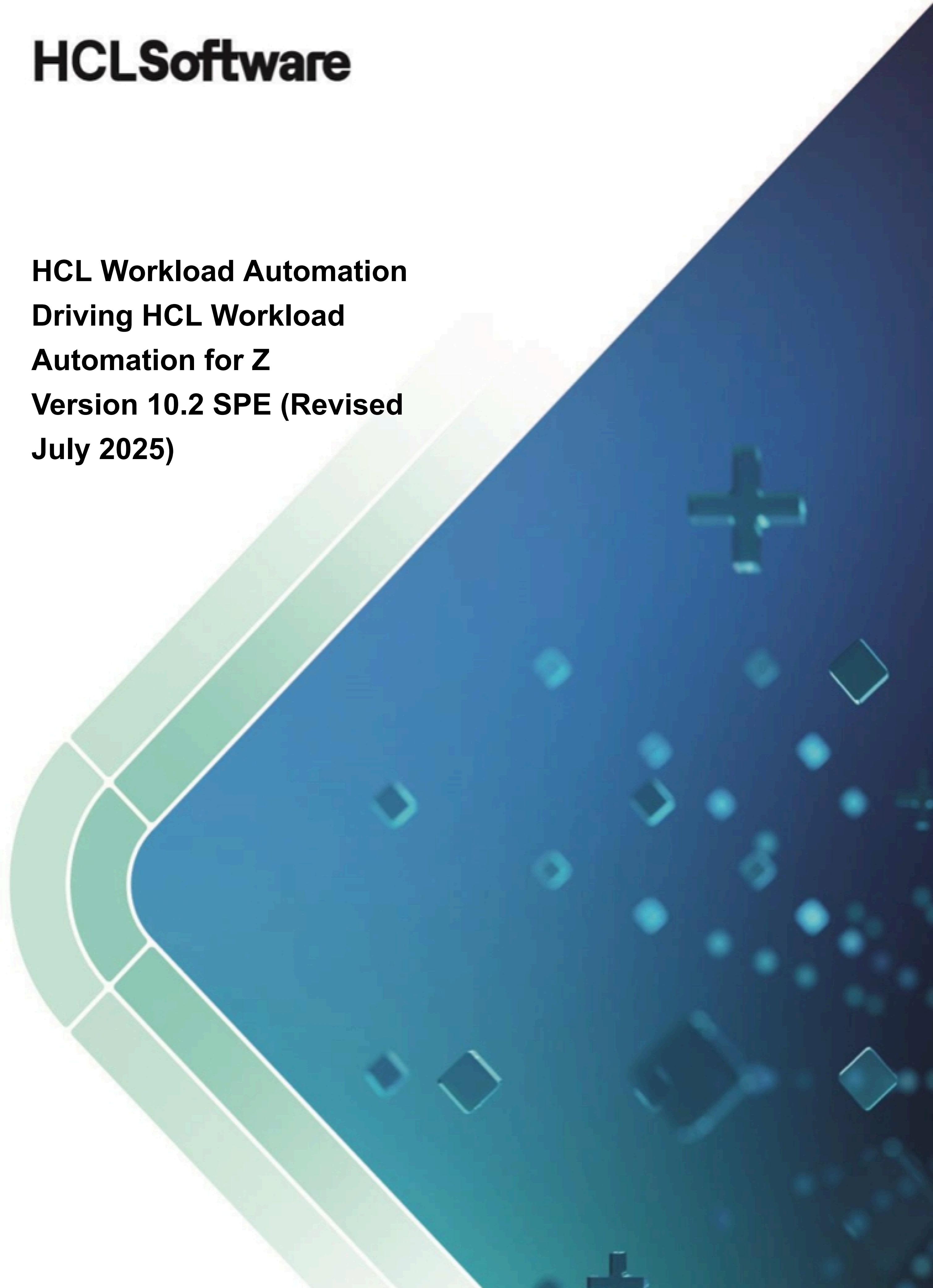
HCLSoftware

HCL Workload Automation Driving HCL Workload Automation for Z Version 10.2 SPE (Revised July 2025)



Note

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

This edition applies to version 10, release 2, modification level 0 of HCL Workload Automation for Z (program number 19OP1249) and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

List of Figures

List of Tables

About this publication

Developer's Guide: Driving HCL Workload Automation for Z shows you how to use the programming interfaces to HCL Workload Automation for Z 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 HCL Workload Automation for Z controller.

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

- *HCL Workload Automation: Developer's Guide: Driving HCL Workload Automation for Z*
- *HCL Workload Automation: Developer's Guide: Extending IBM Workload Automation*
- *Workload Automation Programming Language for z/OS 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 HCL Workload Automation for Z. 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 HCL 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.

Accessibility

Accessibility features help users with a physical disability, such as restricted mobility or limited vision, to use software products successfully.

With this product, you can use assistive technologies to hear and navigate the interface. You can also use the keyboard instead of the mouse to operate all features of the graphical user interface.

For detailed information, see the appendix about accessibility in the *HCL Workload Automation User's Guide and Reference*.

