOCPRE - Occurrence predecessor segment

Current plan occurrence predecessor.

Table 186. CPOCPRE Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	CHARACTER	16	CPOCPREADI	APPLICATION ID
16	(10)	CHARACTER	10	CPOCPREIA	INPUT ARRIVAL
16	(10)	CHARACTER	6	CPOCPREIAD	MODIFIED IF IA IS MODIFIED
22	(16)	CHARACTER	4	CPOCPREIAT	ELSE ORIGINAL FROM PLAN
26	(1A)	SIGNED	4	CPOCPRENO	OPERATION NUMBER
30	(1E)	CHARACTER	1	CPOCPRECO	PREDECESSOR COMPLETED (Y|N)
31	(1F)	CHARACTER	1	CPOCPRENR	PRED. WS WAS NONREPORTING
32	(20)	SIGNED	4	CPOCPRETT	TRANSPORT TIME
36	(24)	CHARACTER	1	CPOCPREND	PENDING PRED
37	(25)	UNSIGNED	1	CPOCPREVERS	VERSION NUMBER=1
38	(26)	CHARACTER	8	CPOCPREJN	PREDECESSOR JOB NAME
46	(2E)	CHARACTER	1	CPOCPREST	PREDECESSOR STATUS
47	(2F)	CHARACTER	1	CPOCPMATC	PREDECESSOR RESOLUTION CRITERIA: BLANK (MANUALLY CHOSEN) C (CLOSEST PRECEDING) S (SAME DAY) A (ABSOLUTE INTERVAL) R (RELATIVE INTERVAL)
48	(30)	SIGNED	4	CPOCPRECRITPATH	PREDECESSOR OF AN OPERATION BELONGING TO A CRITICAL PATH

Table 186. CPOCPRE Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
52	(34)	CHARACTER	21	*	RESERVED PER MAND PEND
73	(49)	CHARACTER	7	*	RESERVED
80	(50)	CHARACTER	4	*	RESERVED

CPOCSUC - Occurrence successor segment

Current plan occurrence successor.

Table 187. CPOCSUC Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	48	CPOCSUC	OPERATION SUCCESSOR
0	(0)	CHARACTER	16	CPOCSUCADI	APPLICATION ID
16	(10)	CHARACTER	10	CPOCSUCIA	INPUT ARRIVAL
16	(10)	CHARACTER	6	CPOCSUCIAD	MODIFIED IF IA IS MODIFIED
22	(16)	CHARACTER	4	CPOCSUCIAT	ELSE ORIGINAL FROM PLAN
26	(1A)	SIGNED	4	CPOCSUCNO	OPERATION NUMBER
30	(1E)	CHARACTER	1	CPOCSUCCR	ON CRITICAL PATH (Y|N)
31	(1F)	UNSIGNED	1	CPOCSUCVERS	VERSION NUMBER=1
32	(20)	CHARACTER	8	CPOCSUCJN	SUCCESSOR JOB NAME
40	(28)	CHARACTER	1	CPOCSUCST	SUCCESSOR STATUS
41	(29)	CHARACTER	7	*	RESERVED

Current plan operation (resource codes CPOP, CPOPCOM)

The current plan operation record can contain these segments:

CPCPR

Conditional predecessor segment.

CPCSU

Conditional successor segment.

CPEXT

Operation extended name segment.

CPLAT

Operation user-defined late info segment.

CPOP

Common segment. Only one CPOP, but it must be provided.

CPPRE

Predecessor segment.

CPREND

Distributed remote job info segment.

CPRENZ

z/OS® remote job info segment.

CPSAI

Operation system automation information segment.

CPSUC

Successor segment.

CPSR

Special resource segment.

CPREC

Operation recovery segment.

CPCPR - Conditional predecessor segment

Current plan operation conditional predecessor.

Example

```
Offsets      Type     Length  Name          Description
    0    (0) STRUCTURE    60  CPCPRE        OPERATION CONDITIONAL PREDECESSOR
    0    (0) CHARACTER    16  CPCPREADI     APPLICATION ID
   16   (10) CHARACTER    10  CPCPREIA      INPUT ARRIVAL,
   16   (10) CHARACTER     6  CPCPREIAD     MODIFIED IF IA IS MODIFIED
   22   (16) CHARACTER     4  CPCPREIAT     ELSE ORIGINAL FROM PLAN
   26   (1A) SIGNED        4  CPCPRENO      OPERATION NUMBER
   30   (1E) SIGNED        4  CPCPRE_CID    CONDITION ID
   34   (22) CHARACTER     1  CPCPRECO      PREDECESSOR COMPLETED (Y!N)
   35   (23) CHARACTER     1  CPCPRENR      --PRED. WS WAS NON-REPORTING
   36   (24) SIGNED        4  CPCPRETT      --TRANSPORT TIME (MIN)
   40   (28) CHARACTER     1  CPCPREND      PENDING PRED. OCCURRENCE
   41   (29) UNSIGNED      1  CPCPREVERS    VERSION NUMBER
   42   (2A) CHARACTER     8  CPCPREJN      JOB NAME
   50   (32) CHARACTER     1  CPCPREST      PREDECESSOR STATUS
```

```
   51   (33) CHARACTER     1  CPCPMATC       PREDECESSOR RESOLUTION CRITERIA:
                                              BLANK (MANUALLY CHOSEN)
                                              C (CLOSEST PRECEDING)
                                              S (SAME DAY)
                                              A (ABSOLUTE INTERVAL)
                                              R (RELATIVE INTERVAL)
   52   (34) SIGNED        4  CPCPRECPATH    --CRITICAL PREDECESSOR
   56   (38) CHARACTER     4  *              FREE
```

CPCSU - Conditional successor segment

Current plan operation conditional successor.

Example

```
Offsets      Type     Length  Name           Description
    0    (0) STRUCTURE    52  CPCSUC         OPERATION CONDITIONAL SUCCESSOR
    0    (0) CHARACTER    16  CPCSUCADI      APPLICATION ID
   16   (10) CHARACTER    10  CPCSUCIA       INPUT ARRIVAL,
   16   (10) CHARACTER     6  CPCSUCIAD      MODIFIED IF IA IS MODIFIED
   22   (16) CHARACTER     4  CPCSUCIAT      ELSE ORIGINAL FROM PLAN
   26   (1A) SIGNED        4  CPCSUCNO       OPERATION NUMBER
   30   (1E) SIGNED        4  CPCSUC_CID     CONDITION ID
   34   (22) CHARACTER     1  CPCSUCCR       -- ON CRITICAL PATH
   35   (23) UNSIGNED      1  CPCSUCVERS     VERSION
   36   (24) CHARACTER     8  CPCSUCJN       JOB NAME
   44   (2C) CHARACTER     1  CPCSUCST       SUCCESSOR STATUS
   45   (2D) CHARACTER     7  *
```

CPEXT - Operation extended name segment

Operation extended name.

Table 188. CPEXT Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	100	CPEXT	EXTENDED INFO OF CP OPERATION
0	(0)	CHARACTER	54	CPEXTNAME	EXTENDED NAME
54	(36)	UNSIGNED	1	CPEXTVERS	RECORD VERSION NUMBER
55	(37)	CHARACTER	1	*	RESERVED
56	(38)	SIGNED	4	CPEXTOWNOP	OWNING OP NUMBER
60	(3C)	CHARACTER	16	CPEXTSENAME	SCHEDULING ENVIRONMENT NAME
76	(4C)	CHARACTER	24	*	RESERVED

CPLAT - Operation user-defined late info segment

User-defined late info segment.

Table 189. CPLAT Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	45	CPLAT	User-defined late info
0	(0)	UNSIGNED	1	CPLATVERS	Version
1	(1)	CHARACTER	16	CPLATALE	Not started alert
1	(1)	CHARACTER	1	CPLATALEBASE	Base date (always 'F')
2	(2)	CHARACTER	1	CPLATALEDIR	Direction (always 'A')
3	(3)	CHARACTER	1	CPLATALEACT	Action (not used)
4	(4)	CHARACTER	6	CPLATALEDATE	Date for not started alert
10	(A)	CHARACTER	4	CPLATALETIME	Time for not started alert
14	(E)	CHARACTER	3	*	Free
17	(11)	CHARACTER	16	CPLATACT	Not started action
17	(11)	CHARACTER	1	CPLATACTBASE	Base date (always 'F')
18	(12)	CHARACTER	1	CPLATACTDIR	Direction (always 'A')
19	(13)	CHARACTER	1	CPLATACTACT	Not Started Action: A = Only an alert message is issued. C = The operation is set to Complete, if its status allows it. Otherwise it is NOPed. E = The operation is set to Error with OLAT, if its status allows it. Otherwise, this setting is postponed at the time when the status allows it. N = The operation and all its internal successors are NOPed, if their status allows NOPing. Otherwise, it is ignored.
20	(14)	CHARACTER	6	CPLATACTDATE	Date for not started action
26	(1A)	CHARACTER	4	CPLATACTTIME	Time for not started action
30	(1E)	CHARACTER	3	*	Free
33	(21)	CHARACTER	12	*	Free

CPOP - Common segment

Current plan operation.

Note: Operation status codes:

A

Waiting for input to arrive

C

Completed

E

Ended in error

I

Interrupted

R

Ready

S

Started

U

Undecided

W

Waiting

Y

Completed by NOERROR processing

*

Ready with at least one predecessor completed on a nonreporting workstation

- Minutes are the unit of duration.
- Y and N are the indicator values.
- SMF reader date formats are 00YYDDDF for the 20th century, and 01YYDDDF for the 21st century.

Table 190. CPOP Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	440	CPOPCOM	CURRENT PLAN OPERATION

Table 190. CPOP Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	CHARACTER	16	CPOPADI	APPLICATION ID
16	(10)	CHARACTER	10	CPOPIA	APPLICATION INPUT ARRIVAL
16	(10)	CHARACTER	6	CPOPIAD	MODIFIED, IF IA IS MODIFIED
22	(16)	CHARACTER	4	CPOPIAT	ELSE ORIGINAL FROM PLAN
26	(1A)	SIGNED	4	CPOPNO	OPERATION NUMBER
30	(1E)	CHARACTER	8	CPOPGRP	AUTHORITY GROUP
38	(26)	CHARACTER	24	CPOPDESC	DESCRIPTIVE TEXT
62	(3E)	CHARACTER	8	CPOPJBN	OP OS JOBNAME | BLANK
70	(46)	CHARACTER	8	CPOPJES	JOB ID
78	(4E)	CHARACTER	4	CPOPWSN	WORKSTATION NAME
82	(52)	CHARACTER	8	CPOPFRM	FORM NUMBER | BLANK
90	(5A)	CHARACTER	10	CPOPPS	PLANNED START
90	(5A)	CHARACTER	6	CPOPPSD	DATE | BLANK
96	(60)	CHARACTER	4	CPOPPST	TIME | BLANK
100	(64)	CHARACTER	10	CPOPPE	PLANNED END
100	(64)	CHARACTER	6	CPOPPED	DATE | BLANK
106	(6A)	CHARACTER	4	CPOPPET	TIME | BLANK
110	(6E)	CHARACTER	10	CPOPOI	OPERATION INPUT ARRIVAL
110	(6E)	CHARACTER	6	CPOPOID	DATE | BLANK
116	(74)	CHARACTER	4	CPOPOIT	TIME | BLANK
120	(78)	CHARACTER	10	CPOPOD	OPERATION DEADLINE
120	(78)	CHARACTER	6	CPOPODD	DATE | BLANK
126	(7E)	CHARACTER	4	CPOPODT	TIME | BLANK
130	(82)	CHARACTER	10	CPOPLO	LATEST OUT FOR OP
130	(82)	CHARACTER	6	CPOPLOD	DATE
136	(88)	CHARACTER	4	CPOPLOT	TIME
140	(8C)	CHARACTER	10	CPOPAS	ACTUAL START

Table 190. CPOP Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
140	(8C)	CHARACTER	6	CPOPASD	DATE | BLANK
146	(92)	CHARACTER	4	CPOPAST	TIME | BLANK
150	(96)	CHARACTER	10	CPOPAA	ACTUAL ARRIVAL
150	(96)	CHARACTER	6	CPOPAAD	DATE | BLANK
156	(9C)	CHARACTER	4	CPOPAAT	TIME | BLANK
160	(A0)	CHARACTER	10	CPOPIS	INTERMED.START, IF INTERRUPTED
160	(A0)	CHARACTER	6	CPOPISD	DATE | BLANK
166	(A6)	CHARACTER	4	CPOPIST	TIME | BLANK
170	(AA)	CHARACTER	10	CPOPAE	ACTUAL END
170	(AA)	CHARACTER	6	CPOPAED	DATE | BLANK
176	(B0)	CHARACTER	4	CPOPAET	TIME | BLANK
180	(B4)	CHARACTER	4	CPOPEDU	ESTIMATED DURATION
180	(B4)	CHARACTER	2	CPOPEDH	ESTIMATED DURATION HOURS HH
182	(B6)	CHARACTER	2	CPOPEDM	ESTIMATED DURATION MINS MM
184	(B8)	CHARACTER	6	CPOPADU	ACTUAL DURATION
184	(B8)	CHARACTER	4	CPOPADH	EST. DURATION HRS HHHH | BLANK
188	(BC)	CHARACTER	2	CPOPADM	EST. DURATION MINS MM | BLANK
190	(BE)	CHARACTER	1	CPOPST	CURRENT STATUS
191	(BF)	CHARACTER	4	CPOPERR	ERROR CODE
195	(C3)	CHARACTER	1	CPOPXST	EXTENDED STATUS
196	(C4)	SIGNED	4	CPOP#PS	NUMBER OF PARALLEL SERVERS REQUIRED
200	(C8)	SIGNED	4	CPOP#R1	WS RESOURCES REQUIRED
204	(CC)	SIGNED	4	CPOP#R2	WS RESOURCES REQUIRED
208	(D0)	SIGNED	4	CPOPPRI	PRIORITY
212	(D4)	SIGNED	4	CPOP#SU	NUMBER OF SUCCESSORS
216	(D8)	SIGNED	4	CPOP#PR	NUMBER OF PREDECESSORS
220	(DC)	SIGNED	4	CPOP#PC	NUMBER OF COMPLETED PREDECESSORS

Table 190. CPOP Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
224	(E0)	SIGNED	4	CPOP#SR	NUMBER OF SPECIAL RESOURCES
228	(E4)	SIGNED	4	CPOPPTT	TRANSPORT TIME IF PRED, MIN
232	(E8)	SIGNED	4	CPOPRDD	SMF READER DATE (00YYDDDF or 01YYDDDF)
236	(EC)	SIGNED	4	CPOPRDT	SMF READER TIME (1/100 SEC)
240	(F0)	CHARACTER	1	CPOPJCL	JOB CLASS, SYSOUT CLASS | BLANK
241	(F1)	CHARACTER	1	CPOPAEC	AUTO ERROR COMPLETION (Y|N)
242	(F2)	CHARACTER	1	CPOPASUB	AUTO JOB SUBMISSION(Y|N)
243	(F3)	CHARACTER	1	CPOPAJR	AUTO HOLD/RELEASE (Y|N)
244	(F4)	CHARACTER	1	CPOPTIME	TIME JOB (Y|N)
245	(F5)	CHARACTER	1	CPOPCLATE	CANCEL IF LATE (Y|N|)
246	(F6)	CHARACTER	8	CPOPMCPUP	TIME OF THE LAST MCP UPDATE. FOR THE 20TH CENTURY, THE FORMAT IS 00YYDDDF HHMM. FOR THE 21TH CENTURY THE FORMAT IS 01YYDDDF HHMM. IF NO MCP UPDATE WAS PERFORMED ON A CPOP RECORD, THIS FIELD CONTAINS BINARY ZEROES.
254	(FE)	CHARACTER	1	CPOPCPTH	ON CRITICAL PATH (F|Y|N)
255	(FF)	CHARACTER	1	CPOPLATE	LATEST OUT PASSED (Y|N)
256	(100)	CHARACTER	1	CPOPURG	URGENT (Y|)
257	(101)	CHARACTER	1	CPOPJST	JOB STATUS (H|Q| |N)
258	(102)	CHARACTER	1	CPOPPREP	JCL PREPARATION OP. (Y|N)
259	(103)	CHARACTER	1	CPOPOIST	OP INSTR EXIST (Y|N|+)
260	(104)	SIGNED	4	CPOPHRC	HIGHEST OK RETURN CODE
264	(108)	UNSIGNED	1	CPOPVERS	VERSION NUMBER=2
265	(109)	CHARACTER	1	CPOPPWTO	DEADLINE WTO Y|N
266	(10A)	CHARACTER	1	CPOPRES	RESTARTABLE Y|N|<BLANK>
267	(10B)	CHARACTER	1	CPOPRER	REROUTABLE Y|N|<BLANK>

Table 190. CPOP Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
268	(10C)	CHARACTER	1	CPOPHRCS	HIGHEST RC SET Y|N|<BLANK>
269	(10D)	CHARACTER	1	CPOPMHLD	MANUALLY HELD OP Y|N|<BLANK>
270	(10E)	CHARACTER	1	CPOPNOP	NOPED OPERATION Y|N|<BLANK>
271	(10F)	CHARACTER	1	CPOPCATM	RESTART AND CLEANUP A=AUTOM., I=IMMED., M=MANUAL, N=NONE
272	(110)	CHARACTER	16	CPOPUDA	USER DATA
288	(120)	CHARACTER	4	CPOPCMDS	OPERATION COMMANDS
288	(120)	CHARACTER	2	CPOPCMD	OPERATION COMMAND
290	(122)	CHARACTER	2	*	RESERVED
292	(124)	CHARACTER	1	CPOPCSTA	CLEANUP STATUS
293	(125)	CHARACTER	8	CPOPWSINFO	WORKSTATION INFORMATION
293	(125)	CHARACTER	1	CPOPWSISET	INFO AVAILABLE Y|N
294	(126)	CHARACTER	1	CPOPWSTYPE	TYPE G|C|P
295	(127)	CHARACTER	1	CPOPWSREP	REPORTING ATTRIBUTE A|S|C|N
296	(128)	CHARACTER	1	CPOPWSSUBT	SUBTYPE JCL, STC, WTO, NONE J|S|W|T|A BLANK
297	(129)	CHARACTER	1	CPOPWSSTAT	STATUS A|F|O|U|<BLANK>
298	(12A)	CHARACTER	1	CPOPWSRRM	REROUTE MODE Y|N
299	(12B)	CHARACTER	2	*	RESERVED
301	(12D)	CHARACTER	1	CPOPJCRT	WORKLOAD MONITOR CRITICAL JOB
302	(12E)	CHARACTER	1	CPOPJPOL	WORKLOAD MONITOR LATE JOB POLICY
303	(12F)	CHARACTER	1	CPOPDPREM	REMOVABLE BY DP
304	(130)	SIGNED	4	CPOPEDUI	ESTIMATED DUR. IN 100th OF SEC.
308	(134)	UNSIGNED	4	CPOPADUI	ACTUAL DUR.IN 100th OF SEC.
312	(138)	SIGNED	4	CPOPPSTI	PLAN. START TIME IN 100th OF SEC.
316	(13C)	SIGNED	4	CPOPPETI	PLAN. END TIME IN 100th OF SEC.

Table 190. CPOP Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
320	(140)	SIGNED	4	CPOPLOTI	LATEST OUT TIME IN 100th OF SEC.
324	(144)	SIGNED	4	CPOPASTI	ACTUAL START TIME IN 100th OF SEC.
328	(148)	SIGNED	4	CPOPAATI	ACTUAL ARR. TIME IN 100th OF SEC.
332	(14C)	SIGNED	4	CPOPISTI	INT. START TIME IN 100th OF SEC.
336	(150)	SIGNED	4	CPOPAETI	ACTUAL END TIME IN 100th OF SEC.
340	(154)	CHARACTER	1	CPOPEXPJCL	EXPANDED JCL NEEDED
341	(155)	CHARACTER	1	CPOPUSRSYS	USER SYSOUT NEEDED
342	(156)	CHARACTER	8	CPOPOCTO	OCCURRENCE TOKEN
350	(15E)	CHARACTER	1	CPOPMON	MONITORING FLAG
351	(15F)	CHARACTER	1	*	RESERVED
352	(160)	SIGNED	4	CPOPNLVL	MAX NESTING LEVEL
356	(164)	CHARACTER	1	CPOPRECIS	Y IF CPREC SEGMENT EXISTS
357	(165)	CHARACTER	1	CPOPTWSJNM	NOT USED
358	(166)	CHARACTER	1	CPOPINSYM	NOT USED
359	(167)	CHARACTER	1	CPOPDELAY	STARTED ON WAIT WORKSTATION (Y|N)
360	(168)	SIGNED	4	CPOPCRITPATH	BELONGING TO CRITICAL PATH
364	(16C)	CHARACTER	8	CPOPWLMCLASS	WLM SERVICE CLASS
372	(174)	CHARACTER	1	CPOPWAITSE	WAITING FOR SCHEDULING ENVIRONMENT (N|S|Y)
373	(175)	CHARACTER	8	CPOPVIRTDEST	SUBMISSION DESTINATION
381	(17D)	CHARACTER	8	CPOPEXECDEST	EXECUTION DESTINATION
389	(185)	CHARACTER	1	CPOPCONDRJOB	CONDITIONAL RECOVERY JOB
390	(186)	CHARACTER	1	CPOPUNEXPRC	UNEXPECTED RC (Y|N)
391	(187)	CHARACTER	1	CPOPSHADOW	SHADOW JOB (Y|N)
392	(188)	SIGNED	4	CPOPFTRC	
396	(18C)	SIGNED	4	CPOP#CPROP	NUMBER OF CONDITIONAL PREDECESSORS
400	(190)	SIGNED	4	CPOP#CSUOP	NUMBER OF CONDITIONAL SUCCESSORS

Table 190. CPOP Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
404	(194)	SIGNED	4	CPOP#CONDTOT	NUMBER OF CONDITIONS
408	(198)	SIGNED	4	CPOP#COND_T	NUMBER OF TRUE CONDITIONS
412	(19C)	SIGNED	4	CPOP#COND_F	NUMBER OF FALSE CONDITIONS
416	(1A0)	SIGNED	4	CPOP#PX	NUMBER OF PREDECESSORS IN X STATUS
420	(1A4)	CHARACTER	4	CPOPORIGRC	ORIGINAL RETURN CODE
424	(1A8)	CHARACTER	1	CPOPBNDST	BIND STATUS FOR SHADOW JOBS. POSSIBLE VALUES FOR CPOPBNDST ARE: • **P**: BIND SENT • **J**: SENDING BIND • **B**: BIND ERROR • **I**: BIND OK
425	(1A9)	CHARACTER	1	CPOPWPEND	WAITING PENDING PREDECESSORS (Y|N)
426	(1AA)	CHARACTER	1	CPOPWMPEND	WAITING MANDATORY PENDING PREDECESSORS (Y|N)
427	(1AB)	CHARACTER	1	CPOPWMPPEND	WAITING MANDATORY OR PENDING PREDECESSORS (Y|N)
428	(1AC)	CHARACTER	1	CPOPLTEL	OPERATION LATE ON LATEST OUT
429	(1AD)	CHARACTER	1	CPOPLTEN	OPERATION LATE ON ALERT/ACTION
430	(1AE)	CHARACTER	1	CPOPLTEE	OPERATION LATE ON LATEST OUT OR LATE ON ALERT/ACTION
431	(1AF)	CHARACTER	1	*	FREE
432	(1B0)	CHARACTER	8	CPOPRUNC	RUN CYCLE ASSOCIATED WITH THE DURATION AND DEADLINE
440	(1B8)	CHARACTER	8	CPOPTOD	NOT USED
448	(1C0)	CHARACTER	6	CPOPORIGDLD	ORIGINAL DEADLINE DATE
454	(1C6)	CHARACTER	4	CPOPORIGDLT	ORIGINAL DEADLINE TIME
458	(1CA)	CHARACTER	1	CPOPORIGDLA	DEADLINE ACTION

Table 190. CPOP Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
					' ' (blank) = Only an alert message is issued. A = Only an alert message is issued. C = The operation is set to Complete, if its status allows it. Otherwise it is NOPed. E = The operation is set to Error with OLAT, if its status allows it. Otherwise, this setting is postponed at the time when the status allows it. N = The operation and all its internal successors are NOPed, if their status allows NOPing. Otherwise, it is ignored.
459	(1CB)	CHARACTER	1	CPOPULATE	USER-DEFINED LATE
460	(1CC)	CHARACTER	1	CPOPMOVEDL	DEADLINE MOVED TO TAIL END
461		CHARACTER	11	*	FREE

CPOPSRU - Special resource usage segment

Current plan operation special resource use.