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:

{ **SRSTAT** } ' *resource name* ' [**SUBSYS** ({ **OPCA** | *subsystem name* | **MSTR** })] [**AVAIL** ({ **KEEP** | **RESET** | **NO** | **YES** })] [**DEVIATION** ({ **KEEP** | *amount* | **RESET** })] [**QUANTITY** ({ **KEEP** | *amount* | **RESET** })] [**CREATE** ({ **YES** | **NO** })] [**TRACE** ({ **0** | *trace level* })]

The symbols have these meanings:

►►────

The statement begins here.

────►

The statement is continued on the next line.

►────

The statement is continued from a previous line.

────►◄

The statement ends here.

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

These are the conventions used in the diagrams:

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

```
STATEMENT required item
```

- Optional items appear below the main path:

```
STATEMENT  [optional item]
```

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

```
STATEMENT repeatable item
```

- 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:

    ```
    STATEMENT { required choice 1  | required choice 2 }
    ```

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

    ```
    STATEMENT [ { optional choice 1  | optional choice 2 } ]
    ```

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

    ```
    STATEMENT [ {  | optional choice 1  | optional choice 2  | optional choice 3 } ]
    ```

    ```
    STATEMENT {  | required choice 1  | required choice 2  | required choice 3 }
    ```

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

```
STATEMENT [ { default  | alternative } ]
```

- Keywords appear in uppercase (for example, STATEMENT).
- Parentheses and commas must be entered as part of the command syntax, as shown.
- For complex commands, the item attributes might not fit on one horizontal line. If that line cannot be split, the attributes appear at the bottom of the syntax diagram:

```
STATEMENT { required choice 1 [ optional choice 1 ( { default  | alternative } ) ] [ optional choice 2 ( { default  |
alternative } ) ] | required choice 2 | required choice 3 }
```

Part I. Programming interfaces

Chapter 1. The program interface (PIF)

About this task

This chapter describes the HCL Workload Automation for Z program interface (PIF), which lets a user-written program issue various requests to the HCL Workload Automation for Z subsystem. For example, you can automate actions that you perform through the HCL Workload Automation for Z 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 HCL Workload Automation for Z-supplied communication subroutine, EQQYCOM.

Program interface samples

The HCL Workload Automation for Z sample library shipped with the HCL Workload Automation for Z 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 267.

Related tools

HCL Workload Automation for Z 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 HCL Workload Automation for Z.

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.

Communicating with EQQYCOM

Requests to HCL Workload Automation for Z 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 HCL Workload Automation for Z to perform the required action. With each call to EQQYCOM, you can make one HCL Workload Automation for Z 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 HCL Workload Automation for Z 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 HCL Workload Automation for Z 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

HCL Workload Automation for Z 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 43.

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 HCL Workload Automation for Z diagnostic data set, EQQDUMP. Debugging information is written to this data set, for example, HCL Workload Automation for Z 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 HCL Workload Automation for Z 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 HCL Workload Automation for Z subsystem (for example, an error detected by HCL Workload Automation for Z 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 HCL Workload Automation for Z subsystem.

Parameter overview

The parameter list contains the necessary information for one request. Figure 1: Program interface parameters on page 20 illustrates the basic structure of the parameter list and the addressing linkage to it.

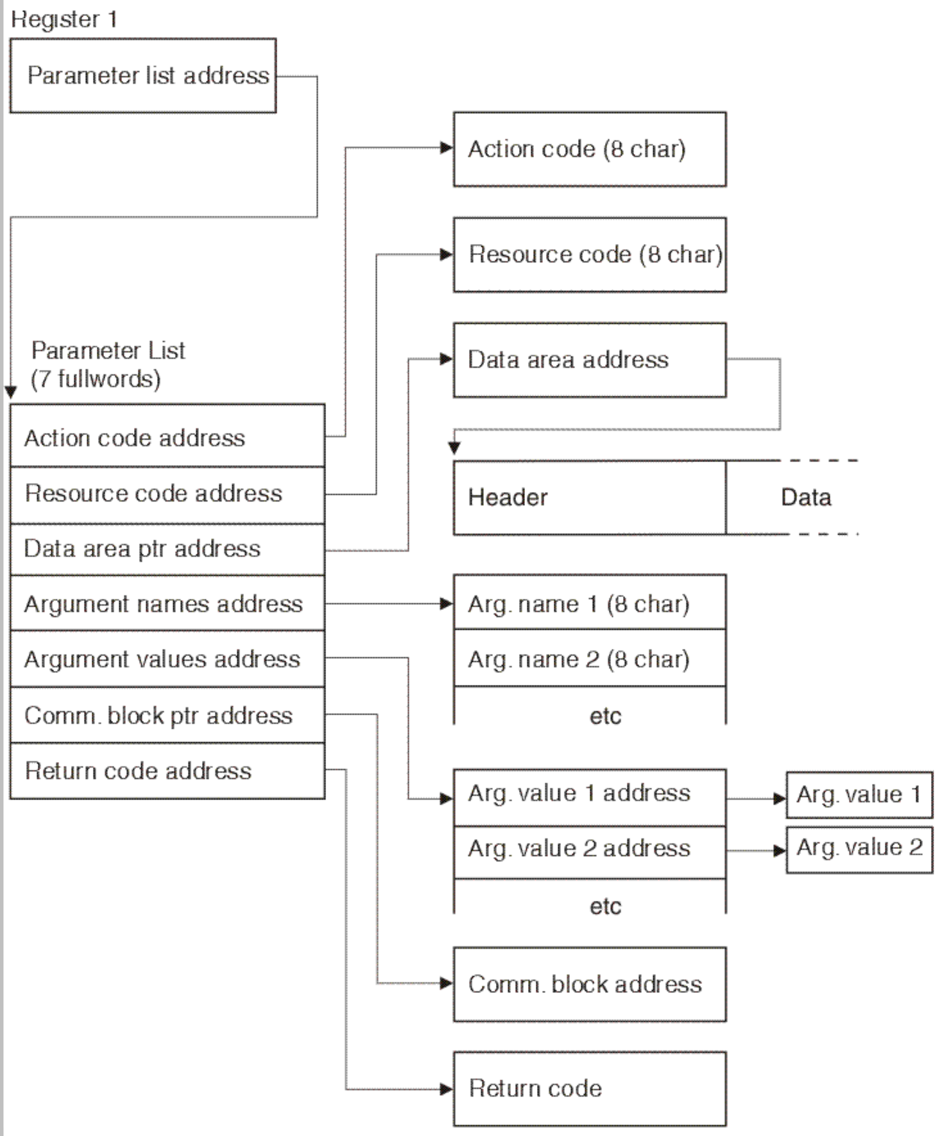
The parameter list must always consist of seven fullwords, representing the seven parameter types outlined here. Not all parameters are required for some requests, in which case you must set the parameter value to hexadecimal zeros. A character-type parameter value that contains blanks also indicates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

Figure 1: Program interface parameters on page 20 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 HCL Workload Automation for Z 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 HCL Workload Automation for Z 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 23.

Argument names and values

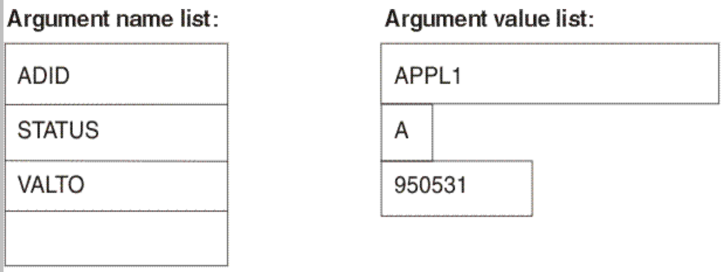
The fourth and fifth fullwords in the parameter list are the addresses of the argument name list and argument value address list.

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 22).

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 HCL Workload Automation for Z 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 HCL Workload Automation for Z 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 144.

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.

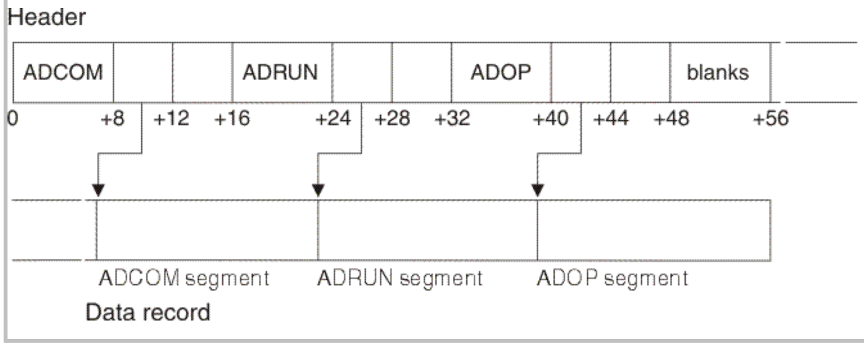
Reserved (4 bytes)

Reserved for use by HCL Workload Automation for Z.

The header is terminated by a header entry with a blank segment name. Figure 3: Program interface data area example on page 24 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 HCL Workload Automation for Z 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 HCL Workload Automation for Z 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

HCL Workload Automation for Z 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 HCL Workload Automation for Z dialogs.

Internal date representation

Internally, HCL Workload Automation for Z works with a two-digit year format, so dates are represented as 00 to 99. In order to handle dates before and after 2000, HCL Workload Automation for Z 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 HCL Workload Automation for Z dialog and report users. They always see the real date. However, PIF requests often involve reading or writing records in an HCL Workload Automation for Z 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 HCL Workload Automation for Z 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 *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. HCL Workload Automation for Z will always interpret this as representing the default valid-to date.