When retrieving information about the operations waiting for a certain resource (LIST CPOPSRU with argument LISTTYPE=WAITQ) or those having a certain resource allocated (LIST CPOPSRU with argument LISTTYPE=INUSE) the information about each operation is shown in the segment.

Table 191. CPOPSRU Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	96	CPOPSRU	CP OPERATION, SR USAGE
0	(0)	CHARACTER	16	CPOPUADI	APPLICATION ID
16	(10)	CHARACTER	10	CPOPUIA	APPLICATION INPUT ARRIVAL
16	(10)	CHARACTER	6	CPOPUIAD	MODIFIED IF IA IS MODIFIED
22	(16)	CHARACTER	4	CPOPUIAT	ELSE ORIGINAL FROM PLAN
26	(1A)	SIGNED	4	CPOPUNO	OPERATION NUMBER

Table 191. CPOPSRU Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
30	(1E)	CHARACTER	8	CPOPUJBN	OP OS JOBNAME | BLANK
38	(26)	CHARACTER	4	CPOPUWSN	WS NAME
42	(2A)	CHARACTER	10	CPOPULO	LATEST OUT
42	(2A)	CHARACTER	6	CPOPULOD	DATE, BLANK IF IN-USE LIST
48	(30)	CHARACTER	4	CPOPULOT	TIME, BLANK IF IN-USE LIST
52	(34)	CHARACTER	10	CPOPUAS	ACTUAL START
52	(34)	CHARACTER	6	CPOPUASD	DATE, BLANK IF WAIT QUEUE
58	(3A)	CHARACTER	4	CPOPUAST	TIME, BLANK IF WAIT QUEUE
62	(3E)	CHARACTER	4	CPOPUEDU	ESTIMATED DURATION
62	(3E)	CHARACTER	2	CPOPUEDH	EST DUR HH
64	(40)	CHARACTER	2	CPOPUEDM	EST DUR MM
66	(42)	CHARACTER	1	CPOPUST	CURRENT STATE
67	(43)	UNSIGNED	1	CPOPUVERS	VERSION
68	(44)	SIGNED	4	CPOPUPRI	PRIORITY
72	(48)	SIGNED	4	CPOPUSRQ	SR QUANTITY USED/NEEDED
76	(4C)	CHARACTER	8	CPOPUWRS	REASON FOR WAIT FOR SR
84	(54)	CHARACTER	1	CPOPUSRU	SR ALLOCATION TYPE
85	(55)	CHARACTER	3	*	RESERVED
88	(58)	SIGNED	4	CPOPUEDUI	ESTIMATED DUR. IN 100th OF SEC.
92	(5C)	CHARACTER	4	*	RESERVED

CPPRE - Predecessor segment

Current plan operation predecessor.

Note: Y and N are the indicator values.

Table 192. CPPRE Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	60	CPPRE	OPERATION PREDECESSOR
0	(0)	CHARACTER	16	CPPREADI	APPLICATION ID
16	(10)	CHARACTER	10	CPPREIA	INPUT ARRIVAL
16	(10)	CHARACTER	6	CPPREIAD	MODIFIED IF IA IS MODIFIED
22	(16)	CHARACTER	4	CPPREIAT	ELSE ORIGINAL FROM PLAN
26	(1A)	SIGNED	4	CPPRENO	OPERATION NUMBER
30	(1E)	CHARACTER	1	CPPRECO	PREDECESSOR COMPLETED (Y|N)
31	(1F)	CHARACTER	1	CPPRENR	PRED. WS WAS NONREPORTING
32	(20)	SIGNED	4	CPPRETT	TRANSPORT TIME
36	(24)	CHARACTER	1	CPPREND	PENDING PRED OCCURRENCE
37	(25)	UNSIGNED	1	CPPREVERS	VERSION NUMBER=1
38	(26)	CHARACTER	8	CPPREJN	PREDECESSOR JOB NAME
46	(2E)	CHARACTER	1	CPPREST	PREDECESSOR STATUS
47	(2F)	CHARACTER	1	CPPMATC	PREDECESSOR RESOLUTION CRITERIA: BLANK (MANUALLY CHOSEN) C (CLOSEST PRECEDING) S (SAME DAY) A (ABSOLUTE INTERVAL) R (RELATIVE INTERVAL)
48	(30)	SIGNED	4	CPPRECRITPATH	PREDECESSOR OF AN OPERATION BELONGING TO A CRITICAL PATH
52	(34)	CHARACTER	1	CPPMANDP	Y: MANDATORY PENDING (CANNOT BE SET)
53	(35)	CHARACTER	10	CPPREFRIA	MANDATORY PENDING INTERVAL START DATE IN THE YYDDMMHHMM FORMAT
63	(3F)	CHARACTER	10	CPPRETOIA	MANDATORY PENDING INTERVAL END DATE IN THE YYDDMMHHMM FORMAT
73	(49)	CHARACTER	7	*	RESERVED

Table 192. CPPRE Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
80	(50)	CHARACTER	4	*	RESERVED

CPREND - Distributed remote job info segment

Distributed remote job info segment.

Note: Y and N are the indicator values.

Table 193. CPREND Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	108	CPREND	OPERATION DISTRIBUTED REMOTE JOB INFO
0	(0)	UNSIGNED	1	CPRDVERS	RECORD VERSION NUMBER
1	(1)	CHARACTER	1	CPRDCOMP	COMPLETE ON FAILED BIND (EDIT) Y|N
2	(2)	CHARACTER	2	*	FREE
4	(4)	SIGNED	4	CPRDOWOP	OWNING OP NUMBER
8	(8)	CHARACTER	16	CPRDJSN	JOB STREAM NAME (EDIT)
24	(18)	CHARACTER	16	CPRDJSWS	JOB STREAM WORKSTATION (EDIT)
40	(28)	CHARACTER	40	CPRJOBN	JOB NAME (EDIT)
80	(50)	CHARACTER	10	CPRDIA	INPUT ARRIVAL (BROWSE)
80	(50)	CHARACTER	6	CPRDIAD	DATE | BLANK
86	(56)	CHARACTER	4	CPRDIAT	TIME | BLANK
90	(5A)	CHARACTER	18	*	

CPRENZ - z/OS remote job info segment

z/OS remote job info segment.

Note: Y and N are the indicator values.

Table 194. CPRENZ Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	72	CPRENZ	OPERATION z/OS REMOTE JOB INFO
0	(0)	UNSIGNED	1	CPRZVERS	RECORD VERSION NUMBER
1	(1)	CHARACTER	1	CPRZCOMP	COMPLETE ON FAILED BIND (EDIT) Y|N
2	(2)	CHARACTER	2	*	FREE
4	(4)	SIGNED	4	CPRZOWOP	OWNING OP NUMBER
8	(8)	SIGNED	4	CPRZOPNO	OP NUMBER (EDIT)
12	(C)	CHARACTER	16	CPRZOCCN	APPLICATION ID (EDIT)
18	(1C)	CHARACTER	4	CPRZWS	JOB WORKSTATION (BROWSE)
32	(20)	CHARACTER	8	CPRZJOBN	JOB NAME (BROWSE)
40	(28)	STRUCTURE	10	CPRZIA	INPUT ARRIVAL (BROWSE)
40	(28)	CHARACTER	6	CPRZIAD	DATE | BLANK
46	(2E)	CHARACTER	4	CPRZIAT	TIME | BLANK
50	(32)	CHARACTER	1	*	FREE
51	(33)	CHARACTER	1	CPRZMATCHTYPE	MATCHING CRITERIA: C|S|R|A
52	(34)	CHARACTER	1	*	FREE
53	(35)	CHARACTER	1	CPRZFWHE	FROM WHEN: B=BEFORE IA | A=AFTER IA
54	(36)	CHARACTER	3	CPRZFHHH	FROM HOURS HHH (ONLY REL)
57	(39)	CHARACTER	2	CPRZFHH	FROM HOURS HH (ONLY ABS)
59	(3B)	CHARACTER	2	CPRZFMM	FROM MINUTES MM
61	(3D)	CHARACTER	1	CPRZFD	FROM DAYS (ONLY ABS)
62	(3E)	CHARACTER	1	CPRZTWHE	TO WHEN: B=BEFORE IA | A=AFTER IA
63	(3F)	CHARACTER	3	CPRZTHHH	TO HOURS HHH (ONLY REL)
66	(42)	CHARACTER	2	CPRZTHH	TO HOURS HH (ONLY ABS)
68	(44)	CHARACTER	2	CPRZTMM	TO MINUTES MM
70	(46)	CHARACTER	1	CPRZTD	TO DAYS (ONLY ABS)

Table 194. CPRENZ Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
71	(47)	CHARACTER	1	*	FREE

CPSAI - Operation system automation information segment

System automation information.

Note: This segment exists for system automation operations only.

Table 195. CPSAI Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	352	CPSAI	SYSTEM AUTOMATION INFO FOR CURRENT PLAN OPERATION
0	(0)	CHARACTER	256	CPSAICOMMTEXT	SYSTEM AUTOMATION OPERATION COMMAND TEXT
0	(0)	CHARACTER	64	CPSAICOMMTEX1	SYSTEM AUTOMATION OPERATION COMMAND TEXT ROW1
64	(40)	CHARACTER	64	CPSAICOMMTEX2	SYSTEM AUTOMATION OPERATION COMMAND TEXT ROW2
128	(80)	CHARACTER	64	CPSAICOMMTEX3	SYSTEM AUTOMATION OPERATION COMMAND TEXT ROW3
192	(C0)	CHARACTER	63	CPSAICOMMTEX4	SYSTEM AUTOMATION OPERATION COMMAND TEXT ROW4
255	(FF)	CHARACTER	1	CPSAIFILLER	RESERVED
256	(100)	CHARACTER	8	CPSAIAUTOOPER	SYSTEM AUTOMATION AUTOMATED FUNCTION (FOR OPERATION)
264	(108)	CHARACTER	8	CPSAISECELEM	SYSTEM AUTOMATION SECURITY ELEMENT
272	(110)	CHARACTER	64	CPSAICOMPINFO	SYSTEM AUTOMATION COMPLETION INFORMATION
336	(150)	CHARACTER	4	*	RESERVED
340	(154)	SIGNED	4	CPSAIOWNOP	OWNING OPERATION NUMBER

Table 195. CPSAI Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
344	(158)	CHARACTER	8	*	RESERVED

CPSUC - Successor segment

Current plan operation successor.

Note: Y and N are the indicator values.

Table 196. CPSUC Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	48	CPSUC	OPERATION SUCCESSOR
0	(0)	CHARACTER	16	CPSUCADI	APPLICATION ID
16	(10)	CHARACTER	10	CPSUCIA	INPUT ARRIVAL
16	(10)	CHARACTER	6	CPSUCIAD	MODIFIED IF IA IS MODIFIED
22	(16)	CHARACTER	4	CPSUCIAT	ELSE ORIGINAL FROM PLAN
26	(1A)	SIGNED	4	CPSUCNO	OPERATION NUMBER
30	(1E)	CHARACTER	1	CPSUCCR	ON CRITICAL PATH (Y|N)
31	(1F)	UNSIGNED	1	CPSUCVERS	VERSION NUMBER=1
32	(20)	CHARACTER	8	CPSUCJN	SUCCESSOR JOB NAME
40	(28)	CHARACTER	1	CPSUCST	SUCCESSOR STATUS
41	(29)	CHARACTER	7	*	RESERVED

CPSR - Special resource segment

Current plan operation special resource use.

Table 197. CPSR Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	66	CPSR	

Table 197. CPSR Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	CHARACTER	44	CPSRN	NAME
44	(2C)	CHARACTER	1	CPSRU	USAGE (S=SHARED, X=EXCLUSIVE)
45	(2D)	UNSIGNED	1	CPSRVERS	VERSION
46	(2E)	CHARACTER	1	CPSRONER	ON ERROR FLAG
47	(2F)	CHARACTER	1	*	FREE
48	(30)	SIGNED	4	CPSRAMNT	QUANTITY
52	(34)	CHARACTER	1	CPSRAVACO	ON COMPLETE (Y|N|R|blank)
53	(35)	CHARACTER	13	*	RESERVED

Note: For CPSRAMNT, the value 0 means the total quantity of the special resource.

CPREC - Operation recovery segment

Table 198. CPREC Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	158	CPREC	OPERATION RECOVERY
0	(0)	CHARACTER	16	CPRECAID	APPLICATION ID
16	(10)	CHARACTER	10	CPRECIA	INPUT ARRIVAL
16	(10)	CHARACTER	6	CPRECIAD	MODIFIED IF IA IS MODIFIED
22	(16)	CHARACTER	4	CPRECIAT	ELSE ORIGINAL FROM PLAN
26	(1A)	SIGNED	4	CPRECNO	OPERATION NUMBER
30	(1E)	CHARACTER	8	CPRECJREID	ID OF RECOVERY JOB
38	(26)	CHARACTER	4	CPRECWSN	WORK STATION NAME OF REC JOB
42	(2A)	SIGNED	10	CPRECS	RECOVERY JOB START
42	(2A)	CHARACTER	6	CPRECSD	DATE | BLANK
48	(30)	SIGNED	4	CPRECST	TIME SEC*100 | 0
52	(34)	CHARACTER	10	CPRECE	RECOVERY JOB END

Table 198. CPREC Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
52	(34)	CHARACTER	6	CPRECED	DATE | BLANK
58	(3A)	SIGNED	4	CPRECET	TIME SEC*100 | 0
62	(3E)	CHARACTER	1	CPRECRJST	RECOVERY JOB STATUS
63	(3F)	CHARACTER	1	CPRECTYPE	RECOVERY TYPE: S - STOP C - CONTINUE R - RERUN
64	(40)	SIGNED	4	CPRECDUR	RECOVERY JOB DURATION
68	(44)	SIGNED	4	CPRECPROMPTID	RECOVERY PROMPT ID
72	(48)	CHARACTER	64	CPRECPRTMSG	RECOVERY MESSAGE
136	(88)	CHARACTER	1	CPRECPRTSTAT	RECOVERY PROMPT STATUS ' ' - NO REPLY 'N' - REPLY WITH N 'Y' - REPLY WITH Y
137	(89)	CHARACTER	8	CPRECJID	ID OF JOB TO RECOVER
145	(91)	CHARACTER	4	CPRECERC	RECOVERY JOB ERROR CODE
149	(95)	UNSIGNED	1	CPRECVERS	VERSION NUMBER
150	(96)	CHARACTER	8	*	RESERVED

Current plan status (resource code CPST)

The current plan status record consists of one segment:

CPST

Common segment. One must always exist.

CPST - Common segment

Current plan status.

The CPSTTURN can have one of these values:

W

A daily plan batch job that creates a new plan sets this value when it runs.

H

A daily plan batch job that has successfully created a new plan sets this value when the plan (the NCP data set) is finished.

N

IBM Z Workload Scheduler sets this value when it is started and no turnover is in progress.

' '

IBM Z Workload Scheduler sets this value (a blank) after a successful turnover if a daily plan batch that set the value W ends without setting the value H.

Table 199. CPST Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	96	CPST	CURRENT PLAN STATUS
0	(0)	UNSIGNED	1	CPSTVERS	VERSION NUMBER=1
1	(1)	CHARACTER	6	CPSTCRD	CURRENT PLAN CREATE DATE
7	(7)	CHARACTER	4	CPSTCRT	CURRENT PLAN CREATE TIME
11	(B)	CHARACTER	6	CPSTENDD	CURRENT PLAN END DATE
17	(11)	CHARACTER	4	CPSTENDT	CURRENT PLAN END TIME
21	(15)	CHARACTER	6	CPSTBUD	LAST BACKUP DATE
27	(1B)	CHARACTER	4	CPSTBUT	LAST BACKUP TIME
31	(1F)	CHARACTER	6	CPST1ED	1ST EVENT AFTER BACKUP DATE
37	(25)	CHARACTER	4	CPST1ET	1ST EVENT AFTER BACKUP TIME
41	(29)	CHARACTER	8	CPST1EDTS	TIMESTAMP DATE FROM 1ST EVENT
49	(31)	CHARACTER	8	CPST1ETTS	TIMESTAMP TIME FROM 1ST EVENT
57	(39)	CHARACTER	1	CPSTTURN	TURNOVER PRODUCES NCP
58	(3A)	CHARACTER	1	CPSTCP	CURRENT PLAN EXIST (Y|N)
59	(3B)	CHARACTER	8	CPSTCPDDN	CURRENT PLAN DDNAME
67	(43)	CHARACTER	8	CPSTJTDDN	JOB TRACKING DDNAME
75	(4B)	CHARACTER	8	CPSTJSDDN	JCL REPOSITORY DDNAME
83	(53)	CHARACTER	13	*	RESERVED

Current plan operation user field (resource codes CPUSRF, CPUSRFELEM)

The current plan operation user field consists of one segment (CPUSRF).

CPUSRF - Operation user field segment

Current plan operation user field.

Table 200. CPUSRF Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	132	CPUSRF	OPERATION USER FIELD
0	(0)	CHARACTER	16	CPUFADI	APPLICATION ID
16	(10)	CHARACTER	10	CPUFIA	APPLICATION INPUT ARRIVAL
16	(10)	CHARACTER	6	CPUFIAD	MODIFIED IF IA IS MODIFIED
22	(16)	CHARACTER	4	CPUFIAT	ELSE ORIGINAL FROM PLAN
26	(1A)	SIGNED	4	CPUFOPNO	OPERATION NUMBER
30	(1E)	CHARACTER	16	CPUFNAME	USER FIELD NAME
46	(2E)	CHARACTER	54	CPUFVALUE	USER FIELD VALUE
100	(64)	SIGNED	4	CPUF#UF	NUMBER OF USER FIELDS
104	(68)	SIGNED	1	CPUFVERS	VERSION
105	(69)	CHARACTER	27	*	FREE

Table 201. CPUSRFELEM Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	70	CPUSRFELEM	ELEMENT (NAME, VALUE)
0	(0)	CHARACTER	16	CPUSRFNAME	USER FIELD NAME
16	(10)	CHARACTER	54	CPUSRFVALUE	USER FIELD VALUE

Current plan workstation (resource codes CPWS, CPWSCOM)

The current plan workstation record consists of the common (CPWS) segment. One must appear as the first segment in each record. The CPWS segment can be followed by a variable number of CPIVL segments that represent the open intervals for the workstation.

CPWS - Common segment

Current plan workstation.

Workstation types:

C

Computer workstation

P

Printer workstation

G

General workstation

R

Remote engine workstation

Reporting attribute:

A

Automatic reporting

S

Manual reporting start and stop

C

Manual reporting, completion only

N

Nonreporting

- The number of started operations is the number of parallel servers in use.
- Minutes are the unit of duration.
- Y and N are the indicator values.

Table 202. CPWS Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	132	CPWS	CURRENT PLAN WORK STATION
0	(0)	CHARACTER	4	CPWSN	WORKSTATION NAME
4	(4)	CHARACTER	32	CPWSDESC	WORKSTATION DESCRIPTION
36	(24)	CHARACTER	12	CPWSSC	COMPLETED OPERATIONS SUMMARY

Table 202. CPWS Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
36	(24)	SIGNED	4	CPWSSC#	NUMBER OF COMPLETED OPS
40	(28)	SIGNED	4	CPWSSCE	ESTIMATED DURATION
44	(2C)	SIGNED	4	CPWSSCR	ACTUAL DURATION
48	(30)	CHARACTER	12	CPWSSI	INTERRUPTED OPERATIONS SUMMARY
48	(30)	SIGNED	4	CPWSSI#	NUMBER OF INTERRUPTED OPS
52	(34)	SIGNED	4	CPWSSIE	ESTIMATED DURATION
56	(38)	SIGNED	4	CPWSSIR	ACTUAL DURATION
60	(3C)	CHARACTER	8	CPWSSS	STARTED OPERATIONS SUMMARY
60	(3C)	SIGNED	4	CPWSSS#	NUMBER OF STARTED OPERATIONS
64	(40)	SIGNED	4	CPWSSSE	ESTIMATED DURATION
68	(44)	CHARACTER	8	CPWSSR	READY OPERATIONS SUMMARY
68	(44)	SIGNED	4	CPWSSR#	NUMBER OF READY OPERATIONS
72	(48)	SIGNED	4	CPWSSRE	ESTIMATED DURATION
76	(4C)	CHARACTER	8	CPWSSW	WAITING OPERATIONS SUMMARY
76	(4C)	SIGNED	4	CPWSSW#	NUMBER OF WAITING OPERATIONS
80	(50)	SIGNED	4	CPWSSWE	ESTIMATED DURATION
84	(54)	SIGNED	4	CPWSR1IU#	NUMBER OF R1 RESOURCES IN USE
88	(58)	SIGNED	4	CPWSR2IU#	NUMBER OF R2 RESOURCES IN USE
92	(5C)	SIGNED	4	CPWSIVL#	NUMBER OF OPEN INTERVALS
96	(60)	CHARACTER	1	CPWSTYPE	WORKSTATION TYPE (G|C|P|R)
97	(61)	CHARACTER	1	CPWSREP	REPORTING ATTRIBUTE (A|S|C|N(
98	(62)	CHARACTER	1	CPWSPSC	CONTROL ON PARALLEL SERVERS
99	(63)	CHARACTER	2	CPWSR1N	R1 RESOURCE NAME
101	(65)	CHARACTER	1	CPWSR1C	R1 RESOURCE USED FOR CONTROL
102	(66)	CHARACTER	2	CPWSR2N	R2 RESOURCE NAME
104	(68)	CHARACTER	1	CPWSR2C	R2 RESOURCE USED FOR CONTROL
105	(69)	CHARACTER	1	CPWSPREP	JOB SETUP ABILITY

Table 202. CPWS Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
106	(6A)	UNSIGNED	1	CPWSVERS	VERSION NUMBER=1
107	(6B)	CHARACTER	1	CPWSSTC	STARTED TASK (Y|N)
108	(6C)	CHARACTER	1	CPWSWTO	DEADLINE WTO (Y|N)
109	(6D)	CHARACTER	1	CPWSSTAT	WORKSTATION STATUS (A|O|F)
110	(6E)	CHARACTER	1	CPWSRERUT	REROUTE MODE (Y|N)
111	(6F)	CHARACTER	4	CPWSALTWS	ALTERNATE WS NAME
115	(73)	CHARACTER	1	*	RESERVED
116	(74)	CHARACTER	1	*	RESERVED
117	(75)	CHARACTER	1	CPWSFLK	FULL LINKED (Y|N)
118	(76)	CHARACTER	1	CPWSAUTO	SYSTEM AUTOMATION WORKSTATION
119	(77)	CHARACTER	1	CPWSVIRT	VIRTUAL WORKSTATION
120	(78)	CHARACTER	8	CPWSDEST	DESTINATION
128	(80)	CHARACTER	1	CPWSWAIT	WAIT WORKSTATION (Y|N)
129	(81)	CHARACTER	1	CPWSFULLYACT	VIRTUAL FULLY ACTIVE (Y|N)
130	(82)	CHARACTER	1	CPWSZCEN	Z-CENTRIC WORKSTATION (Y|N)
131	(83)	CHARACTER	1	CPWSRETY	REMOTE ENGINE TYPE (D|Z|blank)
132	(84)	SIGNED	4	CPWSSX	SUM OF SUPPRESSED COND OP
136	(88)	CHARACTER	8	*	RESERVED

CPIVL - Current plan workstation open interval segment

Workstation open interval.

Table 203. CPIVL Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	64	CPIVL	WORKSTATION IVL
0	(0)	CHARACTER	10	CPIVLFR	INTERVAL START
0	(0)	CHARACTER	6	CPIVLFD	DATE YYMMDD

Table 203. CPIVL Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
6	(6)	CHARACTER	4	CPIVLFT	TIME HHMM
10	(A)	CHARACTER	10	CPIVLTO	INTERVAL END
10	(A)	CHARACTER	6	CPIVLTD	DATE YYMMDD
16	(10)	CHARACTER	4	CPIVLTT	TIME HHMM
20	(14)	SIGNED	4	CPIVL#PS	MAX PARALLEL SERVERS
24	(18)	SIGNED	4	CPIVL#DPPS	PS SET BY DAILY PLANNING
28	(1C)	SIGNED	4	CPIVL#R1	CURRENT R1 CAPACITY
32	(20)	SIGNED	4	CPIVL#DPR1	R1 SET BY DAILY PLANNING
36	(24)	SIGNED	4	CPIVL#R2	CURRENT R2 CAPACITY
40	(28)	SIGNED	4	CPIVL#DPR2	R2 SET BY DAILY PLANNING
44	(2C)	UNSIGNED	1	CPIVLVERS	VERSION NUMBER
45	(2D)	CHARACTER	4	CPIVLDPAWS	DP ALTERNATE WORK STATION
49	(31)	CHARACTER	4	CPIVLAWS	CURRENT ALTERNATE WS
53	(35)	CHARACTER	1	CPIVLMOD	Y – MCP MODIFIED OR ADDED
54	(36)	CHARACTER	1	CPIVLDP	Y – ORIGINATES FROM WSD
55	(37)	CHARACTER	9	*	RESERVED

CPOPT - workstation description record segment

Plan workstation record.

Table 204. CPOPT Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE		CPOPT	Workstation options
0	(0)	CHARACTER	47	CPOPTJOBUSR	Default JOBUSER
47	(2F)	CHARACTER	1	CPOPTJOBPWD	Default JOBPWD
48	(2E)	CHARACTER	40	CPOPTJOBTYPE	Default JOBTYPE
88	(58)	CHARACTER	1	CPOPTBROKER	The workstation is a BROKER workstation

Table 204. CPOPT Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
89	(59)	CHARACTER	40	CPOPTPOOL	Pool
129	(81)	CHARACTER	40	CPOPTDYNPOOL	Dynamic pool
169	(44)	CHARACTER	8		Reserved

Note: The creation of dynamic agents, pools and dynamic pools is not supported using PIF. To perform these operations, use the Dynamic Workload Console. To install dynamic agents, run the related installation program.

Current plan virtual workstation destination (resource codes CPWSV, CPWSVCOM)

The current plan virtual workstation destination record consists of the common (CPWSV) segment. One must appear as the first segment in each record. The CPWSV segment can be followed by a variable number of CPIVVL segments that represent the open intervals for the workstation destination.

CPWSV - Common segment

Current plan virtual workstation destination.

Workstation types:

C

Computer workstation

Reporting attribute:

A

Automatic reporting

- The number of started operations is the number of parallel servers in use.
- Y and N are the indicator values.

Table 205. CPWSV Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	137	CPWSV	CURRENT PLAN VIRTUAL WORK STATION
0	(0)	CHARACTER	12	CPWSVKEY	KEY
0	(0)	CHARACTER	4	CPWSVNAM	WORKSTATION NAME

Table 205. CPWSV Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
4	(4)	CHARACTER	8	CPWSVDST	WORKSTATION DESTINATION
12	(C)	CHARACTER	32	CPWSVDESC	DESCRIPTION (NOT USED)
44	(2C)	CHARACTER	12	CPWSVSC	SUM OF COMPLETED OPERATIONS (NOT USED)
44	(2C)	SIGNED	4	CPWSVSC#	NUMBER (NOT USED)
48	(30)	SIGNED	4	CPWSVSCE	ESTIMATED DURATION (NOT USED)
52	(34)	SIGNED	4	CPWSVSCR	REAL DURATION (NOT USED)
56	(38)	CHARACTER	12	CPWSVSI	SUM OF INTERRUPTED OPERATIONS (NOT USED)
56	(38)	SIGNED	4	CPWSVSI#	NUMBER (NOT USED)
60	(3C)	SIGNED	4	CPWSVSIE	ESTIMATED DURATION (NOT USED)
64	(40)	SIGNED	4	CPWSVSIR	REAL DURATION (NOT USED)
68	(44)	CHARACTER	8	CPWSVSS	SUM OF STARTED OPERATIONS
68	(44)	SIGNED	4	CPWSVSS#	NUMBER
72	(48)	SIGNED	4	CPWSVSSE	ESTIMATED DURATION
76	(4C)	CHARACTER	8	CPWSVSR	SUM OF READY OPERATIONS (NOT USED)
76	(4C)	SIGNED	4	CPWSVSR#	NUMBER (NOT USED)
80	(50)	SIGNED	4	CPWSVSRE	ESTIMATED DURATION (NOT USED)
84	(54)	CHARACTER	8	CPWSVSW	SUM OF WAITING OPERATIONS (NOT USED)
88	(58)	SIGNED	4	CPWSVSWE	ESTIMATED DURATION (NOT USED)
92	(5C)	SIGNED	4	CPWSVR1IU#	NUMBER OF RESOURCE 1 IN USE
96	(60)	SIGNED	4	CPWSVR2IU#	NUMBER OF RESOURCE 2 IN USE
100	(64)	SIGNED	4	CPWSVIVL#	NUMBER OF OPEN INTERVALS
104	(68)	CHARACTER	1	CPWSVTYPE	WORK STATION TYPE (C ONLY)
105	(69)	CHARACTER	1	CPWSVREP	REPORTING ATTRIBUTE (A ONLY)
106	(6A)	CHARACTER	1	CPWSVPSC	CONTROL ON PARALLELL SERVERS
107	(6B)	CHARACTER	2	CPWSVR1N	RESOURCE 1 NAME
109	(6D)	CHARACTER	1	CPWSVR1C	RESOURCE 1 USED AT CONTROL (NOT USED)
110	(6E)	CHARACTER	2	CPWSVR2N	RESOURCE 2 NAME

Table 205. CPWSV Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
112	(70)	CHARACTER	1	CPWSVR2C	RESOURCE 2 USED AT CONTROL (NOT USED)
113	(71)	CHARACTER	1	*	JOB SETUP ABILITY (NOT USED)
114	(72)	UNSIGNED	1	CPWSVVERS	VERSION NUMBER=1
115	(73)	CHARACTER	1	CPWSVSTC	STARTED TASK YN
116	(74)	CHARACTER	1	*	DEADLINE WTO Y|N (NOT USED)
117	(75)	CHARACTER	1	CPWSVSTAT	WORK STATION STATUS A|O|F
118	(76)	CHARACTER	1	*	REROUTE MODE (NOT USED)
119	(77)	CHARACTER	4	*	ALTERNATE WS (NOT USED)
123	(7B)	CHARACTER	1	*	NOT USED
124	(7C)	CHARACTER	1	*	LINK WS STATUS L|U (NOT USED)
125	(7D)	CHARACTER	1	*	FULL LINKED Y|N (NOT USED)
126	(7E)	CHARACTER	1	*	SYSTEM AUTOMATION WS (NOT USED)
127	(7F)	CHARACTER	1	*	VIRTUAL WS (NOT USED)
128	(80)	CHARACTER	8	*	DESTINATION (NOT USED)
136	(88)	CHARACTER	1	*	WAIT WORKSTATION Y|N (NOT USED)

CPVIVL - Current plan virtual workstation destination open interval segment

Workstation open interval.

Table 206. CPVIVL Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	64	CPVIVL	VIRTUAL WORK STATION DESTINATION OPEN INTERVAL
0	(0)	CHARACTER	10	CPVIVLFR	INTERVAL START
0	(0)	CHARACTER	6	CPVIVLFD	DATE YYMMDD
6	(6)	CHARACTER	4	CPVIVLFT	TIME HHMM
10	(A)	CHARACTER	10	CPVIVLTO	INTERVAL END

Table 206. CPVIVL Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
10	(A)	CHARACTER	6	CPVIVLTD	DATE YYMMDD
16	(10)	CHARACTER	4	CPVIVLTT	TIME HHMM
20	(14)	SIGNED	4	CPVIVL#PS	MAX PARALLEL SERVERS
24	(18)	SIGNED	4	CPVIVL#DPPS	PARALLEL SERVERS SET AT DAILY PLANNING
28	(1C)	SIGNED	4	CPVIVL#R1	CURRENT RESOURCE CAPACITY
32	(20)	SIGNED	4	CPVIVL#DPR1	CAPACITY SET AT DAILY PLANNING
36	(24)	SIGNED	4	CPVIVL#R2	CURRENT RESOURCE CAPACITY
40	(28)	SIGNED	4	CPVIVL#DPR2	CAPACITY SET AT DAILY PLANNING
44	(2C)	UNSIGNED	1	CPVIVLVERS	VERSION NUMBER
45	(2D)	CHARACTER	4	*	FREE
49	(31)	CHARACTER	4	*	FREE
53	(35)	CHARACTER	1	CPVIVLMOD	Y - MCP MODIFIED OR ADDED
54	(36)	CHARACTER	1	CPVIVLDP	Y - ORIGINATES FROM WSD
55	(37)	CHARACTER	9	*	FREE

Operation critical successors (resource code CRITSUCS)

Critical successors of an operation.

Table 207. CRITSUCS Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE		CRITJOBS	CRITICAL OPERATION
0	(0)	CHARACTER	16	CRITadId	APPLICATION ID
16	10	CHARACTER	4	CRITJwsn	WORKSTATION NAME
20	14	SIGNED	4	CRITJopNo	OPERATION NUMBER
24	18	SIGNED	4	CRITConfFactor	CONFIDENCE FACTOR
28	1C	CHARACTER	20	*	RESERVED
48	30	CHARACTER	8	CRITJjobN	OP OS JOBNAME | BLANK

Table 207. CRITSUCS Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
56	38	CHARACTER	8	*	RESERVED
64	40	CHARACTER	10	CRITJls	LATEST ARRIVAL TIME
64	40	CHARACTER	6	CRITJlsD	DATE | BLANK
70	46	CHARACTER	4	CRITJlsT	TIME | BLANK
74	4A	CHARACTER	10	CRITJoi	OPERATION INPUT ARRIVAL
74	4A	CHARACTER	6	CRITJoiD	DATE | BLANK
80	50	CHARACTER	4	CRITJoiT	TIME | BLANK
84	54	CHARACTER	10	CRITJps	PLANNED ARRIVAL
84	54	CHARACTER	6	CRITJpsD	DATE | BLANK
90	5A	CHARACTER	4	CRITJpsT	TIME | BLANK
94	5E	CHARACTER	10	CRITJas	ACTUAL ARRIVAL
94	5E	CHARACTER	6	CRITJasD	DATE | BLANK
100	64	CHARACTER	4	CRITJasT	TIME | BLANK
104	68	CHARACTER	10	CRITJod	OPERATION DEADLINE
104	68	CHARACTER	6	CRITJodD	DATE | BLANK
110	6E	CHARACTER	4	CRITJodT	TIME | BLANK
114	72	CHARACTER	10	CRITJae	ACTUAL END
114	72	CHARACTER	6	CRITJaeD	DATE | BLANK
120	78	CHARACTER	4	CRITJaeT	TIME | BLANK
124	7C	CHARACTER	4	*	RESERVED
128	80	CHARACTER	1	CRITJopSt	CURRENT STATUS
129	81	CHARACTER	7	*	RESERVED
136	88	CHARACTER	1	CRITJlate	LATE OPERATION Y|N|<BLANK>
137	89	CHARACTER	1	CRITJurgProm	PROMOTED TO URGENT Y|N|<BLANK>
138	8A	CHARACTER	1	CRITJwlmProm	PROMOTED TO WLM Y|N|<BLANK>
139	8B	CHARACTER	1	*	RESERVED
140	8C	CHARACTER	1	CRITJlongRun	LONG RUNNING OPERATION Y|N|<BLANK>

Table 207. CRITSUCS Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
141	8D	CHARACTER	1	*	RESERVED
142	8E	CHARACTER	10	CRITJes	ESTIMATED START
142	8E	CHARACTER	6	CRITJesD	DATE | BLANK
148	94	CHARACTER	4	CRITJesT	TIME | BLANK
152	98	CHARACTER	10	CRITJee	ESTIMATED END
152	98	CHARACTER	6	CRITJeeD	DATE | BLANK
158	9E	CHARACTER	4	CRITJeeT	TIME | BLANK
162	A2	CHARACTER	49	*	RESERVED
211	D3	CHARACTER	1	CRITJisonPATH	Y=INPUT JOB IS ON CRIT PATH N=INPUT JOB IS ON CRIT NETWORK

Current plan special resource (resource codes CSR, CSRCOM)

A current plan special resource consists of four segments:

CSRCOM

Common segment.

CSRIVL

Special resource interval segment.

CSRIWS

Special resource interval workstation segment.

CSRDWS

Special resource default workstation segment.

CSRIVL and CSRDWS are subsegments to CSRCOM. CSRIWS is a subsegment to CSRIVL.

Note: For a correct interpretation of the fields described as "Tod clock at last update", see TOD fields on page 370.

CSRCOM - Current plan resource common segment

Note:

1. Fields in CSRLSTEXT are set only at LIST requests
2. CSROVAV, blank means no overriding availability
3. CSROVQ , zero means no overriding quantity
4. For REPLACE request: Fields marked by (R) below are updated. Other fields are either the identifier, set implicitely or cannot be changed, except for the identifier their values are ignored.

Table 208. CSRCOM Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	240	CSRCOM	RESOURCE INSTANCE STRUCTURE
0	(0)	CHARACTER	44	CSRNAME	SPECIAL RESOURCE NAME
44	(2C)	CHARACTER	8	CSRRODM	RODM SETTING (N|I|P|A)
44	(2C)	CHARACTER	1	CSRRODMA	AVAILABILLITY
45	(2D)	CHARACTER	1	CSRRODMQ	QUANTITY
46	(2E)	CHARACTER	1	CSRRODMD	DEVIATION
47	(2F)	CHARACTER	5	*	RESERVED
52	(34)	CHARACTER	8	CSRGROUP	GROUP ID
60	(3C)	CHARACTER	1	CSRHIPER	DLF RESOURCE (Y|N)
61	(3D)	CHARACTER	1	CSRUSEDFOR	(R) USED FOR (N|P|C|B)
62	(3E)	CHARACTER	2	CSRONERROR	(R) ON ERROR (F |FX|FS|K)
64	(40)	CHARACTER	3	*	RESERVED
67	(43)	CHARACTER	1	CSROVAV	(R) OVERRID AVAILABILITY(Y|N)
68	(44)	SIGNED	4	SROVQ	(R) OVERRID QUANT, 0 IF NONE
72	(48)	SIGNED	4	CSRDEVI	(R) DEVIATION
76	(4C)	SIGNED	4	CSRIVLNUM	NUMBER OF INTERVALS
80	(50)	SIGNED	4	CSRCIVLN	CURRENT INTERVAL NUMBER
84	(54)	CHARACTER	46	CSRDESC	DESCRIPTION
130	(82)	CHARACTER	10	CSRLIFTIEDAT	LIFESPAN EXPIRATION DATE AND TIME
138	(8A)	CHARACTER	10	*	RESERVED
140	(8C)	SIGNED	4	CSRDEFNWSC	NUMBER CONNECTED WORKSTATIONS
144	(90)	SIGNED	4	CSRDEFQUANT	(R) DEFAULT QUANTITY

Table 208. CSRCOM Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
148	(94)	CHARACTER	1	CSRDEFAVAIL	(R) DEFAULT AVAILABILITY
149	(95)	CHARACTER	1	CSRLIFTIEACT	LIFESPAN ACTION (Y|N|R)
150	(96)	CHARACTER	1	CSRONCOMPL	(R) ON COMPLETE (Y|N|R OR BLANK)
151	(97)	CHARACTER	1	CSRMAXTYPE	(R) MAX USAGE TYPE (Y|N|R)
152	(98)	CHARACTER	8	CSRLUSER	LAST UPDATING USER
160	(A0)	CHARACTER	6	CSRLDATE	DATE OF LAST UPDATE
166	(A6)	CHARACTER	4	CSRLTIME	TIME OF LAST UPDATE
170	(AA)	CHARACTER	8	CSRLUTS	TOD CLOCK LAST UPDATE
178	(B2)	UNSIGNED	1	CSRVER	RECORD VERSION
179	(B3)	CHARACTER	1	CSRACTAVAIL	ACTUAL AVAILABILITY
180	(B4)	SIGNED	4	CSRACTQUANT	ACTUAL QUANTITY
184	(B8)	SIGNED	4	CSRUSAGECNT	(R) USAGE COUNTER
188	(BC)	SIGNED	4	CSRMAXLIMIT	(R) MAX USAGE LIMIT
192	(C0)	CHARACTER	48	CSRLISTX	SET AT LIST REQUEST ONLY
192	(C0)	SIGNED	4	CSRXUSE	AMOUNT CURRENTLY USED EXCL
196	(C4)	SIGNED	4	CSRSUSE	AMOUNT CURRENTLY USED SHARED
200	(C8)	CHARACTER	1	CSRXALL	EXCLUSIVE USER NOW (Y|N)
201	(C9)	CHARACTER	1	CSRSALL	SHARED USER NOW (Y|N)
202	(CA)	CHARACTER	1	CSRWAITQ	ANY ON WAIT QUEUE (Y|N)
203	(CB)	CHARACTER	1	CSRLASTM	LAST MODIFY TYPE
208	(D0)	CHARACTER	32	CSRCURIVL	CURRENT INTERVAL DATA
208	(D0)	CHARACTER	6	CSRCIDATE	DATE
214	(D6)	CHARACTER	2	*	RESERVED
216	(D8)	CHARACTER	4	CSRCIFTIME	FROM TIME
220	(DC)	CHARACTER	4	CSRCITTIME	TO TIME
224	(E0)	SIGNED	4	CSRCIQUANT	ALLOCATION CAPACITY
228	(E4)	SIGNED	4	CSRCIADJQ	ADJUST QUANTITY

Table 208. CSRCOM Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
232	(E8)	CHARACTER	1	CSRCIAVAIL	AVAILABLE (Y|N)
233	(E9)	CHARACTER	7	*	RESERVED

CSRIVL - Current plan special resource interval segment

Table 209. CSRIVL Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	32	CSRIVL	INTERVAL
0	(0)	CHARACTER	6	CSRIDATE	SPECIFIC DATE
6	(6)	CHARACTER	2	*	RESERVED
8	(8)	CHARACTER	4	CSRIFTIME	FROM TIME
12	(C)	CHARACTER	4	CSRITTIME	TO TIME
16	(10)	SIGNED	4	CSRIQUANT	ALLOCATABLE AMOUNT
20	(14)	SIGNED	4	CSRIWSCNUM	NUMBER OF CONNECTED WORKSTATIONS
24	(18)	CHARACTER	1	CSRIAVAIL	AVAILABLE (Y|N)
25	(19)	CHARACTER	7	*	RESERVED