Table 1: Comparison of Date Representations on page 26 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 HCL Workload Automation for Z 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 HCL Workload Automation for Z 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, HCL Workload Automation for Z takes the HCL Workload Automation for Z 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 HCL Workload Automation for Z administrator. Table 2: Access Authority for Program Interface Requests on page 26 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

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

Program interface request	Access type required
DELETE EXECUTE INSERT MODIFY REPLACE SETSTAT	Update

You need authorization to access these HCL Workload Automation for Z fixed resources:

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

Program interface resource	HCL Workload Automation for Z 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 HCL Workload Automation for Z, the program OBJ must be compiled, or at least link-edited, for the current HCL Workload Automation for Z 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 HCL Workload Automation for Z 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 HCL Workload Automation for Z 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.

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 29.

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

Table 5. Delete AD Arguments (continued)

Arg names	Length	Data type	Description
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: HCL Workload Automation for Z 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

Table 9. Delete CPOC Arguments (continued)

Arg names	Length	Data type	Description
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

Table 13. Delete CPPRE Arguments (continued)

Arg names	Length	Data type	Description
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.

Table 14. Delete CPSIMP Arguments (continued)

Arg names	Length	Data type	Description
			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:
			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

Table 23. Delete LTOC Arguments (continued)

Arg names	Length	Data type	Description
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
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

Table 28. Delete RG Arguments (continued)

Arg names	Length	Data type	Description
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 HCL Workload Automation for Z 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 HCL Workload Automation for Z 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 HCL Workload Automation for Z 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, HCL Workload Automation for Z sets the deadline as the IA time + 8 hours.

Argument value=N

Default. HCL Workload Automation for Z 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 HCL Workload Automation for Z V9.5), specify one of the following values:

Argument value=*n*

Identifies the level of the user program. The valid values are:

11

Identifies HCL Workload Automation for Z 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 HCL Workload Automation for Z V9.5.

13

Identifies the enablement of changes done for HCL Workload Automation for Z V9.5 through the Package HWAZ_9502_APAR_HC00001.

14

Identifies the enablement of changes done for HCL Workload Automation for Z V9.5 through the Package HWAZ_9503_APAR_HC00002.

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 HCL Workload Automation for Z 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 HCL Workload Automation for Z 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 HCL Workload Automation for Z:
    - 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.
- 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 HCL Workload Automation for Z 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 91.

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: 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
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 HCL Workload Automation 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:

Table 37. Insert CPCOND Arguments (continued)

Arg names	Length	Data type	Description
			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
JCLASS	1	Char	Job class
JOBCRT	1	Char	Critical job.

Table 38. Insert CPOP Arguments (continued)

Arg names	Length	Data type	Description

			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.

Table 42. Insert CPSR Arguments (continued)

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

Table 45. Insert IVL Arguments (continued)

Arg names	Len	Data type	Description
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

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

Table 48. Insert LTOC Arguments (continued)

Arg names	Length	Data type	Description
JCLVTAB	16	Char	JCL variable table
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 144. 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 HCL Workload Automation 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 23.

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. HCL Workload Automation for Z 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
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

Table 55. List 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: 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 56. 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.
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.

Table 56. List CPOPCOM Arguments (continued)

Arg names	Length	Data type	Description

			‘ ’ (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.
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.

Table 56. List CPOPCOM Arguments (continued)

Arg names	Length	Data type	Description
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 57. List CPOPSRU Arguments

Arg names	Length	Data type	Description
LISTTYPE	5	Char	Type of list, INUSE or WAITQ
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 58. 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 59. 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 60. 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

Table 60. List CRITSUCS Arguments (continued)

Argument name	Length	Data type	Description
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 HCL Workload Automation for Z.			

List CSRCOM arguments

Table 61. 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 62. List ETT Arguments

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

Table 62. List ETT Arguments (continued)

Arg names	Length	Data type	Description
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 63. 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 64. List JCLVCOM Arguments

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

List JLCOM arguments

Table 65. 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 66. 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
WSNAME	4	Char	Workstation name

List LTOCCOM arguments

Table 67. 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 68. List OICOM Arguments

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

List PRCOM arguments

Table 69. List PRCOM Arguments

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

List RGCOM, RGKEY arguments

Table 70. 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 71. 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 72. 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 73. 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 74. 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 75. 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 76. 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 77. Modify 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	Operation descriptive text
DURATION	4	Integer	Estimated duration in 100th of 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
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

Table 77. Modify CPOP Arguments (continued)

Arg names	Length	Data type	Description
MONITOR	1	Char	Y=Operation monitored by an external product N=Operation not monitored by an external product

OPCMD	2	Char	Operation command:
			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 HCL Workload Automation 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.

Table 77. Modify CPOP Arguments (continued)

Arg names	Length	Data type	Description
			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
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 78. 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 79. 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 80. 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 81. 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 82. 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).

Table 82. Modify CPWS Arguments (continued)

Arg names	Length	Data type	Description
			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).
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 83. 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

Table 83. Modify CPWSV 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)

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 84. 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 85. 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 86. 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 87. 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, HCL Workload Automation for Z 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, HCL Workload Automation for Z 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), HCL Workload Automation for Z 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 92. 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 HCL Workload Automation for Z 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 91

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 23. 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 88. 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 HCL Workload Automation for Z by specifying field names and values in arguments, which identify the record you want to retrieve. When you retrieve a record directly from HCL Workload Automation for Z, 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 HCL Workload Automation for Z, 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 HCL Workload Automation for Z, 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 23.

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, HCL Workload Automation for Z 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, HCL Workload Automation for Z 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 HCL Workload Automation for Z on page 105. 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 HCL Workload Automation for Z 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 HCL Workload Automation for Z

When you are retrieving a record directly from HCL Workload Automation for Z 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 29.

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 *Customization and Tuning*.

You can specify the following values.

Select AD, ADCOM arguments

Table 89. 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
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. HCL Workload Automation for Z 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 90. Select AWSCL Arguments

Arg names	Length	Data type	Description
DATE	6	Char YYMMDD	Date

Select CL, CLCOM arguments

Table 91. 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 92. 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 93. 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 94. 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 ********

Table 94. Select CPOP, CPOPCOM Arguments (continued)

Arg names	Length	Data type	Description
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.
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.

Table 94. Select CPOP, CPOPCOM Arguments (continued)

Arg names	Length	Data type	Description
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 95. 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 96. 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 97. 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 98. 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

Table 98. Select CSR, CSRCOM Arguments (continued)

Arg names	Length	Data type	Description
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 99. 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 100. 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 101. 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.

Table 101. Select JCLPREPA Arguments (continued)

Arg names	Length	Data type	Description
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 102. Select JCLV, JCLVCOM Arguments

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

Select JLCOM arguments

About this task

Table 103. 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 104. 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 105. 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 106. 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 107. 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 108. 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

Table 108. Select RG, RGCOM Arguments (continued)

Arg names	Length	Data type	Description
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 109. 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 110. 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
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 111. 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 57.

To identify the new status, use the following argument:

Table 112. Setstat CPSIMP Argument

Arg name	Length	Data type	Description

NEWSTAT	1	Char	Requested status:
			
			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 HCL Workload Automation for Z 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 267 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 HCL Workload Automation for Z. 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 HCL Workload Automation for Z only simulates submission.

- You can supply a time value in the SIMTIME argument for HCL Workload Automation for Z-supplied variables that contain a *current time* value. HCL Workload Automation for Z 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 HCL Workload Automation for Z application programming interface (API) to communicate with HCL Workload Automation for Z. 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 HCL Workload Automation for Z (CREATE request).

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

This chapter describes CPI-C verbs that are supported by HCL Workload Automation for Z. 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 HCL Workload Automation for Z to help you set up and use the API. For a description of these samples, see Sample library (SEQQSAMP) on page 267.

Communicating with HCL Workload Automation for Z

To establish communication with HCL Workload Automation for Z, 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 HCL Workload Automation for Z. 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 HCL Workload Automation for Z 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 138 describes how you broadcast events on the target system.

When communication is established, your ATP sends a request to HCL Workload Automation for Z in a *send buffer*. HCL Workload Automation for Z 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, HCL Workload Automation for Z 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 HCL Workload Automation for Z must be in EBCDIC format. HCL Workload Automation for Z 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 HCL Workload Automation for Z, 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 HCL Workload Automation for Z

Your ATPs can issue requests to HCL Workload Automation for Z 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, HCL Workload Automation for Z, 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 HCL Workload Automation for Z when a confirm verb is issued by the ATP. But HCL Workload Automation for Z 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 HCL Workload Automation for Z 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 HCL Workload Automation for Z.

CMSERR

Send_Error

CMSERR can be issued but is not used by HCL Workload Automation for Z.

CMSLD

Set_Log_Data

CMSLD can be issued but is not used by HCL Workload Automation for Z.

CMSTPN

Set_TP_Name

Specify TP name EQQAPI, which is the default name. HCL Workload Automation for Z 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 HCL Workload Automation for Z 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 113: Contents of a Send Buffer on page 125 shows the sections that you can include for each request:

Table 113. 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]

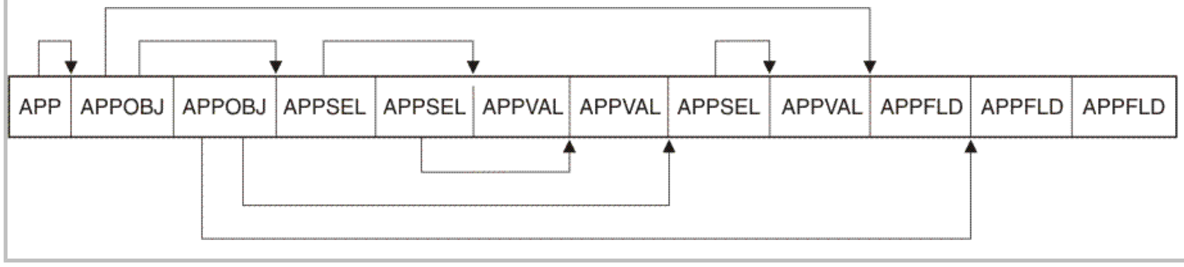
Note:

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 HCL Workload Automation for Z, the buffer contains the entire send buffer. Some fields are updated by HCL Workload Automation for Z, 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, HCL Workload Automation for Z 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 HCL Workload Automation for Z 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 114. 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

Table 114. App-Fixed Section (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
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
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 HCL Workload Automation for Z 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, HCL Workload Automation for Z 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 HCL Workload Automation for Z. In the send buffer, set this field to binary zeros (X'00'). For more information, see Return codes and reason codes generated by HCL Workload Automation for Z on page 138.