CSRIWS - Current plan resource interval "connected" workstation

Table 210. CSRIWS Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	8	CSRIWS	CONNECTED WS SEGMENT
0	(0)	CHARACTER	4	CSRIWSNAME	WORKSTATION NAME
4	(4)	CHARACTER	4	*	RESERVED

CSRDWS - Current plan resource default "connected" workstation

Table 211. CSRDWS Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	8	CSRDWS	CONNECTED WS SEGMENT
0	(0)	CHARACTER	4	CSRDWSNAME	WORKSTATION NAME
4	(4)	CHARACTER	4	*	RESERVED

ETT - Event triggered tracking criteria segment

Table 212. ETT Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	BASED	128	ETT	ETT TRACKING CRITERIA REC
0	(0)	CHARACTER	64	ETTKEY	KEY
0	(0)	CHARACTER	1	ETTTYPE	RECORD TYPE=EVENT TYPE: 2 = JOB, 3 = RESOURCE
1	(1)	CHARACTER	44	ETTNAME	NAME OF TRIGGERING EVENT
45	(2D)	CHARACTER	19	*	RESERVED
64	(40)	CHARACTER	2	ETTVERS	RECORD VERSION
66	(42)	CHARACTER	1	*	RESERVED
67	(43)	CHARACTER	16	ETTAPPL	CORRESPONDING APPLICATION
83	(53)	CHARACTER	1	ETTJREP	JOB REPLACE: Y=YES, N=NO
84	(54)	CHARACTER	8	ETTLUSER	USED OF LAST UPDATED
92	(5C)	CHARACTER	6	ETTLDATE	DATE OF LAST UPDATE
98	(62)	CHARACTER	4	ETTLTIME	TIME OF LAST UPDATE
102	(66)	CHARACTER	8	*	RESERVED
110	(6E)	CHARACTER	1	ETTDEPR	DEP RESOLUTION: Y=YES, N=NO
111	(6F)	CHARACTER	1	ETTASSW	AVAIL STATUS: Y=YES, N=NO
112	(70)	CHARACTER	8	ETTLUTS	TOD CLOCK AT LAST UPDATE
120	(78)	CHARACTER	8	*	RESERVED

Dates generated by run cycle rules (resource code GENDAYS)

The output of a LIST GENDAYS request includes both the original dates and the dates that come from a change in the free day rule. A set of flags provide information about the free day rule actions on the date. The output is made up by the **GNDAY** segment:

Table 213. GNDAY Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	*	GNDAY	RUN DAY GENERATED BY GENDAYS
0	(0)	CHARACTER	6	GNDAYDATE	GENERATED RUN DAY DATE (YYMMDD)
6	(6)	CHARACTER	1	GNDAYFMOB	ORIGINAL DATE: MOVED BEFORE BECAUSE OF FREE DAY RULE (Y/N)
7	(7)	CHARACTER	1	GNDAYFMOA	ORIGINAL DATE: MOVED AFTER BECAUSE OF FREE DAY RULE (Y/N)
8	(8)	CHARACTER	1	GNDAYFKEP	ORIGINAL DATE: KEPT BECAUSE OF FREE DAY RULE (Y/N)
9	(9)	CHARACTER	1	GNDAYFCAN	ORIGINAL DATE: CANCELLED BECAUSE OF FREE DAY RULE (Y/N)
10	(A)	CHARACTER	1	GNDAYFEIA	RUN ON FREE DAY - EARLY INPUT ARRIVAL TIME (Y/N)
11	(B)	CHARACTER	1	GNDAYFOUT	ORIGINAL DATE: MOVED OUTSIDE INTERVAL BECAUSE OF FREE DAY RULE (Y/N)
12	(C)	CHARACTER	1	GNDAYFREM	NEW WORK DATE: OUTSIDE INTERVAL (Y/N)
13	(D)	CHARACTER	7	*	RESERVED

JCL setup variables (resource codes JCLPREP, JCLPREPA

A JCL setup variable record (JSV) can contain these segments:

JSVC

Fixed part of the promptable variables.

JSVV

Variable update part of the promptable variables.

JSVC - Common segment

Common part of JCL setup for promptable variables.

Table 214. JSVC Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	35	JSVC	PROMPTABLE VARIABLES
0	(0)	CHARACTER	35	JSVCCOM	IDENTIFIER
0	(0)	CHARACTER	32	JSVCKEY	KEY OF OPERATION
0	(0)	CHARACTER	16	JSVCADID	APPLICATION ID
16	(10)	CHARACTER	6	JSVCIAD	INPUT ARRIVAL DATE YYMMDD
22	(16)	CHARACTER	2	*	RESERVED
24	(18)	CHARACTER	4	JSVCIAT	INPUT ARRIVAL TIME HHMM
28	(1C)	SIGNED	4	JSVCOPNO	OPERATION NUMBER
32	(20)	SIGNED	2	JSVC#VARS	NUMBER OF VARIABLES
34	(22)	CHARACTER	1	JSVCFROM	JCL FROM JS REPOSITORY Y|N

JSVV - Variable definition segment

Update part of JCL setup for promptable variables.

Table 215. JSVV Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	53	JSVV	PROMPTABLE VARIABLES
0	(0)	CHARACTER	8	JSVVNAME	VARIABLE NAME
8	(8)	CHARACTER	44	JSVVVALUE	VALUE SET OR DEFAULT VALUE
52	(34)	CHARACTER	1	JSVVTYPE	USAGE TYPE (%|&|?)

JCL variable table (resource codes JCLV, JCLVCOM)

A JCL variable table record (JCLV) can contain these segments:

JCLVC

Common part of the JCL variable table record

JCLVV

Variable definition part of the JCL variable table record

JCLVD

Dependency part of a JCL variable table record.

Note: For a correct interpretation of the fields described as "Tod clock at last update", see TOD fields on page 370.

JCLVC - Common segment

Identifies a JCL variable table.

Table 216. JCLVC Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	96	JCLVC	COMMON PART
0	(0)	CHARACTER	96	JCLVCCOM	IDENTIFIER
0	(0)	CHARACTER	1	*	RESERVED
1	(1)	CHARACTER	16	JCLVCKEY	KEY OF RECORD TABLE
1	(1)	CHARACTER	16	JCLVCTAB	JCL VARIABLE TABLE ID
17	(11)	CHARACTER	1	*	RESERVED
18	(12)	CHARACTER	8	JCLVCLU	LAST UPDATING USER
26	(1A)	CHARACTER	4	JCLVCLT	LAST UPDATE TIME HHMM
30	(1E)	CHARACTER	6	JCLVCLD	LAST UPDATE DATE YYMMDD
36	(24)	SIGNED	2	JCLVC#V	NUMBER OF VARIABLES IN TABLE
38	(26)	CHARACTER	24	JCLVCDSC	DESCRIPTION
62	(3E)	CHARACTER	16	JCLVCOWN	OWNER ID
78	(4E)	CHARACTER	2	*	RESERVED
80	(50)	CHARACTER	8	JCLVCLUTS	TOD CLOCK AT LAST UPDATE
88	(58)	CHARACTER	8	*	RESERVED

JCLVV - Variable definition segment

Defines a JCL variable.

Table 217. JCLVV Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	464	JCLVV	JCL VARIABLE DEFINITIONS
0	(0)	CHARACTER	8	JCLVVVAR	JCL VARIABLE NAME
8	(8)	CHARACTER	44	JCLVVDFL	JCL VARIABLE DEF VALUE
52	(34)	CHARACTER	1	JCLVVSTP	PROMPT | SETUP | SUBMIT
53	(35)	CHARACTER	1	JCLVVUC	UPPER CASE (Y|N)
54	(36)	SIGNED	2	JCLVVLG	VALUE LENGTH
56	(38)	CHARACTER	7	JCLVVTYP	VERIFICATION TYPE
63	(3F)	CHARACTER	8	JCLVVEX	SUBSTITUTION EXIT NAME
71	(47)	CHARACTER	1	JCLVVINP	INPUT REQUIRED
72	(48)	SIGNED	2	JCLVVPOS	REPLACE POSITION IN JCL DATA
74	(4A)	CHARACTER	1	JCLVVNUM	NUMERIC
75	(4B)	CHARACTER	2	JCLVVCMP	COMPARISON OPERATOR
77	(4D)	CHARACTER	44	JCLVVPAT	VALIDATION PATTERN
121	(79)	CHARACTER	102	JCLVVVLD	VALID VALUES
223	(DF)	CHARACTER	204	JCLVVTXT	DIALOG TEXT
427	(1AB)	CHARACTER	20	JCLVVDES	DESCRIPTION
447	(1BF)	CHARACTER	1	*	RESERVED
448	(1C0)	SIGNED	2	JCLVVNRP	NUMBER OF DEPENDENT VALUES
450	(1C2)	CHARACTER	8	JCLVVIND	INDEPENDENT VARIABLE NAME
458	(1CA)	CHARACTER	2	JCLVVVER	RECORD VERSION NUMBER=1
460	(1CC)	CHARACTER	2	JCLVVSUS	SUBSTRING START POSITION
462	(1CE)	CHARACTER	2	JCLVVSUL	SUBSTRING LENGTH

Note: JCLVVVLD is 2 lines each of 51 characters. If values continue to the second line, the first line must end with a comma.

JCLVD - Dependency segment

Defines dependencies for a JCL variable.

Table 218. JCLVD Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	88	JCLVD	DEPENDENCY VALUES
0	(0)	CHARACTER	44	JCLVDIV	VALUE OF SETTING VARIABLE
44	(2C)	CHARACTER	44	JCLVDDV	OVERRIDE VALUE FOR DEPEND

Job control language (resource codes JS, JSCOM)

A job control language record consists of only one segment, but there are two forms to choose from:

JSCOM

Job control language segment excluding JCL lines.

JS

Job control language segment including JCL lines. The text that starts at field JST is included.

JS - Job control language segment

Description of the JCL of an operation.

Status can be:

S

Submitted.

T

Temporarily saved.

V

Saved.

C

Complete.

Blank. The JCL was not retrieved from the JS data set.

Last updating function can be:

L

LTP

W

WSD

R

RL

S

Submit

M

MCP

P

PIF

Table 219. JS Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	96	JS	JCL OF AN OPERATION
0	(0)	CHARACTER	30	JSKEY	KEY
0	(0)	CHARACTER	16	JSADID	APPLICATION ID
16	(10)	CHARACTER	10	JSIA	OCCURRENCE INPUT ARRIVAL
16	(10)	CHARACTER	6	JSIAD	DATE
22	(16)	CHARACTER	4	JSIAT	TIME
26	(1A)	SIGNED	4	JSOPNO	OPERATION NUMBER
30	(1E)	CHARACTER	8	JSJOBN	JOBNAME
38	(26)	CHARACTER	4	JSWSN	WORKSTATION NAME
42	(2A)	CHARACTER	1	JSST	STATUS
43	(2B)	CHARACTER	1	JSUPDT	LAST UPDATING FUNCTION
44	(2C)	CHARACTER	10	JSLUPD	LAST UPDATED
44	(2C)	CHARACTER	6	JSLDATE	DATE
50	(32)	CHARACTER	4	JSLTIME	TIME
54	(36)	CHARACTER	8	JSLUSER	USERID OF LAST UPDATER
62	(3E)	UNSIGNED	1	JSVERS	RECORD VERSION NUMBER=1
63	(3F)	CHARACTER	1	*	RESERVED
64	(40)	SIGNED	4	JSLINES	NUMBER OF TEXT ROWS
68	(44)	CHARACTER	1	JSJFROM	JCL FROM JS REPOSITORY Y|N

Table 219. JS Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
69	(45)	CHARACTER	27	*	RESERVED
96	(60)	CHARACTER		JST	START OF TEXT ROWS. THE LENGTH OF EACH ROW IS 80 CHARACTERS.

Job log (resource code JLCOM)

A job log record (JLC) consists of one segment:

JLCOM

Common segment.

JLCOM - Common segment

Common part of job log.

Table 220. JLCOM Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	64	JLCOM	JOBLOG OF AN OPERATION
0	(0)	CHARACTER	30	JLKEY	KEY
0	(0)	CHARACTER	16	JLADID	APPLICATION ID
16	(10)	CHARACTER	10	JLIA	OCC. INPUT ARRIVAL YMMDDHHMM
16	(10)	CHARACTER	6	JLIAD	INPUT ARRIVAL DATE YYMMDD
22	(16)	CHARACTER	4	JLIAT	INPUT ARRIVAL TIME HHMM
26	(1A)	SIGNED	4	JLOPNO	OPERATION NUMBER
30	(1E)	CHARACTER	8	JLJOBN	JOB NAME
38	(26)	CHARACTER	4	JLWSN	WORKSTATION NAME
42	(2A)	CHARACTER	8	JLJOBID	JES JOB NUMBER
50	(32)	CHARACTER	14	*	RESERVED

Long-term plan occurrence (resource codes LTOC, LTOCCOM)

Each LTP occurrence can contain these segments:

LTOC

Common segment. Only one must always exist.

LTOP

Operation segment.

LTCPRE

Conditional predecessor segment.

LTCSUC

Conditional successor segment.

LTPRE

Predecessor segment.

LTSUC

Successor segment.

LTEXT

External run cycle group for variable duration and deadline.

LTOC - Common segment

LTP occurrence.

Notes:

1. Minutes are the unit of duration.
2. Y and N are the indicator values.

Table 221. LTOC Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	176	LTOC	LONG-TERM PLAN OCCURRENCE
0	(0)	CHARACTER	26	LTOCKEY	OCCURRENCE IDENTIFIER
0	(0)	CHARACTER	6	LTOCIAD	RUN DATE
6	(6)	CHARACTER	16	LTOCADI	APPLICATION ID
22	(16)	CHARACTER	4	LTOCIAT	INPUT ARRIVAL TIME
26	(1A)	CHARACTER	10	LTOCIAO	ORIGINAL INPUT ARRIVAL
26	(1A)	CHARACTER	6	LTOCIAOD	DATE

Table 221. LTOC Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
32	(20)	CHARACTER	4	LTOCIAOT	TIME
36	(24)	CHARACTER	10	LTOCDL	DEADLINE
36	(24)	CHARACTER	6	LTOCDLD	DATE
42	(2A)	CHARACTER	4	LTOCDLT	TIME
46	(2E)	CHARACTER	8	LTOCGRP	AUTHORITY GROUP
54	(36)	CHARACTER	16	LTOCOID	OWNER ID
70	(46)	CHARACTER	4	LTOCERR	OCCURRENCE ERROR CODE
74	(4A)	CHARACTER	1	LTOCRDST	RUN DAY STATUS W|F
75	(4B)	UNSIGNED	1	LTOCVERS	VERSION NUMBER=1
76	(4C)	SIGNED	4	LTOCPRI	PRIORITY
80	(50)	SIGNED	4	LTOC#PRE	NUMBER OF EXTERNAL PREDECESSORS
84	(54)	SIGNED	4	LTOC#SUC	NUMBER OF EXTERNAL SUCCESSORS
88	(58)	SIGNED	4	LTOC#OP	NUMBER OF CHANGED OPERATIONS
92	(5C)	CHARACTER	1	LTOCDEL	DELETED ONLINE
93	(5D)	CHARACTER	1	LTOCADD	ADDED TO LTP
94	(5E)	CHARACTER	1	LTOCMOD	MODIFIED IN LTP
95	(5F)	CHARACTER	1	LTOCMOV	RUN DATE OR TIME MODIFIED
96	(60)	CHARACTER	1	LTOCDEPM	EXTERNAL DEPENDENCY MODIFIED
97	(61)	CHARACTER	1	LTOCCOMP	COMPLETED BY JOB TRACKING
98	(62)	CHARACTER	1	LTOCMOVO	MOVED BECAUSE OF OPTIONAL RULE
99	(63)	CHARACTER	16	LTOJVT	JCL VARIABLE TABLE
115	(73)	CHARACTER	16	LTGROUPID	GROUP DEFINITION ID
131	(83)	CHARACTER	16	LTOCCAL	CALENDAR NAME
147	(93)	CHARACTER	1	*	RESERVED
148	(94)	CHARACTER	4	LTOC#CPRE	NUMBER OF CONDITIONAL PREDECESSORS
152	(98)	CHARACTER	4	LTOC#CSUC	NUMBER OF CONDITIONAL SUCCESSORS
156	(9C)	CHARACTER	4	*	UNUSED

Table 221. LTOC Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
160	(A0)	SIGNED	4	LTOC#MAND	NUMBER OF MANDATORY PENDING PREDECESSORS
164	(A4)	CHARACTER	8	LTOCRUNC	RUN CYCLE THAT GENERATED THE OCCURRENCE
172	(A5)	SIGNED	8	LTOCRUNN	NUMBER OF EXTERNAL RUN CYCLE GROUPS FOR VDD

LTOP - Operation segment

LTP changed operation.

Note: Y and N are the indicator values.

Table 222. LTOP Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	64	LTOP	CHANGED OPERATION
0	(0)	CHARACTER	4	LTOPWSN	WORKSTATION NAME
4	(4)	SIGNED	4	LTOPNO	OPERATION NUMBER
8	(8)	CHARACTER	10	LTOPOI	OPERATION INPUT ARRIVAL
8	(8)	CHARACTER	6	LTOPOID	DATE | BLANK
14	(E)	CHARACTER	4	LTOPOIT	TIME | BLANK
18	(12)	CHARACTER	10	LTOPOD	OPERATION DEADLINE
18	(12)	CHARACTER	6	LTOPODD	DATE | BLANK
24	(18)	CHARACTER	4	LTOPODT	TIME | BLANK
28	(1C)	CHARACTER	24	LTOPDESC	OPERATION TEXT
52	(34)	UNSIGNED	1	LTOPVERS	VERSION NUMBER=1
53	(35)	CHARACTER	11	*	RESERVED

LTCPRE- Conditional predecessor segment.

LTP conditional predecessor.

Example

```
Offsets      Type     Length  Name          Description
    0    (0) STRUCTURE    33  LTCPRE        OCCURRENCE CONDITIONAL PREDECESSOR
    0    (0) CHARACTER    26  LTCPREKEY     CONDITIONAL PREDECESSOR IDENTIFIER
    0    (0) CHARACTER     6  LTCPREIAD     RUN DATE
    6    (6) CHARACTER    16  LTCPREADI     APPLICATION ID
   22   (16) CHARACTER     4  LTCPREIAT     INPUT ARRIVAL TIME
   26   (1A) CHARACTER     1  LTCPREDEL     DEPENDENCY DELETED
   27   (1B) CHARACTER     1  LTCPREPDONE   PREDECESSOR COMPLETED
   28   (1C) CHARACTER     1  LTCPREMPEND   IF Y IT IS A MANDATORY PENDING
   29   (1D) CHARACTER     1  LTCPREMAND    REQUIRED VALUE: C,P,OR N
   30   (1E) UNSIGNED      1  LTCPREVERS    VERSION NUMBER
   31   (1F) CHARACTER     2  *             UNUSED
```

LTCSUC- Conditional successor segment.

LTP conditional successor.

Example

```
Offsets      Type     Length  Name          Description
    0    (0) STRUCTURE    32  LTCSUC        OCCURRENCE CONDITIONAL SUCCESSOR
    0    (0) CHARACTER    26  LTCSUCKEY     CONDITIONAL SUCCESSOR IDENTIFIER
    0    (0) CHARACTER     6  LTCSUCIAD     RUN DATE
    6    (6) CHARACTER    16  LTCSUCADI     APPLICATION ID
   22   (16) CHARACTER     4  LTCSUCIAT     INPUT ARRIVAL TIME
   26   (1A) CHARACTER     1  LTCSUCDEL     DEPENDENCY DELETED
   27   (1B) CHARACTER     2  *             UNUSED
   29   (1D) UNSIGNED      1  LTCSUCVERS    VERSION NUMBER
   30   (1E) CHARACTER     2  *             UNUSED
```

LTPRE - Predecessor segment

LTP occurrence predecessor.

Note: Y and N are the indicator values.

Table 223. LTPRE Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	32	LTPRE	OCCURRENCE PREDECESSOR
0	(0)	CHARACTER	26	LTPREKEY	PREDECESSOR IDENTIFIER
0	(0)	CHARACTER	6	LTPREIAD	RUN DATE
6	(6)	CHARACTER	16	LTPREADI	APPLICATION ID
22	(16)	CHARACTER	4	LTPREIAT	INPUT ARRIVAL TIME
26	(1A)	CHARACTER	1	LTPREDEL	DEPENDENCY DELETED

Table 223. LTPRE Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
27	(1B)	CHARACTER	1	LTPREADD	MANUALLY ADDED
28	(1C)	CHARACTER	1	LTPREDONE	PREDECESSOR COMPLETED
29	(1D)	UNSIGNED	1	LTPREVERS	VERSION NUMBER=1
30	(1E)	CHARACTER	1	LTPREMPEND	Y: IS MANDATORY PENDING
31	(1F)	CHARACTER	1	LTPREMAND	C|P|N IS A REQUIRED VALUE

LTSUC - Successor segment

Table 224. LTSUC Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	32	LTSUC	OCCURRENCE SUCCESSOR
0	(0)	CHARACTER	26	LTSUCKEY	SUCCESSOR IDENTIFIER
0	(0)	CHARACTER	6	LTSUCIAD	RUN DATE
6	(6)	CHARACTER	16	LTSUCADI	APPLICATION ID
22	(16)	CHARACTER	4	LTSUCIAT	INPUT ARRIVAL TIME
26	(1A)	CHARACTER	1	LTSUCDEL	DEPENDENCY DELETED
27	(1B)	CHARACTER	1	LTSUCADD	MANUALLY ADDED
28	(1C)	UNSIGNED	1	LTSUCVERS	VERSION NUMBER=1
29	(1D)	CHARACTER	3	*	RESERVED

LTEXT - External run cycle group for variable duration and deadline

Table 225. LTEXT Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	14	LTEXT	EXTERNAL RUN CYCLE GROUP FOR VDD
0	(0)	CHARACTER	6	LTEXTOPID	OPERATION IDENTIFIER
0	(0)	CHARACTER	4	LTEXTOPWS	WORKSTATION

Table 225. LTEXT Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
4	(4)	SIGNED	2	LTEXTOPNUM	OPERATION NUMBER
6	(6)	CHARACTER	8	LTEXTOPRG	EXTERNAL RUN CYCLE GROUP
14	(E)	UNSIGNED	1	LTEXTVERS	VERSION NUMBER = 1

Operator instruction (resource codes OI, OICOM)

The operator instruction record consists of only one segment, but there are two forms to choose from:

OICOM

Operator instruction segment excluding text.