APP_RSNCODE

Is the reason code that is set by HCL Workload Automation for Z. In the send buffer, set this field to binary zeros (X'00'). For more information, see Return codes and reason codes generated by HCL Workload Automation for Z on page 138.

APP_ERR_OFF

Is set by HCL Workload Automation for Z 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 HCL Workload Automation for Z 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 115. 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

Table 115. APPOBJ-Object Section (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
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)

Note: Descriptions prefixed with an asterisk (*) indicate fields that HCL Workload Automation for Z updates.

In the object section:

APPOBJ_NAME

Identifies the object type. For a description of valid names, see Specifying object names on page 135.

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 136 .

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, HCL Workload Automation for Z 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. HCL Workload Automation for Z 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 HCL Workload Automation for Z. In the send buffer set this field to binary zeros (X'00'). For more information, see Return codes and reason codes generated by HCL Workload Automation for Z on page 138.

APPOBJ_RSN

Is the object level reason code that is set by HCL Workload Automation for Z. In the send buffer set this field to binary zeros (X'00'). For more information, see Return codes and reason codes generated by HCL Workload Automation for Z on page 138.

APPOBJ_AUTH

Is the access authority (read or update) that your ATP has to the specified object. For GET, PUT, and DEL requests, HCL Workload Automation for Z 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 HCL Workload Automation for Z 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, HCL Workload Automation for Z returns all instances of the object.

The selection section has this format:

Table 116. 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

Table 116. APPSEL-Selection Section (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
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 136. Field names are described in API object fields on page 244.

APPVAL - Selection value section

This section contains a value that you want HCL Workload Automation for Z 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 117. 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 244.

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. HCL Workload Automation for Z 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 118. 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

Table 118. APPFLD-Field Section (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
20	(14)	CHARACTER	4	APPFLD_TYPE	*FIELD DATA TYPE
Note: Descriptions prefixed with an asterisk (*) indicate fields that HCL Workload Automation for Z 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 138.

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 HCL Workload Automation for Z 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. HCL Workload Automation for Z returns in the receive buffer one data section for each instance of the object. Each APPOBJ section in the send buffer is updated by HCL Workload Automation for Z 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 119. APPDAT-Data Section

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	*	APPDAT	DATA SECTION ADDRESS OF FIRST DATA SECTION FOR THIS

Table 119. APPDAT-Data Section (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
					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 HCL Workload Automation for Z retrieves. See API object fields on page 244.

Specifying object names

You identify the object type by specifying a name in the APPOBJ_NAME field of the object section. Table 120: API Object Names on page 135 describes the object names that you can specify:

Table 120. 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.
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

HCL Workload Automation for Z 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 HCL Workload Automation for Z 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 HCL Workload Automation for Z. 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 244 describes the field names and field values of each object, and the selection type of each field. There are select fields:

Required

You can specify these operators in the APPSEL section:

Table 121. Operators That You Can Specify in the APPSEL Section

Operator	Description
EQ or =	Equal to
NE or ^=	Not equal to
GT or >	Greater than
LT or <	Less than
GE or >=	Greater than or equal to
LE or <=	Less than or equal to

Table 121. Operators That You Can Specify in the APPSEL Section (continued)

Operator	Description
GN	Generic compare

Broadcasting events

The LU that your ATP sends requests to is owned by an HCL Workload Automation for Z 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 HCL Workload Automation for Z 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 HCL Workload Automation for Z 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 HCL Workload Automation for Z 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 244.

Return codes and reason codes generated by HCL Workload Automation for Z

If a request through the API causes a severe error in an HCL Workload Automation for Z 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 HCL Workload Automation for Z 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, HCL Workload Automation for Z can generate return codes and reason codes for the various requests that are made. Your program can test the results of the call to HCL Workload Automation for Z 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 HCL Workload Automation for Z 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. HCL Workload Automation for Z 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 HCL Workload Automation for Z 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 HCL Workload Automation for Z address space that the request was sent to.

Security

The access to HCL Workload Automation for Z can be controlled through security mechanisms provided by:

- APPC and RACF®
- HCL Workload Automation for Z 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.

HCL Workload Automation for Z recognizes these TP names:

EQQTRK

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*.

HCL Workload Automation for Z and RACF®

HCL Workload Automation for Z 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, HCL Workload Automation for Z does not perform security checking, because the request could be directed to more than one HCL Workload Automation for Z 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 HCL Workload Automation for Z resources at these levels:

1. The HCL Workload Automation for Z 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 HCL Workload Automation for Z 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 122: Subresource Protection for Requests through the API on page 142.

Table 122. 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
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 HCL Workload Automation for Z 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*.