OI

Operator instruction segment including text. The text that starts at field OIT is included.

OI - Operator instruction segment

An operator instruction.

Table 226. OI Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	96	OICOM	OPERATOR INSTRUCTION
0	(0)	CHARACTER	30	OIKEY	KEY
0	(0)	CHARACTER	16	OIADID	APPLICATION ID
16	(10)	SIGNED	4	OIOPNO	OPERATION NUMBER
20	(14)	CHARACTER	10	OITO	VALID TO
20	(14)	CHARACTER	6	OITOD	DATE
26	(1A)	CHARACTER	4	OITOT	TIME
30	(1E)	CHARACTER	10	OIFROM	VALID FROM
30	(1E)	CHARACTER	6	OIFROMD	DATE
36	(24)	CHARACTER	4	OIFROMT	TIME
40	(28)	CHARACTER	4	OIWSN	WORKSTATION NAME
44	(2C)	CHARACTER	8	OIJOBN	JOBNAME

Table 226. OI Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
52	(34)	CHARACTER	10	OILUPD	LAST UPDATED
52	(34)	CHARACTER	6	OILDATE	DATE
58	(3A)	CHARACTER	4	OILTIME	TIME
62	(3E)	CHARACTER	8	OILUSER	USERID OF LAST UPDATER
70	(46)	UNSIGNED	1	OIVERS	RECORD VERSION NUMBER=1
71	(47)	CHARACTER	1	*	RESERVED
72	(48)	SIGNED	4	OILINES	NUMBER OF TEXT ROWS
76	(4C)	CHARACTER	8	OILUTS	TOD CLOCK AT LAST UPDATE
84	(54)	CHARACTER	12	*	RESERVED
96	(60)	CHARACTER		OIT	START OF TEXT ROWS. THE LENGTH OF EACH ROW IS 72 CHARACTERS.

Period (resource codes PR, PRCOM)

A period record consists of only one segment, but there are two forms to choose from:

PRCOM

Period segment excluding origin dates and interval end dates.

PR

Period segment including origin dates and interval end dates. The text that starts at field PRTAB is included.

PR - Period segment

Description of a period. Defines a program interface data area. PRTYPE can be:

A

A cyclic period that includes both work days and free days

W

A cyclic period that includes only work days

N

A noncyclic period

Interval end dates are optional and follow the origin dates array. They are paired with origin dates; the first origin date with the first interval end date, and so on. If the segment contains interval end dates, they must match the number of origin dates, but they can be blank.

Table 227. PR Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	96	PRCOM	PERIOD DEFINITION
0	(0)	CHARACTER	8	PRKEY	UNIQUE IDENTIFIER
0	(0)	CHARACTER	8	PRNAME	PERIOD NAME
8	(8)	UNSIGNED	1	PRVERS	RECORD VERSION=1
9	(9)	CHARACTER	1	PRTYPE	CYCLIC/NONCYCLIC TYPE A|W|N
10	(A)	CHARACTER	30	PRDESC	DESCRIPTION OF PERIOD
40	(28)	SIGNED	4	PRINTVL	INTERVAL OF CYCLIC ORIGINS
44	(2C)	SIGNED	4	PRORIG#	NUMBER OF ORIGIN DATES IN PERIOD
48	(30)	CHARACTER	6	PRLDATE	DATE LAST UPDATED
54	(36)	CHARACTER	4	PRLTIME	TIME LAST UPDATED
58	(3A)	CHARACTER	8	PRLUSER	USERID OF LAST UPDATER
66	(42)	CHARACTER	16	PRJVT	JCL VARIABLE TABLE
82	(52)	CHARACTER	6	*	RESERVED
88	(58)	CHARACTER	8	PRLUTS	TOD CLOCK AT LAST UPDATE
96	(60)	CHARACTER	*	PRTAB	START OF ORIGIN DATES

Note: For a correct interpretation of the fields described as "Tod clock at last update", see TOD fields on page 370.

Table 228. Period Origin Dates

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
96	(60)	CHARACTER	*	PRTAB	START OF ORIGIN DATES
96	(60)	CHARACTER	6	PRORIG	ORIGIN DATE (YYMMDD)

Table 229. Period Interval End Dates

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
		CHARACTER	6	PRIVLEND	INTERVAL END DATE (YYMMDD) (PRTAB+(PRORIG# * 6))

Run cycle group (resource codes RG, RGCOM)

A run cycle group record can contain these segments:

RGCOM

Common segment. Only one common segment must appear as the first segment in each record.

RGRUN

Run cycle group segment. One segment for every run cycle in the group.

RGCOM - Common segment

The common part of a run cycle group.

The reserved fields marked by an * in the name column should be treated as record data. Their value should be preserved when a record is updated and set to zero when a new segment is created.

Table 230. RGCOM Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	160	RGCOM	COMMON SECTION OF RG
0	(0)	CHARACTER	8	RGKEY	KEY
0	(0)	CHARACTER	8	RGID	RUN CYCLE GROUP ID
8	(8)	CHARACTER	4	RGIAT	DEFAULT INPUT ARRIVAL TIME
12	(C)	CHARACTER	16	RGJVTAB	DEFAULT JCL VARIABLE TABLE
28	(1C)	CHARACTER	16	RGCAL	DEFAULT CALENDAR
44	(2C)	CHARACTER	50	RGDESC	RUN CYCLE GROUP DESCRIPTION
94	(5E)	CHARACTER	8	RGLUSER	USERID OF LAST UPDATER
102	(66)	CHARACTER	6	RGLDATE	DATE OF LAST UPDATE
108	(6C)	CHARACTER	4	RGLTIME	TIME OF LAST UPDATE
112	(70)	CHARACTER	8	RGLUTS	TOD CLOCK AT LAST UPDATE

Table 230. RGCOM Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
120	(78)	UNSIGNED	1	RGCOMVERS	RECORD VERSION NUMBER
121	(79)	CHARACTER	3	*	RESERVED
124	(7C)	CHARACTER	16	RGOWNER	OWNER
140	(8C)	SIGNED	4	RGDD	DEFAULT DEADLINE DATE
144	(90)	CHARACTER	4	RGDT	DEFAULT DEADLINE TIME
148	(94)	CHARACTER	8	*	RESERVED

RGRUN - Run cycle segment

Each run cycle in a run cycle group. The run cycles of a run cycle group are based on rules only. The segment contains the fixed part plus the rule definition.

Type

Required input.

The type can be one of the following:

R

Regular run cycle that identifies times and days when the application runs.

E

Exclusion run cycle that identifies times and days when the application does NOT run. If you specify a particular day and time as an exclusion run cycle, no occurrences of the application are generated for that day and time, regardless of what is generated by a regular or normal run cycle. Run cycles are used in conjunction; exclusion run cycles are used to suppress run days generated by regular or normal run cycles.

A

Rule-based run cycle group or subset. Applies to all the run cycles within a run cycle group or a run cycle group subset.

D

Exclusion rule-based run cycle group or subset. Applies to all the run cycles within a run cycle group or a run cycle group subset.

Free day rule

Required input for all run cycles, which indicates how run days are treated:

E

Free days excluded; only work days are taken into account

1

Free days included; run on the nearest day *before* the free day

2

Free days included; run on the nearest day *after* the free day

3

Free days included; run *on* the free day

4

Free days included; do *not* run at all.

Table 231. RGRUN Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	160	RGRUN	RUN CYCLE SECTION
0	(0)	CHARACTER	8	RGRNAME	RULE NAME
8	(8)	CHARACTER	6	RGRVALF	RUN CYCLE VALID-FROM
14	(E)	CHARACTER	6	RGRVALT	RUN CYCLE VALID-TO
20	(14)	CHARACTER	50	RGRDESC	RUN CYCLE DESCRIPTION
70	(46)	CHARACTER	1	RGRRULE	RULE FOR WORK/FREE DAYS
71	(47)	CHARACTER	1	RGRTYPE	TYPE (R | E | A | D)
72	(48)	CHARACTER	4	RGRIAT	INPUT ARRIVAL TIME
76	(4C)	UNSIGNED	1	RGRUNVERS	RECORD VERSION NUMBER
77	(4D)	CHARACTER	3	*	RESERVED
80	(50)	CHARACTER	16	RGRJVTAB	JCL VARIABLE TABLE
96	(60)	CHARACTER	4	*	RESERVED
100	(64)	SIGNED	2	RGRIRDLEN	RULE DEFINITION LENGTH
102	(66)	CHARACTER	4	RGRREPEATEVERY	REPEAT EVERY
106	(6A)	CHARACTER	4	RGRREPEATENDT	REPEAT END TIME
110	(6E)	CHARACTER	8	RGRSETID	RUN CYCLE CORRELATOR
118	(76)	CHARACTER	16	RGRCALENDAR	RUN CYCLE CALENDAR

Table 231. RGRUN Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
134	(86)	CHARACTER	2	*	RESERVED
136	(88)	SIGNED	4	RGDD	DEADLINE DAY RELATIVE TO START
140	(8C)	CHARACTER	4	RGDT	DEADLINE TIME
144	(90)	CHARACTER	16	*	RESERVED
160	(A0)	CHARACTER	*	RGRIADALL	RULE DEFINITION

Table 232. Rule Definition

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
160	(A0)	STRUCTURE	*	RGRIADALL	RULE DEFINITION
160	(A0)	SIGNED	4	RGRULEL	RULE LENGTH (RGRULEL + RGRULET)
164	(A4)	CHARACTER	*	RGRULET	RULE TEXT

RGRIRDLEN identifies the length of the rule definition. The RGRIADALL structure contains a fullword copy of RGRIRDLEN (RGRULEL), which is followed by the rule text. RGRULEL must specify the same length as RGRIRDLEN. You can insert comments or extra blanks when creating a rule, but these characters are not saved in the RG database.

Special resource (resource codes SR, SRCOM)

A special resource consists of four segments:

SRCOM

Common segment which is followed by the first SRIVL segment, the second SRIVL segment, and so forth.

SRIVL

Special resource interval segment.

SRIWS

Special resource interval workstation segment.

SRDWS

Special resource default workstation segment.

SRIVL and SRDWS are subsegments to SRCOM. SRIWS is a subsegment to SRIVL.

Table 233. SRCOM Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	BASED	192	SRCOM	RESOURCE INSTANCE STRUCTURE
0	(0)	CHARACTER	44	SRCKEY	KEY
0	(0)	CHARACTER	44	SRCNAME	SPECIAL RESOURCE NAME
44	(2C)	CHARACTER	8	*	RESERVED
52	(34)	CHARACTER	8	SRCGROUP	GROUP ID
60	(3C)	CHARACTER	1	SRCHIPER	DLF RESOURCE (Y|N)
61	(3D)	CHARACTER	1	SRCUSEDFOR	USED FOR (N|P|C|B)
62	(3E)	CHARACTER	2	SRCONERROR	ON ERROR OPTION
64	(40)	SIGNED	4	SRCIVLNUM	NUMBER OF INTERVALS
68	(44)	CHARACTER	46	SRCDESC	DESCRIPTION
114	(72)	CHARACTER	1	SRCONCOMPL	ON COMPLETE (Y|N|R|blank)
115	(73)	CHARACTER	1	SRCMAXTYPE	MAX LIMIT TYPE (Y|N|R)
116	(74)	SIGNED	4	SRCMAXLIMIT	MAX LIMIT VALUE
120	(78)	CHARACTER	12	*	RESERVED
132	(84)	SIGNED	4	SRCDEFQUANT	DEFAULT QUANTITY
136	(88)	CHARACTER	1	SRCDEFAVAIL	DEFAULT AVAILABILITY
137	(89)	CHARACTER	11	*	RESERVED
148	(94)	CHARACTER	8	SRCLUSER	LAST UPDATING USER
156	(9C)	CHARACTER	6	SRCLDATE	DATE OF LAST UPDATE
162	(A2)	CHARACTER	4	SRCLTIME	TIME OF LAST UPDATE
166	(A6)	CHARACTER	2	*	RESERVED
168	(A8)	CHARACTER	8	SRCLUTS	TOD CLOCK AT LAST UPDATE
176	(B0)	SIGNED	1	SRCVER	RECORD VERSION
177	(B1)	CHARACTER	15	*	RESERVED

Notes:

1. Day number must be from 1 to 7 for Monday to Sunday or 8 for standard.
2. For a correct interpretation of the fields described as "Tod clock at last update", see TOD fields on page 370.

Table 234. SRIVL Segment

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	CHARACTER	32	SRIVL	INTERVAL
0	(0)	CHARACTER	32	SRIVLCOM	COMMON DATA
0	(0)	SIGNED	4	SRIVLDAY	DAY NUMBER
4	(4)	CHARACTER	6	SRIVLDATE	SPECIFIC DATE
10	(A)	CHARACTER	2	*	RESERVED
12	(C)	CHARACTER	4	SRIVLFTIME	FROM TIME
16	(10)	CHARACTER	4	SRIVLTTIME	TO TIME
20	(14)	SIGNED	4	SRIVLQUANT	MAX NUMBER OF SRs TO ALLOCATE
24	(18)	SIGNED	4	SRIVLWSCNUM	NUMBER OF CONNECTED WSs
28	(1C)	CHARACTER	1	SRIVLAVAIL	AVAILABLE (Y|N)
29	(1D)	CHARACTER	3	RESERVED	RESERVED

Table 235. SRIWS Segment

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	CHARACTER	8	SRIWS	CONNECTED WS SEGMENT
0	(0)	CHARACTER	4	SRIWSNAME	WORKSTATION NAME
4	(4)	CHARACTER	4	*	RESERVED

Table 236. SRDWS Segment

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	CHARACTER	8	SRDWS	CONNECTED WS SEGMENT
0	(0)	CHARACTER	4	SRDWSNAME	WORKSTATION NAME

Table 236. SRDWS Segment (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
4	(4)	CHARACTER	4	*	RESERVED

Workstation description (resource codes WS, WSCOM)

The workstation description record can contain these segments:

WSCOM

Common segment. One, and only one, common segment must appear as the first segment in each record.

WSDEST

Workstation destination segment.

WSIVL

Workstation open interval segment.

WSSD

Workstation specific date segment.

WSWD

Workstation weekday segment.

WSAM

Workstation access method segment.

Note: For a correct interpretation of the fields described as "Tod clock at last update", see TOD fields on page 370.

WSCOM - Common segment

Common description of a workstation.

Workstation types:

G

General

C

Computer

P

Printer

R

Remote engine

Reporting attribute:

A

Automatic reporting

S

Manual reporting start and stop

C

Manual reporting, completion only

N

Nonreporting

Table 237. WSCOM Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	128	WSCOM	
0	(0)	CHARACTER	4	WSKEY	UNIQUE IDENTIFIER
0	(0)	CHARACTER	4	WSNAME	WORKSTATION NAME
4	(4)	UNSIGNED	1	WSVERS	VERSION OF RECORD=1
5	(5)	CHARACTER	1	WSTYPE	WORKSTATION TYPE (G|C|P|R)
6	(6)	CHARACTER	1	WSREP	REPORTING ATTRIBUTE A|S|C|N
7	(7)	CHARACTER	1	WSPREP	JOBSETUP ABILITY
8	(8)	SIGNED	4	WSTRSPT	TRANSPORT TIME FROM PREDECESSOR WS
12	(C)	SIGNED	4	WSOPDUR	DEFAULT OPERATION DURATION
16	(10)	SIGNED	4	WSDAY#	TOTAL NUMBER OF DAYS
20	(14)	SIGNED	4	WSTOTIVL#	NUMBER OF OPEN INTERVALS
24	(18)	CHARACTER	8	WSROUT	PRINTOUT ROUTING FOR DP
32	(20)	CHARACTER	32	WSDESC	WORKSTATION DESCRIPTION
64	(40)	CHARACTER	1	WSPSJT	CONTROL ON SERVERS
65	(41)	CHARACTER	1	WSSPLIT	SPLITTABLE ATTRIBUTE
66	(42)	CHARACTER	2	WSR1NAM	WS RESOURCE NAME

Table 237. WSCOM Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
68	(44)	CHARACTER	1	WSR1PLAN	RESOURCE USED AT PLANNING
69	(45)	CHARACTER	1	WSR1CONT	RESOURCE USED FOR CONTROL
70	(46)	CHARACTER	2	WSR2NAM	WS RESOURCE NAME
72	(48)	CHARACTER	1	WSR2PLAN	RESOURCE USED AT PLANNING
73	(49)	CHARACTER	1	WSR2CONT	RESOURCE USED FOR CONTROL
74	(4A)	CHARACTER	8	WSSUDS	DESTINATION
82	(52)	CHARACTER	6	WSLDATE	DATE LAST UPDATED
88	(58)	CHARACTER	4	WSLTIME	TIME LAST UPDATED
92	(5C)	CHARACTER	8	WSLUSER	USERID OF LAST UPDATER
100	(64)	CHARACTER	1	WSSTC	STARTED TASK Y|N
101	(65)	CHARACTER	1	WSWTO	WTO ABILITY Y|N
102	(66)	CHARACTER	1	WSPSPL	PLANNING ON SERVERS Y|N
103	(67)	CHARACTER	1	WSAUTO	SYSTEM AUTOMATION WORKSTATION
104	(68)	CHARACTER	8	WSLUTS	TOD CLOCK AT LAST UPDATE
112	(70)	SIGNED	4	WSOPDURI	DEFAULT OP. DURATION, IN 100th OF SECOND
116	(74)	CHARACTER	1	WSTWS	NOT USED
117	(75)	CHARACTER	1	WSWAIT	WAIT WORKSTATION (Y|N)
118	(76)	CHARACTER	1	WSVIRT	VIRTUAL WORKSTATION (Y|N)
119	(77)	CHARACTER	1	WSZCENTR	Z-CENTRIC WORKSTATION (Y|N)
120	(78)	SIGNED	4	WSDES#	NUMBER OF DESTINATIONS
124	(7C)	CHARACTER	1	WSRENG	REMOTE ENGINE TYPE: Z, D OR BLANK
125	(7D)	CHARACTER	1	WSDYN	DYNAMIC SCHEDULING (Y|N)
126	(7E)	CHARACTER	2	*	RESERVED

WSDEST – Destination segment

Workstation description: a virtual workstation destination name.

For each destination segment, the database contains a virtual workstation destination description record. WSVCOM is the corresponding segment.

Table 238. WSDEST Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	16	WSDEST	WORK STATION DESTINATION
0	(0)	CHARACTER	8	WSDVDEST	WORK STATION DESTINATION NAME
8	(8)	CHARACTER	8	*	FREE

WSIVL - Open interval segment

Workstation description: an open interval.

Table 239. WSIVL Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	32	WSIVL	
0	(0)	CHARACTER	4	WSIVLS	START TIME OF INTERVAL
4	(4)	CHARACTER	4	WSIVLE	END TIME OF INTERVAL
8	(8)	SIGNED	4	WSIVLPS#	NUMBER OF PARALLEL SERVERS
12	(C)	SIGNED	4	WSIVLR1#	R1 CAPACITY
16	(10)	SIGNED	4	WSIVLR2#	R2 CAPACITY
20	(14)	CHARACTER	4	WSIVLAWS	ALTERNATE WORKSTATION NAME
24	(18)	CHARACTER	8	*	RESERVED

WSSD - Specific date segment

Workstation description: a specific date.

Table 240. WSSD Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	48	WSSD	
0	(0)	CHARACTER	6	WSSDDATE	SPECIFIC DATE

Table 240. WSSD Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
6	(6)	CHARACTER	2	*	RESERVED
8	(8)	CHARACTER	24	WSSDDESC	DESCRIPTION OF THE DATE
32	(20)	SIGNED	4	WSSDIVL#	NUMBER OF OPEN INTERVALS
36	(24)	CHARACTER	12	*	RESERVED

WSWD - Weekday segment

Workstation description: a weekday.

Weekday can be:

MONDAY
TUESDAY
WEDNESDAY
THURSDAY
FRIDAY
SATURDAY
SUNDAY
STANDARD

Note: WEDNESDAY is actually stored as WEDNESDA.

Table 241. WSWD Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	48	WSWD	
0	(0)	CHARACTER	8	WSWDDAY	WEEK DAY
8	(8)	CHARACTER	24	WSWDDESC	DESCRIPTION OF THE DAY
32	(20)	SIGNED	4	WSWDDIVL#	NUMBER OF OPEN INTERVALS
36	(24)	CHARACTER	12	*	RESERVED

WSAM - Workstation access method segment

Workstation access method.

Note: The Workstation access method segment is no longer supported. If you specify this segment in a workstation description record, the segment is ignored.

Table 242. WSAM Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	80	WSAM	
0	(0)	CHARACTER	12	WSAMACC	ACCESS METHOD NAME
12	(C)	CHARACTER	52	WSAMADDR	NODE ADDRESS
64	(40)	SIGNED	4	WSAMPORT	PORT NUMBER
68	(44)	CHARACTER	12	*	RESERVED

WSOPT - workstation description record segment

Workstation description record.

Table 243. WSOPT Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE		WSOPT	Workstation options
0	(0)	CHARACTER	47	WSOPTJOBUSR	Default JOBUSER
47	(2F)	CHARACTER	1	WSOPTJOBPWD	Default JOBPWD
48	(2E)	CHARACTER	40	WSOPTJOBTYPE	Default JOBTYPE
88	(58)	CHARACTER	1	WSOPTBROKER	The workstation is a BROKER workstation
89	(59)	CHARACTER	40	WSOPTPOOL	Pool
129	(81)	CHARACTER	40	WSOPTDYNPOOL	Dynamic pool
169	(44)	CHARACTER	8		Reserved

Note: The creation of dynamic agents, pools and dynamic pools is not supported using PIF. To perform these operations, use the Dynamic Workload Console. To install dynamic agents, run the related installation program.

Virtual workstation destination description (resource codes WSV, WSVCOM)

The virtual workstation destination description record can contain these segments:

WSVCOM

Common segment. One, and only one, common segment must appear as the first segment in each record.

WSVIVL

Virtual workstation destination open interval segment.

WSVSD

Virtual workstation destination specific date segment.

WSVWD

Virtual workstation destination weekday segment.

Note:

1. For REPLACE request: you can only update fields marked by (R). Other fields are either the identifier, set implicitly, or cannot be changed.
2. For a correct interpretation of the fields described as "Tod clock at last update", see TOD fields on page 370.

WSVCOM - Common segment

Common description of a virtual workstation destination.

Workstation types:

C

Computer

Reporting attribute:

A

Automatic reporting

Table 244. WSVCOM Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	96	WSVCOM	
0	(0)	CHARACTER	12	WSVKEY	UNIQUE IDENTIFIER
0	(0)	CHARACTER	4	WSVNAME	WORKSTATION NAME
4	(4)	CHARACTER	8	WSVDESTN	WORKSTATION DESTINATION
12	(C)	UNSIGNED	1	WSVVERS	VERSION OF RECORD=1
13	(D)	CHARACTER	1	*	WORKSTATION TYPE (NOT USED)
14	(E)	CHARACTER	1	*	REPORTING ATTRIBUTE (NOT USED)
15	(F)	CHARACTER	1	*	JOBSETUP ABILITY (NOT USED)
16	(10)	SIGNED	4	WSVDAY#	TOTAL NUMBER OF DAYS
20	(14)	SIGNED	4	WSVTOTIVL#	NUMBER OF OPEN INTERVALS
24	(18)	CHARACTER	8	*	PRINTOUT ROUTING FOR DP (NOT USED)
32	(20)	CHARACTER	1	WSVPSJT	CONTROL ON SERVERS (R)
33	(21)	CHARACTER	1	*	SPLITTABLE ATTRIBUTE (NOT USED)
34	(22)	CHARACTER	2	WSVR1NAM	WS RESOURCE NAME (R)
36	(24)	CHARACTER	1	WSVR1PLAN	RESOURCE USED AT PLANNING (NOT USED)
37	(25)	CHARACTER	1	WSVR1CONT	RESOURCE USED FOR CONTROL (R)
38	(26)	CHARACTER	2	WSVR2NAM	WS RESOURCE NAME (R)
40	(28)	CHARACTER	1	WSVR2PLAN	RESOURCE USED AT PLANNING (NOT USED)
41	(29)	CHARACTER	1	WSVR2CONT	RESOURCE USED FOR CONTROL (R)
42	(2A)	CHARACTER	8	*	DESTINATION (NOT USED)
50	(32)	CHARACTER	6	WSVLDATE	DATE LAST UPDATED
56	(38)	CHARACTER	4	WSVLTIME	TIME LAST UPDATED
60	(3C)	CHARACTER	8	WSVLUSER	USERID OF LAST UPDATER
68	(44)	CHARACTER	1	*	STARTED TASK Y|N (NOT USED)
69	(45)	CHARACTER	1	*	WTO ABILITY Y|N (NOT USED)
70	(46)	CHARACTER	1	*	PLANNING ON SERVERS Y|N (NOT USED)

Table 244. WSVCOM Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
71	(47)	CHARACTER	1	*	SYSTEM AUTOMATION WORKSTATION (NOT USED)
72	(48)	CHARACTER	8	WSVLUTS	TOD CLOCK AT LAST UPDATE
80	(50)	SIGNED	4	*	DEFAULT OP. DURATION, IN 100th OF SECOND (NOT USED)
84	(54)	CHARACTER	1	*	NOT USED
85	(55)	CHARACTER	1	*	WAIT WORKSTATION (Y|N) (NOT USED)
86	(56)	CHARACTER	10	*	RESERVED

WSVIVL - Open interval segment

Workstation description: an open interval.

Table 245. WSVIVL Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	32	WSVIVL	
0	(0)	CHARACTER	4	WSVIVLS	START TIME OF INTERVAL
4	(4)	CHARACTER	4	WSVIVLE	END TIME OF INTERVAL
8	(8)	SIGNED	4	WSVIVLPS#	NUMBER OF PARALLEL SERVERS
12	(C)	SIGNED	4	WSVIVLR1#	R1 CAPACITY
16	(10)	SIGNED	4	WSVIVLR2#	R2 CAPACITY
20	(14)	CHARACTER	4	*	RESERVED
24	(18)	CHARACTER	8	*	RESERVED

WSVSD - Specific date segment

Workstation description: a specific date.

Table 246. WSVSD Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	48	WSVSDD	
0	(0)	CHARACTER	6	WSVSDDDATE	SPECIFIC DATE
6	(6)	CHARACTER	2	*	RESERVED
8	(8)	CHARACTER	24	WSVSDDDESC	DESCRIPTION OF THE DATE
32	(20)	SIGNED	4	WSVSDDIVL#	NUMBER OF OPEN INTERVALS
36	(24)	CHARACTER	12	*	RESERVED

WSVWD - Weekday segment

Workstation description: a weekday.

Weekday can be:

MONDAY
TUESDAY
WEDNESDAY
THURSDAY
FRIDAY
SATURDAY
SUNDAY
STANDARD

Note: WEDNESDAY is actually stored as WEDNESDA.

Table 247. WSWD Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	48	WSVWDD	
0	(0)	CHARACTER	8	WSVWDDDAY	WEEK DAY
8	(8)	CHARACTER	24	WSVWDDDESC	DESCRIPTION OF THE DAY
32	(20)	SIGNED	4	WSVWDDDIVL#	NUMBER OF OPEN INTERVALS
36	(24)	CHARACTER	12	*	RESERVED

Appendix B. API object fields

This appendix describes the field names of each API object. It also identifies the fields that you can specify in APPSEL and APPFLD sections of a buffer.

Fields in the IBM Z Workload Scheduler API data dictionary are defined in one of these formats:

BIN

A binary value.

CHAR

A character value.

DATE

A character value, in the format YYMMDD.

TIME

A character value, in the format HHMM.

DUR

A character value, in the format HHMM or HHHHMM depending on the field length.

FLAG

A character value, in the format Y or N.

Each APPSEL and APPFLD column has one of these values:

R

Required. You must specify this field and the operator value must be EQ or =. For a GET request with a key type of OWNER, PRED, or SUCC, you must specify these fields and the operator must be EQ to ensure that there is only one possible match. When the key type is SAME, these fields are optional.

For PUT and DEL requests you must specify these fields. Also, the key type must be SAME and the operator EQ.

O

Optional.

N

Not supported.

Current plan status object

This option is valid for the current plan status object:

- GET request with key type SAME.

The default key type is SAME.

Table 248. CP_STATUS Object Fields

Field	Type	Len	Description	APPSEL	APPFLD
CP_CREATE_DATE	DATE	6	Current plan creation date	O	O
CP_CREATE_TIME	TIME	4	Current plan creation time	O	O
CP_END_DATE	DATE	6	Current plan end date	O	O
CP_END_TIME	TIME	4	Current plan end time	O	O
BACKUP_DATE	DATE	6	Last backup date	O	O
BACKUP_TIME	TIME	4	Last backup time	O	O
FIRST_EV_DATE	DATE	6	First event after backup date	O	O
FIRST_EV_TIME	TIME	4	First event after backup time	O	O
FIRST_EV_D_TS	CHAR	8	First event after backup date. The field format is 00YYDDDF for dates in the 20th century, and 01YYDDDF for dates in the 21st century.	O	O
FIRST_EV_T_TS	CHAR	8	First event after backup time in format HHMMSSTH	O	O
TURNOVER_NCP	CHAR	1	Turnover in progress, Y or N	O	O
CP_EXIST	CHAR	1	Current plan exists, Y or N	O	O
CP_DDNAME	CHAR	8	Current plan ddname	O	O
JT_DDNAME	CHAR	8	Job-tracking ddname	O	O
JCL_REP_DDNAME	CHAR	8	JCL repository ddname	O	O
NUM_PIF_ADDS	BIN	4	Number of occs added by PIF	O	O
NUM_MCP_ADDS	BIN	4	Number of occs added by MCP	O	O
NUM_ETT_ADDS	BIN	4	Number of occs added by ETT	O	O
NUM_AR_ADDS	BIN	4	Number of occs added by autorec	O	O
NUM_OCCS	BIN	4	Number of occurrences	O	O
NUM_OPERS	BIN	4	Number of operations	O	O

Current plan operation object

These options are valid for the current plan operation object:

- GET request with key type SAME, PRED, or SUCC
- PUT request with key type SAME
- DEL request with key type SAME.

The default key type is SAME.

Table 249. CP_OPERATION Object Fields

Field	Type	Len	Description	APPSEL	APPFLD	
					Get	Put
OPER_NUM	BIN	2	Operation number	R	O	N
AUTHORITY_GROUP	CHAR	8	Authority group	O	O	N
CATMGMT_STATUS	CHAR	1	CleanUp status: **<blank>** None **C** Completed **E** Ended in error **I** Initiated **O** OPInfo is available **R** OpInfo requested **S** Started **W** Waiting for OPInfo	O	O	N
APPL_ID	CHAR	16	Application ID	R	O	N
APPL_IA_DATE	DATE	6	Application input arrival date	R	O	N
APPL_IA_TIME	TIME	4	Application input arrival time	R	O	N

Table 249. CP_OPERATION Object Fields (continued)

Field	Type	Len	Description	APPSEL	APPFLD	
					Get	Put
OPER_TEXT	CHAR	24	Descriptive text for the operation	O	O	N
JOBNAME	CHAR	8	Job name	O	O	N
WS_NAME	CHAR	4	Workstation name	O	O	N
CLASS	CHAR	1	Job class or SYSOUT class value	O	O	O
IA_DEFAULTED	FLAG	1	Operation input arrival defaulted	N	O	N
IMM_CATMGMT_DEF	FLAG	1	Immediate Clean Up is defined	N	O	N
DEFR_CATMGMT_DEF	FLAG	1	Manual or automatic Clean up is defined	N	O	N
MANUALLY_HELD	FLAG	1	Manually held operation	N	O	O
NOP_OPER	FLAG	1	NOP operation	N	O	O
EXECUTE_OPER	FLAG	1	Execute requested for operation	N	O	O
WAIT_MAN_CATMGMT	FLAG	1	Always N	N	O	N
FORM_NUMBER	CHAR	8	Form number	O	O	O
PLAN_START_DATE	DATE	6	Planned start date	O	O	N
PLAN_START_TIME	TIME	6	Planned start time	O	O	N
PLAN_END_DATE	DATE	6	Planned end date	O	O	N
PLAN_END_TIME	TIME	4	Planned end time	O	O	N
OPER_IA_DATE	DATE	6	Operation input arrival date	O	O	N
OPER_IA_TIME	TIME	4	Operation input arrival time	O	O	N
DL_DATE	DATE	6	Operation deadline date	O	O	N
DL_TIME	TIME	4	Operation deadline time	O	O	N
LATEST_OUT_DATE	DATE	6	Operation latest out date	O	O	N
LATEST_OUT_TIME	TIME	4	Operation latest out time	O	O	N

Table 249. CP_OPERATION Object Fields (continued)

Field	Type	Len	Description	APPSEL	APPFLD	
					Get	Put
ACT_START_DATE	DATE	6	Actual start date	O	O	N
ACT_START_TIME	TIME	4	Actual start time	O	O	N
ACT_ARRIVAL_DATE	DATE	6	Actual arrival date	O	O	N
ACT_ARRIVAL_TIME	TIME	4	Actual arrival time	O	O	N
INTER_START_DATE	DATE	6	Intermediate start date	O	O	N
INTER_START_TIME	TIME	4	Intermediate start time	O	O	N
ACT_END_DATE	DATE	6	Actual end date	O	O	N
ACT_END_TIME	TIME	4	Actual end time	O	O	N
EST_DUR	DUR	4	Estimated duration	O	O	O
ACT_DUR	DUR	6	Actual duration	O	O	N
NUM_PAR_SERV_REQ	BIN	2	Number of parallel servers required	O	O	N
NUM_WS_R1_REQ	BIN	2	Number of R1 resources required	O	O	N
NUM_WS_R2_REQ	BIN	2	Number of R2 resources required	O	O	N
CURRENT_STATUS	CHAR	1	Current status of the operation: **A** Arriving **C** Completed **D** Deleted **E** Ended in error **I** Interrupted	O	O	O

Table 249. CP_OPERATION Object Fields (continued)

Field	Type	Len	Description	APPSEL	APPFLD	
					Get	Put
			R Ready, all preds complete **S** Started **U** Undecided **W** Waiting, uncompleted preds ***** Ready, nonreporting pred			
ERROR_CODE	CHAR	4	Error code	O	O	O
AUTO_ERROR_COMPL	CHAR	1	Auto error completion Y or N	O	O	N
PRIORITY	CHAR	1	Priority 1 to 9	O	O	N
EXTENDED_STATUS	CHAR	1	Extended status: **A** Waiting for deferred CM **C** Waiting for CM to complete **E** Error during job submission	O	O	N

Table 249. CP_OPERATION Object Fields (continued)

Field	Type	Len	Description	APPSEL	APPFLD	
					Get	Put
			G Started on WAIT workstation **H** Manually held **L** Time operation is late **M** Status set manually **N** NOP operation **Q** Job added to JES queue **R** Automatic error reset **S** Job or started task executing **T** Waiting for time **U** Submit in progress			

Table 249. CP_OPERATION Object Fields (continued)

Field	Type	Len	Description	APPSEL	APPFLD	
					Get	Put
			X Waiting for special resource			
NUM_SUCC	BIN	2	Number of successors	O	O	N
NUM_PRED	BIN	2	Number of predecessors	O	O	N
NUM_DEPENDENCIES	BIN	2	Number of successors and predecessors	O	O	N
NUM_COMPL_PRED	BIN	2	Number of predecessors completed	O	O	N
NUM_SR	BIN	2	Number of special resources	O	O	N
RERUN_RECORD	FLAG	1	Rerun record for this operation	N	O	N
VALID_EXIT_PASS	FLAG	1	Validation exit passed	N	O	N
ASSUMED_COMPLETE	FLAG	1	Assumed completed	N	O	N
SPECIFY_IA	FLAG	1	Specified input arrival for op	N	O	N
SPECIFY_DL	FLAG	1	Specified deadline for op	N	O	N
AUTO_SUBMISSION	FLAG	1	Auto submission of job	N	O	N
AUTO_HOLD_REL	FLAG	1	Automatic hold/release	N	O	N
LATE_MSG_ISSUED	FLAG	1	Late operator message issued	N	O	N
JOB_SUBMITTED	FLAG	1	Job submitted	N	O	N
TIME_JOB	FLAG	1	Time job	N	O	N
PREP_WS_NOTCOMPL	FLAG	1	Prep op exists but is not complete	N	O	N
SUPPRESS_IF_LATE	FLAG	1	Suppress if late	N	O	N
HIGH_RC_USED	FLAG	1	High return code used	N	O	N
PENDING_PRED	FLAG	1	Pending predecessor	N	O	N

Table 249. CP_OPERATION Object Fields (continued)

Field	Type	Len	Description	APPSEL	APPFLD	
					Get	Put
LONG_DUR_ISSUED	FLAG	1	Long duration message issued	N	O	N
LAST_MCP_UP_DATE	BIN	4	Date of last MCP update	N	O	N
LAST_MCP_UP_TIME	BIN	4	Time of last MCP update	N	O	N
DEPENDENCY_TYPE	CHAR	1	Dependency type: **P** Predecessor **S** Successor	N	O	N
RESTARTABLE	FLAG	1	Restartable operation	N	O	N
INSTPARM_RESTART	FLAG	1	Installation default for workload restart	N	O	N
REROUTABLE	FLAG	1	Reroutable operation	N	O	N
INSTPARM_REROUTE	FLAG	1	Installation default for workload reroute	N	O	N
REROUTED	FLAG	1	Operation rerouted	N	O	N
DL_WTO_WANTED	FLAG	1	Deadline WTO required	N	O	N
DL_WTO_REQ_SENT	FLAG	1	Deadline WTO request sent	N	O	N
DL_WTO_REQ_PROC	FLAG	1	Deadline WTO request processed	N	O	N
HIGHRC_NOT_ERROR	BIN	2	Highest return code not in error	O	O	N
ALT_WS_NAME	CHAR	4	Alternate workstation name	O	O	N
USER_FIELD	CHAR	16	User field	O	O	N
ON_CRITICAL_PATH	CHAR	1	Critical path indicator, Y, N, or F	O	O	N
LATEST_OUT_PASS	CHAR	1	Latest out passed, Y or N	O	O	N
URGENT	CHAR	1	Urgent, Y or N	O	O	N
TRANSPORT_TIME	BIN	4	Transport time HHMM	O	O	N

Table 249. CP_OPERATION Object Fields (continued)

Field	Type	Len	Description	APPSEL	APPFLD	
					Get	**Put**
APPL_TEXT	CHAR	24	Application text	O	O	N
APPL_OWNER_ID	CHAR	16	Application owner ID	O	O	N
JOB_ID	CHAR	8	JES job number	O	O	N
SMF_READER_DATE	BIN	4	SMF reader date	O	O	N
SMF_READER_TIME	BIN	4	SMF reader time	O	O	N
JOB_STATUS	CHAR	1	Job status H, Q, N, or blank	O	O	N
JCL_PREPARATION	CHAR	1	JCL preparation operation, Y or N	O	O	N
OI_EXIST	CHAR	1	Op instruction exists Y, N, or +	O	O	N
RESOURCE_USE	CHAR	1	Blank in OPC/ESA Release 3	O	O	N
EXTENDED_STATUS2	CHAR	1	Additional status explanation: **A** Automatic error reset **C** Workstation closed **D** Job submission deactivated **F** Job submission failed **H** Workstation close in progress	O	O	N

Table 249. CP_OPERATION Object Fields (continued)

Field	Type	Len	Description	APPSEL	APPFLD	
					Get	Put
			J No auto job submission **L** Time job is late **P** All parallel servers in use **S** Resource unavailable **T** Start time not reached **U** Work station is unlinked **1** Insufficient WS resource 1 **2** Insufficient WS resource 2			
WS_TYPE	CHAR	1	Workstation type: **1** General **2** Computer **3** Print	O	O	N

Table 249. CP_OPERATION Object Fields (continued)

Field	Type	Len	Description	APPSEL	APPFLD	
					Get	Put
WTO_WS	CHAR	1	WTO workstation type, Y or N	O	O	N
OCC_GROUP_DEF	CHAR	16	Occurrence group name	O	O	N

Current plan special resource object

This option is valid for the current plan special resource object:

- GET request with key type OWNER.

The default key type is OWNER.

Table 250. CP_RESOURCE Object Fields

Field	Type	Len	Description	APPSEL	APPFLD
SR_NAME	CHAR	44	Special resource name	N	O
ALLOCATION_TYPE	CHAR	1	Allocation type (S or X)	N	O
AVAILABLE	FLAG	1	Availability indicator	N	O
SHR_IN_USE	FLAG	1	Special resource allocated - SHR	N	O
IN_USE_EXCLUSIVE	FLAG	1	Special resource allocated - EXCL	N	O
KEPT_AT_ERROR	FLAG	1	Special resource has been kept on error	N	O
KEPT_EXCLUSIVE	FLAG	1	EXCL special resource kept on error	N	O
QUANTITY	BIN	31	Quantity requested by the operation	N	O
KEEP_ON_ERROR	CHAR	1	On-error indicator (Y, N, blank)	N	O

Note:

1. Because you must identify the owning operation to retrieve special resource information, you must specify the selection fields that are mandatory for the CP_OPERATION object. No CP_OPERATION fields are returned in the receive buffer.
2. The values returned for fields ALLOCATION_TYPE, QUANTITY, and KEEP_ON_ERROR depend on the operation that is used to access the CP_RESOURCE object.

Current plan workstation object

This option is valid for the current plan workstation object:

- GET request with key type SAME.

The default key type is SAME.

Table 251. CP_WORK_STATION Object Fields

Field	Type	Len	Description	APPSEL	APPFLD
WS_NAME	CHAR	4	Workstation name	O	O
WS_TEXT	CHAR	32	Workstation description	O	O
NUM_COMPL	BIN	4	Number of completed operations	O	O
EST_DUR_COMPL	BIN	4	Estimated duration of completed operations	O	O
ACT_DUR_COMPL	BIN	4	Actual duration of completed operations	O	O
NUM_INTER	BIN	4	Number of interrupted operations	O	O
EST_DUR_INTER	BIN	4	Estimated duration of interrupted operations	O	O
ACT_DUR_INTER	BIN	4	Actual duration of interrupted operations	O	O
NUM_START	BIN	4	Number of started operations	O	O
EST_DUR_START	BIN	4	Estimated duration of started operations	O	O
NUM_READY	BIN	4	Number of ready operations	O	O
EST_DUR_READY	BIN	4	Estimated duration of ready operations	O	O
NUM_WAITING	BIN	4	Number of waiting operations	O	O
EST_DUR_WAITING	BIN	4	Estimated duration of waiting operations	O	O
NUM_ARRIVING	BIN	4	Number of arriving operations	O	O

Table 251. CP_WORK_STATION Object Fields (continued)

Field	Type	Len	Description	APPSEL	APPFLD
NUM_NONREP_READY	BIN	4	Number of nonreporting ready operations	O	O
NUM_UNDECIDED	BIN	4	Number of undecided operations	O	O
NUM_ERROR	BIN	4	Number of error operations	O	O
NUM_LATE	BIN	4	Number of late operations	O	O
WS_TYPE	CHAR	1	Workstation type: **1** General **2** Computer **3** Print	O	O
REPORTING_ATTR	CHAR	1	Reporting attribute: **1** Automatic **2** Manual, start and complete **3** Manual, completion only **4** Nonreporting	O	O
R1_NAME	CHAR	2	R1 resource name	O	O
NUM_R1_IN_USE	BIN	2	Number of R1 resources in use	O	O
R1_USED_AT_CNTL	FLAG	1	R1 resource used at control	N	O
R2_NAME	CHAR	2	R2 resource name	O	O
NUM_R2_IN_USE	BIN	2	Number of R2 resources in use	O	O
R2_USED_AT_CNTL	FLAG	1	R2 resource used at control	N	O
READY_LIST_TYPE	CHAR	1	Ready list type	O	O

Table 251. CP_WORK_STATION Object Fields (continued)

Field	Type	Len	Description	APPSEL	APPFLD
JOB_SETUP_ABIL	FLAG	1	Job setup ability	N	O
IVL_NOT_USED	FLAG	1	Interval not used at all	N	O
NO_PAR_SERV	FLAG	1	Parallel servers used for control	N	O
STARTED_TASK_SUP	FLAG	1	Started task support	N	O
WTO_DL_SUP	FLAG	1	WTO workstation	N	O
PENDING_OFFLINE	FLAG	1	Workstation is pending offline	N	O
T_EVENT_PENDING	FLAG	1	T-event pending	N	O
ALT_WS_VARIED	FLAG	1	Varied alternate workstation set	N	O
PREV_EVENT_DATE	BIN	4	Previous event date	O	O
PREV_EVENT_TIME	BIN	4	Previous event time	O	O
NUM_IVL	BIN	2	Number of open intervals	O	O
MAX_NUM_EVENTS	BIN	2	Max number of events in 15 minutes	O	O
WS_STATUS	CHAR	1	Workstation status: **A** Active **O** Offline **F** Failed **U** Unknown	O	O
DEF_TRANS_TIME	BIN	2	Transport time default	O	O
OFFLINE_DATE	BIN	4	Offline date	O	O
OFFLINE_TIME	BIN	4	Offline time	O	O
CURRENT_ALT_WS	CHAR	4	Alternate workstation name	O	O

Current plan open interval object

This option is valid for the current plan open interval object:

- GET request with key type OWNER.