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 HCL Workload Automation for Z databases or data sets.

Note: For a correct interpretation of the fields described as "Tod clock at last update", see TOD fields on page 144.

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 123. 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 124. 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 124. 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 144.

ADAPD - Application dependency segment

The application dependency part of an Application Description.

Table 125. 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 125. 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 126. 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 126. 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 127. 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

Table 127. ADCOM Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
56	(38)	CHARACTER	8	ADGROUP	AUTHORITY GROUP NAME
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 128. 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

Table 128. ADDEP Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
32	(20)	CHARACTER	50	ADDEPDESC	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 129. 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 130. 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 131. 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 131. 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

Table 131. ADOP Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
120	(78)	CHARACTER	1	ADOPMON	OPERATION MONITORED
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.

Table 131. ADOP Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
139	(8B)	CHARACTER	21	*	RESERVED

ADRE - Remote job information segment

A segment containing the remote job information.

Table 132. ADRE Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	96	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	12	*	RESERVED

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 133. ADRUN Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	228	ADRUN	RUNCYCLE SECTION

Table 133. ADRUN Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
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 134. 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 135. 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.

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 136. 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 137. 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 137. 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 138. 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 139. 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 139. 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 140. 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 140. 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 141. 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 141. 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 144.

CLCOM - Common segment

Common description of a calendar.

Table 142. 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 142. 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 143. 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 144. 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 145. CPOC Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	428	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 145. 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 145. 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 145. CPOC Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
420	(1A4)	CHARACTER	8	CPOCRUNC	RUN CYCLE THAT GENERATED THE OCCURRENCE

CPOCPRE - Occurrence predecessor segment

Current plan occurrence predecessor.

Table 146. 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 146. 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 147. 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 148. 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 149. 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 150. CPOP Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	440	CPOPCOM	CURRENT PLAN OPERATION

Table 150. 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 150. 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 150. 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 150. 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 150. 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
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		NOT USED
396	(18C)	SIGNED	4	CPOP#CPROP	NUMBER OF CONDITIONAL PREDECESSORS
400	(190)	SIGNED	4	CPOP#CSUOP	NUMBER OF CONDITIONAL SUCCESSORS
404	(194)	SIGNED	4	CPOP#CONDTOT	NUMBER OF CONDITIONS
408	(198)	SIGNED	4	CPOP#COND_T	NUMBER OF TRUE CONDITIONS

Table 150. CPOP Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
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 ' ' (blank) = Only an alert message is issued. A = Only an alert message is issued.

Table 150. CPOP Control Block (continued)

Offsets					
Dec	Hex	Type	Len	Name	Description
					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 151. 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
30	(1E)	CHARACTER	8	CPOPUJBN	OP OS JOBNAME | BLANK

Table 151. CPOPSRU Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
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 152. 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 152. 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 153. 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 154. CPRENZ Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	68	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)	CHARACTER	10	CPRZIA	INPUT ARRIVAL (BROWSE)
40	(28)	CHARACTER	6	CPRZIAD	DATE | BLANK
46	(2E)	CHARACTER	4	CPRZIAT	TIME | BLANK
50	(32)	CHARACTER	28	*	

CPSAI - Operation system automation information segment

System automation information.

Note: This segment exists for system automation operations only.

Table 155. 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

Table 155. CPSAI Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
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
344	(158)	CHARACTER	8	*	RESERVED

CPSUC - Successor segment

Current plan operation successor.

Note: Y and N are the indicator values.

Table 156. 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

Table 156. CPSUC Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
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 157. CPSR Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	66	CPSR	
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 158. 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
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

Table 158. CPREC Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
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

HCL Workload Automation for Z sets this value when it is started and no turnover is in progress.

' '

HCL Workload Automation for Z 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 159. 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 160. 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 161. 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 162. 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
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

Table 162. CPWS Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
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

Table 162. CPWS Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
105	(69)	CHARACTER	1	CPWSPREP	JOB SETUP ABILITY
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 163. CPIVL Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	64	CPIVL	WORKSTATION IVL
0	(0)	CHARACTER	10	CPIVLFR	INTERVAL START

Table 163. CPIVL Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	CHARACTER	6	CPIVLFD	DATE YYMMDD
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 164. CPOPT Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE		CPOPT	Workstation options

Table 164. CPOPT Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
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
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 165. 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
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)

Table 165. CPWSV Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
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
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	*	RESERVED
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 166. 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
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 167. CRITSUCS Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE		CRITJOBS	CRITICAL OPERATION

Table 167. CRITSUCS Control Block (continued)

Offsets					
Dec	Hex	Type	Len	Name	Description
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
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

Table 167. CRITSUCS Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
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>
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 144.

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 168. 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)

Table 168. CSRCOM Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
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
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

Table 168. CSRCOM Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
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
232	(E8)	CHARACTER	1	CSRCIAVAIL	AVAILABLE (Y|N)
233	(E9)	CHARACTER	7	*	RESERVED

CSRIVL - Current plan special resource interval segment

Table 169. 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

Table 169. CSRIVL Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
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 170. 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 171. 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 172. ETT Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	BASED	128	ETT	ETT TRACKING CRITERIA REC

Table 172. ETT Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
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 173. 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 174. 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 175. 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 144.