The default key type is OWNER.

Note: Because you must identify the owning workstation to retrieve open interval information, you must specify selection fields from the CP_WORK_STATION object. No CP_WORK_STATION fields are returned in the receive buffer.

Table 252. CP_OPEN_INTERVAL Object Fields

Field	Type	Len	Description	APPSEL	APPFLD
START_DATE	DATE	6	Start date	N	O
START_TIME	TIME	4	Start time	N	O
END_DATE	DATE	6	End date	N	O
END_TIME	TIME	4	End time	N	O
MAX_PAR_SERV	BIN	2	Maximum parallel servers	N	O
MAX_PAR_SERV_DP	BIN	2	Maximum parallel servers set at daily planning	N	O
SET_BY_MCP	FLAG	1	Interval created by MCP	N	O
SET_BY_DP	FLAG	1	Interval created by DP	N	O
CURR_R1_CAP	BIN	2	Current R1 resource capacity	N	O
R1_CAP_SET_BY_DP	BIN	2	R1 resource capacity set at DP	N	O
CURR_R2_CAP	BIN	2	Current R2 resource capacity	N	O
R2_CAP_SET_BY_DP	BIN	2	R2 resource capacity set at DP	N	O
ALT_WS_NAME	CHAR	4	Alternate workstation name	N	O
ALT_WS_NAME_DP	CHAR	4	Alternate wsname set by DP	N	O

Current plan operation event object

This option is valid for the current plan operation event object:

- CREATE request with key type SAME.

The default key type is SAME.

Table 253. CP_OPER_EVENT Object Fields

Field	Type	Len	Description	APPSEL	APPFLD
SUBSYSTEM_NAME	CHAR	4	Subsystem name	O	N

Table 253. CP_OPER_EVENT Object Fields (continued)

Field	Type	Len	Description	APPSEL	APPFLD
WS_NAME	CHAR	4	Workstation name	O	N
JOBNAME	CHAR	8	Job name	O	N
APPL_ID	CHAR	16	Application ID	O	N
OPER_NUM	BIN	15	Operation number (decimal 1–99)	O	N
APPL_IA_DATE	CHAR	6	Input arrival date	O	N
APPL_IA_TIME	CHAR	4	Input arrival time	O	N
FORM_NUMBER	CHAR	8	Form number for operations at print workstations	O	N
CLASS	CHAR	1	SYSOUT class for operations at print workstations	O	N
OPER_TOKEN	CHAR	8	Operation token	O	N
STATUS	CHAR	1	New status: **C** Complete **E** Ended in error **I** Interrupted **Q** Extended status of a started operation (S) is Q (queued awaiting execution) **S** Started **T** Extended status of a started operation (S) is S (operation is executing) **X** Reset the current status for this operation	N	O

Table 253. CP_OPER_EVENT Object Fields (continued)

Field	Type	Len	Description	APPSEL	APPFLD
ERROR_CODE	CHAR	4	Error code (for new status E)	N	O
ACT_DUR	CHAR	4	Actual duration HHMM (for new status C or E)	N	O
EV_CREATION_DATE	CHAR	4	Event creation date. The field format is 00YYDDDF for dates in the 20th century, and 01YYDDDF for dates in the 21st century. Default is current date	N	O
EV_CREATION_TIME	BIN	31	Event creation time (100 * secs). Default is current time	N	O
JOB_NUMBER	CHAR	5	Job number	N	O

Note:

1. To select an operation, specify at least OPER_TOKEN, or WS_NAME with either JOBNAME or APPL_ID. The remaining values can be initialized to zeros or blanks.

   OPER_TOKEN is a hexadecimal value that uniquely identifies an operation. If you stored the token set in the OPCTOKEN parameter of the operation-initiation exit (EQQUX009), you can provide this token to identify the operation. OPER_TOKEN is valid only for operations at workstations that have a user-defined destination.

2. SUBSYSTEM_NAME is the name of the IBM Z Workload Scheduler subsystem that the event should be reported to. It is used only to select the target for the event and is not stored in the representation of the object.

   If you specify SUBSYSTEM_NAME in APPSEL but do not provide a value in the APPVAL section, or you specify MSTR, the event is broadcast to all IBM Z Workload Scheduler subsystems on the same z/OS image. If you do not specify SUBSYSTEM_NAME, the event is reported to the IBM Z Workload Scheduler subsystem that owns the target LU.

   If your ATP invokes the EQQUSIN subroutine directly, and you do not specify SUBSYSTEM_NAME, the event is broadcast to all IBM Z Workload Scheduler subsystems on the same z/OS image.

3. If you do not provide enough information to uniquely identify the operation, and IBM Z Workload Scheduler finds more than one operation that matches the criteria you specified, IBM Z Workload Scheduler must determine the most applicable operation to update. IBM Z Workload Scheduler selects the operation from operations in status R, A, *, S, I, or E, by investigating these characteristics in the stated order:
   a. The operation has priority 9.
   b. Earliest latest start time.

c. Priority 8-1.
d. Input arrival time specified for the operation or the occurrence input arrival if the operation does not have input arrival specifically defined.

So from the operations that match the selection criteria, the operation with priority 9 is updated. If more than one operation has priority 9, the operation with the earliest latest start time is updated. If latest start is equal, the operation with the highest priority is updated. If priority is equal, the operation with the earliest input arrival time is updated. If input arrival is also equal, the update is performed on a first-in first-out basis.

4. In the APPFLD section, you must specify at least STATUS.
5. JOB_NUMBER is a number that you can provide for the job. It is valid only for operations at general automatic workstations and workstations that have a user-defined destination. Do not specify JOB_NUMBER for operations that are submitted through a tracker.

Current plan OPINFO event object

This option is valid for the current plan OPINFO event object:

- CREATE request with key type SAME.

The default key type is SAME.

Table 254. CP_OPINFO_EVENT Object Fields

Field	Type	Len	Description	APPSEL	APPFLD
SUBSYSTEM_NAME	CHAR	4	Subsystem name	O	N
WS_NAME	CHAR	4	Workstation name	O	N
JOBNAME	CHAR	8	Job name	O	N
APPL_ID	CHAR	16	Application ID	O	N
OPER_NUM	BIN	15	Operation number (decimal 1–99)	O	N
APPL_IA_DATE	CHAR	6	Input arrival date	O	N
APPL_IA_TIME	CHAR	4	Input arrival time	O	N
FORM_NUMBER	CHAR	8	Form number for operations at print workstations	O	N
CLASS	CHAR	1	SYSOUT class for operations at print workstations	O	N
USERDATA	CHAR	16	User data (free form text)	N	O

Note:

1. If the OPINFOSCOPE keyword of the JTOPTS statement is IP, which is the default, you must specify WS_NAME for IBM Z Workload Scheduler to identify the operation. If OPINFOSCOPE keyword is set to ALL, you must specify JOBNAME or APPL_ID. The remaining values can be initialized to zeros or blanks.
2. SUBSYSTEM_NAME. See the explanation of this field 2 on page 483.
3. If you do not provide enough information to uniquely identify the operation, and IBM Z Workload Scheduler finds more than one operation that matches the criteria you specified, IBM Z Workload Scheduler must determine the most applicable operation to update. IBM Z Workload Scheduler considers operations in status R, A, *, S, I, or E when selecting the operation. IBM Z Workload Scheduler selects the operation to update by investigating these characteristics in the stated order:
    a. The operation has priority 9.
    b. Earliest latest start time.
    c. Priority 8-1.
    d. Input arrival time specified for the operation or the occurrence input arrival if the operation does not have input arrival specifically defined.
    e. Longest in Ready status.

    So from the operations that match the selection criteria, the operation with priority 9 is updated. If more than one operation has priority 9, the operation with the earliest latest start time is updated. If latest start is equal, the operation with the highest priority is updated. If priority is equal, the operation with the earliest input arrival time is updated.

    If no match has been found, IBM Z Workload Scheduler uses the value of the OPINFOSCOPE keyword of JTOPTS to determine if operations in status C and W are also considered. OPINFOSCOPE can have the value IP (in progress) or ALL. Operations in status C and W are considered only if the value is ALL. The operation with the earliest latest-start-time is selected.

Current plan special resource event object

This option is valid for the current plan special resource event object:

- CREATE request with key type SAME.

The default key type is SAME.

Table 255. CP_SR_EVENT Object Fields

Field	Type	Len	Description	APPSEL	APPFLD
SUBSYSTEM_NAME	CHAR	4	Subsystem name	O	N
SR_NAME	CHAR	44	Name of special resource	R	N
AVAILABLE	CHAR	1	Resource availability (Y|N|K|R)	N	O
QUANTITY	BIN	31	Number available (1-999999)	N	O

Table 255. CP_SR_EVENT Object Fields (continued)

Field	Type	Len	Description	APPSEL	APPFLD
QUANTITY_OPTION	CHAR	8	Quantity option (KEEP|RESET)	N	O
DEVIATION	BIN	31	Number to deviate (-999999 to 999999)	N	O
DEVIATION_OPTION	CHAR	8	Deviation option (KEEP|RESET)	N	O
CREATE	CHAR	1	Create resource if undefined (Y|N)	N	O

Note:

1. SUBSYSTEM_NAME. See the explanation of this field2 on page 483.
2. AVAILABLE updates the Available field of the special resource, which overrides interval and default values. Specify Y (YES) to make the resource available or N (NO) to make it unavailable. Specify R (RESET) to set the availability status to the planned status in the current plan, or K (KEEP) to leave availability unchanged.
3. QUANTITY and QUANTITY_OPTION fields are mutually exclusive. They update the Quantity field in the special resource, which overrides interval and default values. Use QUANTITY to set a numeric value or QUANTITY_OPTION to specify KEEP or RESET. If you specify both fields, message EQQE056W is written to the Z controller message log and the event is ignored.
4. DEVIATION and DEVIATION_OPTION fields are mutually exclusive. Use DEVIATION to set a numeric value or QUANTITY_OPTION to specify KEEP or RESET. If you specify both fields, message EQQE056W is written to the Z controller message log and the event is ignored. The deviation field in the special resource can contain a positive or negative number, which varies the total amount of the resource. IBM Z Workload Scheduler determines the total amount by adding together the quantity and the deviation. For example, if you specify -2 and the current quantity is 10, the total amount that operations can allocate reduces to 8.
5. CREATE specifies if IBM Z Workload Scheduler should create a resource in the current plan if the resource does not exist. NO indicates that the resource should not be added to the resource definitions of the receiving IBM Z Workload Scheduler subsystem. If the resource is already defined in the receiving subsystem, NO has no effect. You can specify NO if the resource is being used only as a means to generate an event for ETT: the event is generated even if the resource does not exist.

   If YES is specified and the DYNAMICADD keyword of the RESOPTS initialization statement is set to YES or EVENT, a resource definition is created in the receiving IBM Z Workload Scheduler subsystem if the resource is not already defined.
6. When you set the quantity or availability of a resource through the API (or other interfaces such as the SRSTAT TSO command or the MCP dialog), the specified value lasts over interval boundaries, even though the next interval can specify a different value. Specify RESET to restore the planned value.

Current plan backup event object

This option is valid for the current plan backup event object:

- CREATE request with key type SAME.

The default key type is SAME.

Table 256. BACKUP_EVENT Object Fields

Field	Type	Len	Description	APPSEL	APPFLD
SUBSYSTEM_NAME	CHAR	4	Subsystem name	O	N
FILENAME	CHAR	2	Name of data set (CP or JS)	R	N

Note: SUBSYSTEM_NAME. See the explanation of this field 2 on page 483.

Current plan workstation event object

This option is valid for the current plan workstation event object:

- CREATE request with key type SAME.

The default key type is SAME.

Table 257. CP_WS_EVENT Object Fields

Field	Type	Len	Description	APPSEL	APPFLD
SUBSYSTEM_NAME	CHAR	4	Subsystem name	O	N
WS_NAME	CHAR	4	Workstation name	R	N
WS_STATUS	CHAR	1	New workstation status: **A** Active **O** Offline **F** Failed	N	R
STARTED_FAIL_OPT	CHAR	1	For new status O or F: **R** Restart operations automatically on the alternate workstation	N	O

Table 257. CP_WS_EVENT Object Fields (continued)

Field	Type	Len	Description	APPSEL	APPFLD
			L Leave the operations in started status **E** Set all started operations to ended in error		
REROUTE_OPT	CHAR	1	For new status O or F: **Y** Reroute operations to alternate workstation **N** Leave operations at the inactive workstation	N	O
ALT_WS	CHAR	4	For new status O or F, workstation for rerouted operations	N	O

Note:

1. SUBSYSTEM_NAME. See the explanation of this field 2 on page 483.
2. If the value provided for WS_STATUS is equal to the current status, the event is ignored.

Appendix C. Sample library (SEQQSAMP)

The SEQQSAMP library contains samples to help you use IBM Z Workload Scheduler programming interfaces. In most cases, you need only add installation-specific JCL to adapt a member in SEQQSAMP to your requirements. Table 258: SEQQSAMP Library Members for Programming Interfaces and the API on page 489 lists the members in the SEQQSAMP library that apply to programming interfaces, and provides a brief description of each member. The pages that follow describe the members in more detail. A list of all samples provided with IBM Z Workload Scheduler is found in *IBM Z Workload Scheduler: Planning and Installation*.

If you need to change a sample member, copy the source to a separate library; the original sample member is then available for reference. Also, create an SMP/E usermod for each sample member you execute in the production environment. Changes to the sample source code are then flagged for your attention, and subsequent updates can be reflected in the production code as soon as possible.

Table 258. SEQQSAMP Library Members for Programming Interfaces and the API

Member	Brief description
EQQAPISM	ASCII file containing a sample API application
EQQOCWTO	Assembler routine for programmers to communicate with operators
EQQPIFAD	Program-interface PL/I sample that creates a two-operation application in the AD database
EQQPIFAP	Program-interface PL/I sample that resolves nonpromptable JCL variables.
EQQPIFCB	Program-interface assembler samples for various current plan or LTP actions
EQQPIFCL	Program-interface assembler sample that uses the DAYSTAT command to return work or free status for a particular date
EQQPIFDJ	Program-interface assembler sample that deletes JCL for completed occurrences from the JCL repository (JS) data set
EQQPIFJC	Program-interface COBOL sample to manipulate JCL variable tables
EQQPIFJD	Program-interface PL/I sample that can either list or delete records in the JCL repository data set (JS)
EQQPIFJV	Program-interface PL/I sample to manipulate JCL variable tables
EQQPIFOP	Program-interface REXX sample to modify an operation in the current plan
EQQPIFPR	Program-interface REXX sample to list all cyclic periods
EQQPIFWI	Program-interface PL/I sample to modify capacity values in an open interval of a current plan workstation
EQQRXSTG	An assembler routine to get and free storage for the REXX PIF samples

IBM Z Workload Scheduler Application Programming Interface

This section provides details of SEQQSAMP members that can help you use the application programming interface (API).

API buffer examples

SEQQSAMP contains samples that show you how to use the EQQUSIN subroutine. Because the format of the buffers used by EQQUSIN is the same for requests made through the API, you can use these samples to develop API applications. But these samples do not show you how to invoke APPC services; you must develop your own transaction programs (TPs) that initialize and allocate a conversation with IBM Z Workload Scheduler. Use the EQQUSIN samples to create buffers that your TPs can pass to IBM Z Workload Scheduler.

The IBM Z Workload Scheduler sample library SYS1.SAMPLIB contains many APPC samples in a variety of languages. All APPC samples have member names starting with ATB.

IBM Z Workload Scheduler program interface

This section provides details of the SEQQSAMP members, which are samples that use the IBM Z Workload Scheduler program interface (PIF).

The IBM Z Workload Scheduler program interface lets you automate and integrate tasks that must otherwise be performed manually by operators or schedulers.

Install all PIF programs that you use in the production environment as SMP/E usermods to ensure that they are correctly relinked if PIF maintenance is received.

These samples demonstrate practical implementations. Some might fit your requirements exactly.

JS data set maintenance

The sample library contains PIF programs to perform maintenance on the JCL repository (JS) data set. EQQPIFDJ deletes the JCL for an operation from the JS file if the entire occurrence of the application is completed. JCL is deleted from the JS file if the JCL can be located and the input arrival time of the application is earlier than current-plan end.

EQQPIFJD can either list or delete records in the JS file for the given SYSIN criteria. This program deletes the JCL for occurrences from the JS file if the entire application status is complete. JCL is deleted from the JS file if the JCL can be located and the input arrival time of the occurrence is earlier than the current-plan end and the input arrival time specified for the input parameter.

The application name can be specified generically. You can use this program to delete all JCL from the JS file with input arrival equal to or earlier than a specific date. Consider scheduling this program regularly.

JCL variable substitution

SEQQSAMP contains PIF samples for JCL variable substitution actions. EQQPIFAP provides a PL/1 program to retrieve JCL and resolve all nonpromptable setup variables. The program can be called as a CLIST, REXX exec, or ISPF edit macro. Both CLIST and REXX versions are included in the sample.

You might find this program useful in resolving date and day variables that are shared between business systems, particularly in cases where one system is running late.

EQQPIFJV is a sample written in PL/I that can perform general maintenance on the JCL variable tables. You can delete, copy, create, and modify JCL variable tables using this program. EQQPIFJC is a sample written in COBOL that provides the same functions as EQQPIFJV.

Current plan and LTP actions

The majority of the current plan and LTP PIF samples can be found in a single SEQQSAMP member called EQQPIFCB. This member contains these assembler language programs:

- EQQADD adds an occurrence to the CP or LTP.
- EQQDEL deletes an occurrence from the CP or LTP.
- EQQARES adds a special resource to an operation in the CP.
- EQQDRES deletes a special resource from an operation in the CP.
- EQQAPRE adds a predecessor dependency to an occurrence in the CP.
- EQQDPRE deletes a predecessor dependency from an occurrence in the CP.

These programs can be of use if your business systems have dynamic job scheduling requirements. You can use them to build large job streams with a minimum of effort.

Member EQQPIFWI contains a PL/I sample to modify the capacity ceilings of parallel servers and workstation fixed resources for a particular open interval of a workstation in the current plan. You could use this to automatically reflect fixed resource availability affected by hardware problems.

Member EQQPIFOP contains a REXX sample to modify an operation in the current plan.

Other PIF samples

EQQPIFAD provides a PIF sample to define an application description. Use the program-interface for AD and OI updates only if you cannot satisfy your requirements using the batch loader. The batch loader, although itself implemented using the program interface, provides a purpose-built interface for batch updates to application descriptions and operator instructions.

The SEQQSAMP member EQQPIFCL shows the use of the DAYSTAT CLIST. DAYSTAT retrieves calendar information from IBM Z Workload Scheduler and determines if an input date is a work day or a free day. The result is returned as an indicator in a TSO CLIST variable.

EQQPIFPR is a REXX sample that lists all cyclic periods.

EQQRXSTG is an assembler routine that you can use to get and free storage for PIF samples that are written in REXX.

Notices

This document provides information about copyright, trademarks, terms and conditions for product documentation.

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.

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

access authorization 244
AD record
  deleting 35
  format 371
  high date considerations 29
  inserting 52
  replacing 99
  selecting 106
AD segment, replacing 99
ADAPD segment, record format 372
ADCIV segment, record format 373
ADCNC segment
  record format 376
ADCNS segment
  record format 376
ADCOM segment
  listing 69
  record format 374
  selecting 106
ADD OCL instruction 251
ADDCOND OCL instruction 253
ADDEP segment, record format 375
ADDPOP OCL instruction 258
ADDPRED OCL instruction 263
ADDRES OCL instruction 266
ADDSIMP OCL instruction 269
ADEXT segment, record format 377
ADKEY segment
  listing 69
  record format 377
ADLAT segment, record format 378
ADOP segment, record format 378
ADRE segment, record format 381
ADRROUT 29
ADRUN segment
  ADRVALT field 29
  record format 382
ADSAI segment
  record format 385
ADSR segment, record format 386
ADUSF segment, record format 387
advanced program-to-program communication (APPC) 122
ADVDD segment, record format 387
ADXIV segment, record format 389
APAR
  PH50320 406
  PH68409 49
APARs
  PH11184 338
  PH20767 219
  PH31359 442
  PH47355 128
  PH47477 75
  PH50349 241, 242, 247, 247, 248, 248, 248, 248, 249
  PH51620 48
  PH66871 48
  PI26372 46
  PI63875 145
  PI70574 119
  PI77416 119
  PI79679 242
  PK00109 253
  PK40969 46, 241
  PK57435 121
  PK68762 356, 362
  PK87319 476
  PK88734 374, 395, 457
  PK96689 145
  PM01090 21
  PM05335 145, 145
  PM07014 292
  PM10370 46, 241
  PM23805 93
  PM60019 171
  PQ77970 146, 180, 258, 403
  PQ82419 304
  PQ87120 421
  PQ89261 150, 167
  PQ92888 145
  PQ95011 146, 147
  PQ95637 151, 152, 152, 153, 156, 171, 217, 218
  PQ96176 146, 147, 219
  PQ97482 55
  PQ99252 146, 188
  PQ99633 160
API (application programming interface)
  APPC 122
  application transaction program (ATP) 122
  buffer layouts 124
    APP section 126
    APPDAT section 133
    APPFLD section 132
    APPOBJ section 128
    APPSEL section 131
    APPVAL section 132
  communication, establishing 122, 122
  CPI-C verbs 123
    CMACCP 123
    CMALLC 123
    CMCFMD 123
    CMINIT 123
    CMRCV 123
    CMSED 124
    CMSERR 124
    CMSLD 124
    CMSTPN 124
  EBCDIC to ASCII
    code conversion 123
  objects 134
    BACKUP_EVENT fields 486
    CP_OPEN_INTERVAL fields 480
    CP_OPER_EVENT fields 481
    CP_OPERATION fields 467
    CP_OPINFO_EVENT fields 484
    CP_RESOURCE fields 477
    CP_SR_EVENT fields 485
    CP_STATUS fields 466
    CP_WORK_STATION fields 478
    CP_WS_EVENT fields 487
    selecting instances 135
    selection field types 136
  partner transaction program 123
  reason codes 137
  return codes 137
  samples 489
  security 140
  selection criteria, specifying 136
  send/receive buffer formats 124
  transaction program (TP) names 122, 124, 140
  uses of 122
APP (fixed section)
  description 126
  record format 126
APPC (advanced program-to-program communication) 122
APPDAT (data section)
  description 133
  record format 133
APPFLD (field section)
  description 132
  record format 132
application program interface
  overview 369
  REST 369
application programming interface (API)
  APPC 122
  application transaction program (ATP) 122
  buffer layouts 124
    APP section 126
    APPDAT section 133
    APPFLD section 132
    APPOBJ section 128
    APPSEL section 131
    APPVAL section 132
  communication, establishing 122, 122
  CPI-C verbs 123
    CMACCP 123
    CMALLC 123
    CMCFMD 123
    CMINIT 123
    CMRCV 123
    CMSED 124
    CMSERR 124
    CMSLD 124
    CMSTPN 124
  EBCDIC to ASCII code conversion 123
  objects 134
    BACKUP_EVENT fields 486
    CP_OPEN_INTERVAL fields 480
    CP_OPER_EVENT fields 481
    CP_OPERATION fields 467
    CP_OPINFO_EVENT fields 484
    CP_RESOURCE fields 477
    CP_SR_EVENT fields 485
    CP_STATUS fields 466
    CP_WORK_STATION fields 478
    CP_WS_EVENT fields 487
    selecting instances 135
    selection field types 136
  partner transaction program 123
  reason codes 137
  return codes 137
  samples 489
  security 140
  selection criteria, specifying 136
  send/receive buffer formats 124
  transaction program (TP) names 122, 124, 140
  uses of 122
application transaction program (ATP) 122
APPOBJ (object section)
  description 128
  record format 128
APPSEL (selection section)
  description 131
  record format 131
APPVAL (selection value section)
  description 132

record format 132
ASCII 123
ATP (application transaction program) 122
AWSCL record, deleting 36
AWSCL segment
listing 70
record format 390
selecting 107

B

BACKUP_EVENT object, field names 486
Batch Command Interface Tool
BCIT output 147
description 144
GROUPDEF support 170
instructions 149
COPY 151
COPY AD 152
COPY JCLV 153
COPY OI 153
COPY RG 154
DELETE 154
DELETE AD 155
DELETE CPCOND 156
DELETE CPLAT 157
DELETE CPOC 157
DELETE CPOCPRE 158
DELETE CPOP 159
DELETE CPPRE 160
DELETE CPSIMP 162
DELETE CPSR 164
DELETE JS 164
DELETE JSCOM 165
DELETE LTCPRE 165
DELETE LTOC 166
DELETE LTPRE 167
DELETE OI 167
DELETE RG 168
description 149
EXPORT 169
EXPORT RG 169
IMPORT 171
IMPORT AD 171
IMPORT OI 172
IMPORT RG 173
INSERT 173
INSERT CPCOND 174
INSERT CPLAT 176
INSERT CPOC 177
INSERT CPOP 180
INSERT CPPRE 179, 184
INSERT CPSIMP 186
INSERT CPSR 188
INSERT LTOC 189
INSERT LTPRE 189
LIST 190
LIST ADCOM 192
LIST ADKEY 193
LIST CLCOM 193
LIST CPCONDCO 193
LIST CPOC 194
LIST CPOCCOM 194
LIST CPOPCOM 195
LIST CPWSCOM 198
LIST CPWSVCOM 198
LIST JCLVCOM 199
LIST JSCOM 199
LIST LTOCCOM 199
LIST OICOM 200
LIST PRCOM 201
LIST RGCOM 201
LIST RGKEY 202
LIST WSCOM 202
LIST WSVCOM 203
LISTSTAT 204
LISTSTAT CPOC 204
LISTSTAT CPOPCOM 205
MODIFY 207
MODIFY CPCOND 208
MODIFY CPEXT 209
MODIFY CPOC 209
MODIFY CPOP 210
MODIFY CPREND 215
MODIFY CPRENZ 215
MODIFY CPSAI 216
MODIFY LTOC 216
REPLACE 219
SELECT 219
SELECT AD 220
SELECT CL 221
SELECT CPCOND 221
SELECT CPOC 222
SELECT CPOP 223
SELECT CPST 226
SELECT CPWS 226
SELECT CPWSV 227
SELECT JCLPREP 227
SELECT JCLPREPA 227
SELECT JCLV 228
SELECT JS 228
SELECT LTOC 228
SELECT OI 229
SELECT PR 229
SELECT RG/RGCOM 230
SELECT WS 230
SELECT WSV 231
SETSTAT 232
SETSTAT CPSIMP 232
list 190
options 217
program example 146
return codes 149
uses 144
branching (OCL GOTO instruction) 313
branching target (OCL LABEL instruction) 326
broadcasting events 137, 483
buffer layouts (API)
APP section 126
APPDAT section 133
APPFLD section 132
APPOBJ section 128
APPSEL section 131
APPVAL section 132
description 124
built-in functions (OCL) 354

C

CALL OCL instruction 273
CHGEXTNAME OCL instruction 273
CHGJOB OCL instruction 275
CHGOPSAI OCL instruction 277
CHKAPPL OCL instruction 279
CHKDATE OCL instruction 284
CL record
deleting 36
format 391
selecting 107
CLCOM segment
listing 70
record format 391
selecting 107
CLSD segment
record format 392
CLWD segment
record format 392
Common Programming Interface for Communications (CPI-C)
introduction 122
verbs 123
comparison operations (OCL) 317
COMPL OCL instruction 294
CP_OPEN_INTERVAL objec, field names 480
CP_OPER_EVENT object, field names 481
CP_OPERATION object, field names 467
CP_OPINFO_EVENT object, field names 484
CP_RESOURCE object, field names 477
CP_SR_EVENT object, field names 485
CP_STATUS object, field names 466
CP_WORK_STATION objec, field names 478
CP_WS_EVENT object, field names 487
CPCOND record
deleting 36
inserting 56
modifying 82
CPCOND segment
record format 393
CPCONDCO record
format 393, 393
CPCONDCO segment
listing 70
selecting 108
CPCPR segment
record format 400
CPCSU segment
record format 401
CPEXT record
format 401
CPEXT segment
modifying 82
CPI-C (Common Programming Interface for Communications)
introduction 122
verbs 123
CPIVL, record format 423
CPLAT record
deleting 36
format 401
CPLAT segment
inserting 53
CPOC record
deleting 36
format 395
inserting 54
listing 71
modifying 83
selecting 108
CPOCCOM record
listing 71
selecting 108
CPOCCOM segment
record format 394
CPOCPRE segment
deleting 37
inserting 55
record format 398
CPOCSUC segment
deleting 37
inserting 55
record format 399
CPOP record
deleting 37
format 403
inserting 56

modifying 83
selecting 109
CPOPCOM segment
listing 71
record format 399
selecting 109
CPOPSRU segment
listing 73
record format 410
CPOPT segment
record format 424
CPPRE segment
deleting 37
inserting 58
record format 411
CPREND segment
modifying 86
record format 413
CPRENZ segment
modifying 87
record format 413
CPSAI segment
inserting 58
modifying 87
record format 415
CPSIMP record
deleting 38
inserting 59
CPSIMP segment
record format 394
CPSR segment
deleting 39
inserting 60
record format 416
CPST record
format 418
selecting 106
CPSUC segment
deleting 39
inserting 61
record format 416
CPUSRF record
format 420
inserting 61
CPUSRF segment
deleting 40
modifying 88
record format 420
selecting 111
CPVIVL, record format 427
CPWS record
format 420
modifying 88
selecting 111
CPWSCOM segment
listing 74
record format 420
selecting 111
CPWSV record
format 425
modifying 89
selecting 111
CPWSVCOM segment
listing 74
record format 425
selecting 111
CRITSUCS segment
listing 75
CSR record
modifying 90
selecting 112
CSR segment
selecting 112
CSRCOM record
selecting 112
CSRCOM segment
listing 75
selecting 112

D

data area (PIF), format of 26
default date 28
DEL OCL instruction 297
DELCOND OCL instruction 299
DELETE request (PIF) 32
DELPRED OCL instruction 301
DELRES OCL instruction 303
DELSIMP OCL instruction 306
diagnostic data set, EQQDUMP 21

E

EBCDIC 123
ELSE OCL clause 316
EQQDUMP data set 21
EQQMLIB data set 21
EQQMLOG (message log data set) 21
EQQYCOM subroutine
program interface 19
data records 370
EQQYCOM subroutine (PIF)
communicating with 20
EQQYPARM 28
EQQYPARM (parameter data set) 21
EQQYRJCL sample job 365
EQQYRPRC
sample procedure 366
error messages
application programming interface 137
program interface 22
ETT record
deleting 40
selecting 112
ETT segment
listing 76
selecting 112
events, broadcasting 137, 483
EXECUTE request (PIF), updating the current plan 26, 45
EXIT OCL instruction 310

F

FORCE OCL instruction 310

G

GENDAYS record
listing 76
GOTO OCL instruction 313

H

high date 28
HOLD OCL instruction 313

I

IF-THEN-ELSE OCL instruction 316
INIT initialization statement 28
INIT OCL instruction 319
INIT request (PIF)
beginning a communication session 26
description 46
initialization parameters, specifying 243
input arrival date and time, specifying 247
INSERT request (PIF)
description 49
instructions
logging 245
summary of OCL 237
INTFOPTS initialization statement 28
IVL record
deleting 40
inserting 61
modifying 90

J

JCL preparation
performing 119
simulating 121
trial 121
variable substitution 119
JCLPREP record
inserting 62
selecting 112
JCLPREPA record
selecting 113
JCLV record
deleting 40
format 436
inserting 62
replacing 99
selecting 113
JCLVC segment
format 437
JCLVCOM segment
listing 77
record format 436
selecting 113
JCLVD segment, format 438
JCLVV segment
format 437
JL record, deleting 41
JLCOM segment
listing 77
record format 441
selecting 113
JS record
deleting 41
format 439
inserting 52
replacing 99
selecting 113
JSCOM segment
listing 77
record format 439
selecting 113
JSUACT OCL instruction 320
JSVC segment
record format 435
JSVV segment
record format 436

K

KILLJOB OCL instruction 321
KILLREC OCL instruction 323

L

LABEL OCL instruction 326
layout of records and segments
API buffer 126, 134
APP (fixed section) 126
APPDAT (data section) 133
APPFLD (field section) 132
APPOBJ (object section) 128
APPSEL (selection section) 131
APPVAL (selection value section) 132
API objects 466, 488
BACKUP_EVENT 486
CP_OPEN_INTERVAL 480

CP_OPER_EVENT 481
CP_OPERATION 467
CP_OPINFO_EVENT 484
CP_RESOURCE 477
CP_SR_EVENT 485
CP_STATUS 466
CP_WORK_STATION 478
CP_WS_EVENT 487
PIF 371, 450, 461
AD 371
ADAPD 372
ADCIV 373
ADCNC 376
ADCNS 376
ADCOM 374
ADDEP 375
ADEXT 377
ADKEY 377
ADLAT 378
ADOP 378
ADRE 381
ADRUN 382
ADSAI 385
ADSR 386
ADUSF 387
ADVDD 387
ADXIV 389
AWSCL 390
CL 391
CLCOM 391
CLSD 392
CLWD 392
CPCOND 393
CPCONDCO 393
CPCPR 400
CPCSU 401
CPEXT 401
CPIVL 423
CPLAT 401
CPOC 395
CPOCCOM 394
CPOCPRE 398
CPOCSUC 399
CPOP 403
CPOPCOM 399
CPOPSRU 410
CPOPT 424
CPPRE 411
CPREND 413
CPRENZ 413
CPSAI 415
CPSIMP 394
CPSR 416
CPST 418
CPSUC 416
CPUSRF 420, 420
CPVIVL 427
CPWS 420
CPWSCOM 420
CPWSV 425
CPWSVCOM 425
CSRCOM 430
CSRIVL 433
ETT 434
GENDAYS 435
JCLV 436, 436
JCLVC 437
JCLVCOM 436
JCLVD 438
JCLVV 437
JLCOM 441, 441
JS 439
JSVC 435
JSVV 436
LTCPRE 444
LTCSUC 445
LTEXT 446
LTOC 441, 441, 442
LTOCCOM 441
LTOP 444
LTPRE 445
LTSUC 446
OI 447
OICOM 447
PR 448
PRCOM 448
RG 450
RGCOM 450
RGRUN 451
SR 453
SRCOM 453
WS 456
WSAM 460
WSCOM 456
WSDEST 458
WSIVL 459
WSOPT 461
WSSD 459
WSV 462
WSVCOM 462
WSVIVL 464
WSVSD 464
WSVWD 465
WSWD 460
LEFT built-in function (OCL) 355
LIST request (PIF) 64
LTCPRE segment
deleting 41
record format 444
LTCSUC segment
record format 445
LTEXT segment
record format 446
LTOC record
deleting 41
format 441
inserting 62
modifying 91
selecting 114
LTOC segment
record format 442
LTOCCOM segment
listing 78
record format 441
selecting 114
LTOP segment
record format 444
LTPRE segment
deleting 42
inserting 63
record format 445
LTSUC segment
record format 446

M

MATCHTYP record
list 69
message library data set (EQQMLIB) 21
message log data set
EQQMLOG 21
MLOGDDN 21, 47
MLOGDDN data set 21, 47
MODCOND OCL instruction 326
MODIFY request (PIF) 80
MODOP OCL instruction 329

N

NOP OCL instruction 335

O

OCL built-in functions 354
LEFT 355
RIGHT 354
SUBSTR 354
OCL instructions
comparison operations 317
description 251
ELSE clause 316
specifying 246
THEN clause 316
OI record
deleting 42
format 447
inserting 52
replacing 99
selecting 114
temporary, deleting 42
OICOM segment
listing 78
record format 447
selecting 114
online tools 144
OPC Control Language (OCL)
access authorization 244
advantages 236
built-in functions 354
customizing 240
description of instructions 251
executed instruction logging 245
input arrival date and time 247
instruction summary 237
message format 367
overview 236
requirements 365
uses 236
OPSTAT OCL instruction 337
OPTIONS request (PIF) 92

P

parameter data set
EQQYPARM 21
parameter list (PIF)
action code 23
argument names 24
argument values 24
communication block 25
data area 24
overview 22
resource code 23
return code 25
partner transaction program 123
PIF (program interface)
communication session
beginning 26, 26, 46
ending 26, 118
data area, format of 26
data sets
EQQDUMP 21
EQQMLIB 21
EQQYPARM 21
MLOGDDN 47
required 21
description 19, 19
error messages 22

JCL preparation 119
simulating 121, 121
trial 121
parameter list 22
action code 23
argument names 24, 24
argument values 24
communication block 25
data area 24
resource code 23
return code 25
record formats 371, 450, 461
requests
DELETE 32
description 31
EXECUTE 45
INIT 46
INSERT 49
LIST 64
MODIFY 80
OPTIONS 92
REPLACE 97
RESET 100
SELECT 101
sequence of 26
SETSTAT 117
TERM 118
samples 490
security 29
variable substitution 119
PIF high date 28
PR record
deleting 43
format 448
selecting 114
PRCOM segment
listing 78
record format 448
selecting 114
program interface (PIF) 19
communication session
beginning 26, 26, 46
ending 26, 118
data area, format of 26
data sets
EQQDUMP 21
EQQMLIB 21
EQQMLOG 21
EQQYPARM 21
MLOGDDN 47
required 21
description 19
error messages 22
JCL preparation 119
simulating 121, 121
trial 121
parameter list 22
action code 23
argument names 24, 24
argument values 24
communication block 25
data area 24
resource code 23
return code 25
record formats 371, 450, 461
requests
DELETE 32
description 31
EXECUTE 45
INIT 46
INSERT 49
LIST 64
MODIFY 80
OPTIONS 92
REPLACE 97
RESET 100
SELECT 101
sequence of 26
SETSTAT 117
TERM 118
samples 490
security 29
variable substitution 119
PROMPTN OCL instruction 341
PROMPTY OCL instruction 343

R

RACF
APPC/MVS 140
application programming interface 140
program interface 29
record formats
API buffer 126, 134
APP (fixed section) 126
APPDAT (data section) 133
APPFLD (field section) 132
APPOBJ (object section) 128
APPSEL (selection section) 131
APPVAL (selection value section) 132
API objects 466, 488
BACKUP_EVENT 486
CP_OPEN_INTERVAL 480
CP_OPER_EVENT 481
CP_OPERATION 467
CP_OPINFO_EVENT 484
CP_RESOURCE 477
CP_SR_EVENT 485
CP_STATUS 466
CP_WORK_STATION 478
CP_WS_EVENT 487
PIF 371, 430, 433, 434, 450, 461
AD 371
ADAPD 372
ADCIV 373
ADCNC 376
ADCNS 376
ADCOM 374
ADDEP 375
ADEXT 377
ADKEY 377
ADLAT 378
ADOP 378
ADRE 381
ADRUN 382
ADSAI 385
ADSR 386
ADUSF 387
ADVDD 387
ADXIV 389
AWSCL 390
CL 391
CLCOM 391
CLSD 392
CLWD 392
CPCOND 393
CPCONDCO 393
CPCPR 400
CPCSU 401
CPEXT 401
CPIVL 423
CPLAT 401
CPOC 395
CPOCCOM 394
CPOCPRE 398
CPOCSUC 399
CPOP 403
CPOPCOM 399
CPOPSRU 410
CPOPT 424
CPPRE 411
CPREND 413
CPRENZ 413
CPSAI 415
CPSIMP 394
CPSR 416
CPST 418
CPSUC 416
CPUSRF 420, 420
CPVIVL 427
CPWS 420
CPWSCOM 420
CPWSV 425
CPWSVCOM 425
GENDAYS 435
JCLV 436, 436
JCLVC 437
JCLVCOM 436
JCLVD 438
JCLVV 437
JLCOM 441, 441
JS 439
JSVC 435
JSVV 436
LTCPRE 444
LTCSUC 445
LTEXT 446
LTOC 441, 441, 442
LTOCCOM 441
LTOP 444
LTPRE 445
LTSUC 446
OI 447
OICOM 447
PR 448
PRCOM 448
RG 450
RGCOM 450
RGRUN 451
SR 453
SRCOM 453
WS 456
WSAM 460
WSCOM 456
WSDEST 458
WSIVL 459
WSOPT 461
WSSD 459
WSV 462
WSVCOM 462
WSVIVL 464
WSVSD 464
WSVWD 465
WSWD 460
RELEASE OCL instruction 346
RELOP OCL instruction 349
RELSUCC OCL instruction 351
REPLACE request (PIF) 97
requests
API
CREATE 125
DEL 125
GET 122, 125, 125
PUT 125

PIF
DELETE 32
description 31
EXECUTE 26, 45
INIT 26, 46
INSERT 49
LIST 64
MODIFY 80
OPTIONS 92
REPLACE 97
RESET 100
SELECT 101
sequence of 26, 26
SETSTAT 117
TERM 26, 118
requirements, OCL 365
RESET request (PIF) 100
REST API
overview 369
return codes
DELETE 169
INSERT 190, 234
LIST 204
LISTSTAT 206
MODIFY 217
SELECT 231
RG record
deleting 43
format 450
selecting 115
RGCOM record
listing 78
selecting 115
RGCOM segment
record format 450
RGKEY record
listing 78
RGRUN segment
record format 451
RIGHT built-in function (OCL) 354

S

sample job EQQYRJCL 365
sample library (SEQQSAMP)
API samples 489
description 489
PIF samples 490
sample procedure EQQYRPRC 366
security
APPC/MVS 140
application programming interface 140
program interface 29
SELECT request (PIF) 101
send/receive buffer formats 124
SEQQSAMP (sample library)
API samples 489
description 489
PIF samples 490
SET OCL instruction 353
SETSTAT request (PIF) 117
SETUPD OCL instruction 355
SR record
deleting 43
format 453
selecting 115
SR segment, listing 79
SRCOM record, selecting 115
SRCOM segment
record format 453
selecting 115
SRSTAT OCL instruction 356
substitution of variables 249
SUBSTR built-in function (OCL) 354
syntax diagrams, how to read xv

T

TERM request (PIF)
description 118
ending a communication session 26, 118
terminate program (OCL EXIT instruction) 310
THEN OCL clause 316
TOD fields 370
transaction program (TP) names 122, 124, 140

U

UNNOP OCL instruction 360
UPD OCL instruction 362

V

valid-to date 28
VALTO argument
high date considerations 29
variable
assign a value 353
change value of 362
set value of 355
variable substitution
introduction 249
program interface 119
simulating 121
trial 121
VIVL record
deleting 43
inserting 63
modifying 91

W

WS record
deleting 44
format 456
selecting 115
WSAM segment
record format 460
WSCOM segment
listing 79
record format 456
selecting 115
WSDEST segment
record format 458
WSIVL segment
record format 459
WSOPT segment
record format 461
WSSD segment
record format 459
WSSTAT OCL instruction 362
WSV record
deleting 44
format 462
selecting 116
WSVCOM segment
listing 79
record format 462
selecting 116
WSVIVL segment
record format 464
WSVSD segment
record format 464
WSVWD segment
record format 465
WSWD segment
record format 460
WTO OCL instruction 364