JCLVC - Common segment

Identifies a JCL variable table.

Table 176. 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 177. 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 178. 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 179. 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

Table 179. JS Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
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
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 180. 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

Table 180. JLCOM Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
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 181. 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
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

Table 181. LTOC Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
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
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 182. 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

Table 182. LTOP Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
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 183. 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
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 184. LTSUC Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	32	LTSUC	OCCURRENCE SUCCESSOR
0	(0)	CHARACTER	26	LTSUCKEY	SUCCESSOR IDENTIFIER

Table 184. LTSUC Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
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 185. 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
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 186. 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
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

Table 186. OI Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
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 187. PR Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	96	PRCOM	PERIOD DEFINITION

Table 187. PR Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
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 144.

Table 188. 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 189. 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 190. 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 190. 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 191. 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 191. 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 192. 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 193. 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

Table 193. SRCOM Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
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 144.

Table 194. 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

Table 194. SRIVL Segment (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
24	(18)	SIGNED	4	SRIVLWSCNUM	NUMBER OF CONNECTED WSs
28	(1C)	CHARACTER	1	SRIVLAVAIL	AVAILABLE (Y|N)
29	(1D)	CHARACTER	3	RESERVED	RESERVED

Table 195. 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 196. SRDWS Segment

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	CHARACTER	8	SRDWS	CONNECTED WS SEGMENT
0	(0)	CHARACTER	4	SRDWSNAME	WORKSTATION NAME
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 144.

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 197. 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
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

Table 197. WSCOM Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
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		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 198. 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 199. 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 200. WSSD Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	48	WSSD	
0	(0)	CHARACTER	6	WSSDDATE	SPECIFIC DATE
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 201. 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 202. WSAM Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	80	WSAM	
0	(0)	CHARACTER	12	WSAMACC	ACCESS METHOD NAME

Table 202. WSAM Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
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 203. 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 144.

WSVCOM - Common segment

Common description of a virtual workstation destination.

Workstation types:

C

Computer

Reporting attribute:

A

Automatic reporting

Table 204. 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 204. 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 205. 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 206. 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 207. WSWD Control Block

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	48	WSVWDD	
0	(0)	CHARACTER	8	WSVWDDDAY	WEEK DAY

Table 207. WSWD Control Block (continued)

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
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 HCL Workload Automation for Z 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 208. 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 209. 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 209. 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 209. 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 209. 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 209. 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 209. 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 209. 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 209. 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 209. 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 209. 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 210. 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 211. 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 211. 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 211. 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 212. 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 213. CP_OPER_EVENT Object Fields

Field	Type	Len	Description	APPSEL	APPFLD
SUBSYSTEM_NAME	CHAR	4	Subsystem name	O	N

Table 213. 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 213. 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 HCL Workload Automation for Z 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 HCL Workload Automation for Z subsystems on the same z/OS image. If you do not specify SUBSYSTEM_NAME, the event is reported to the HCL Workload Automation for Z 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 HCL Workload Automation for Z subsystems on the same z/OS image.
3. If you do not provide enough information to uniquely identify the operation, and HCL Workload Automation for Z finds more than one operation that matches the criteria you specified, HCL Workload Automation for Z must determine the most applicable operation to update. HCL Workload Automation for Z 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 214. 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 HCL Workload Automation for Z 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 261.
3. If you do not provide enough information to uniquely identify the operation, and HCL Workload Automation for Z finds more than one operation that matches the criteria you specified, HCL Workload Automation for Z must determine the most applicable operation to update. HCL Workload Automation for Z considers operations in status R, A, *, S, I, or E when selecting the operation. HCL Workload Automation for Z 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, HCL Workload Automation for Z 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 215. 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 215. 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 261.
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. HCL Workload Automation for Z 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 HCL Workload Automation for Z 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 HCL Workload Automation for Z 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 HCL Workload Automation for Z 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 216. 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 261.

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 217. 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 217. 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 261.
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 HCL Workload Automation for Z programming interfaces. In most cases, you need only add installation-specific JCL to adapt a member in SEQQSAMP to your requirements. Table 218: SEQQSAMP Library Members for Programming Interfaces and the API on page 267 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 HCL Workload Automation for Z is found in *HCL Workload Scheduler for Z: 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 218. 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

HCL Workload Automation for Z 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 HCL Workload Automation for Z. Use the EQQUSIN samples to create buffers that your TPs can pass to HCL Workload Automation for Z.

The HCL Workload Automation for Z sample library SYS1.SAMPLIB contains many APPC samples in a variety of languages. All APPC samples have member names starting with ATB.

HCL Workload Automation for Z program interface

This section provides details of the SEQQSAMP members, which are samples that use the HCL Workload Automation for Z program interface (PIF).

The HCL Workload Automation for Z 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

The SEQQSAMP member EQQPIFCL shows the use of the DAYSTAT CLIST. DAYSTAT retrieves calendar information from HCL Workload Automation for Z 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

Trademarks

Terms and conditions for product documentation

Index