



@server

iSeries

CL Commands Volume 1





@server

iSeries

CL Commands Volume 1

Contents

Command Descriptions	1
ADDACC (Add Access Code) Command Description	1
ADDALRACNE (Add Alert Action Entry) Command Description	2
ADDALRD (Add Alert Description) Command Description	4
ADDALRSLTE (Add Alert Selection Entry) Command Description	8
ADDAUTLE (Add Authorization List Entry) Command Description	13
ADDAJE (Add Autostart Job Entry) Command Description	16
ADDBNDDIRE (Add Binding Directory Entry) Command Description	17
ADDBKP (Add Breakpoint) Command Description	20
ADDCCSCLT (Add Change Control Server Client) Command Description	26
ADDCRQA (Add Change Request Activity) Command Description	34
ADDCLUNODE (Add Cluster Node Entry) Command Description	57
ADDCRGDEVE (Add Cluster Resource Group Device Entry) Command Description	59
ADDCRGNODE (Add Cluster Resource Group Node Entry) Command Description	61
ADDCMDCRQA (Add Command Change Request Activity) Command Description	63
ADDCMNE (Add Communications Entry) Command Description	71
ADDCOMSNMP (Add Community for SNMP) Command Description	74
ADDCFGLE (Add Configuration List Entries) Command Description	76
ADDCNNLE (Add Connection List Entry) Command Description	84
ADDDTADFN (Add Data Definition) Command Description	88
ADDDEVDMNE (Add Device Domain Entry) Command Description	90
ADDDIRE (Add Directory Entry) Command Description	91
ADDDIRSHD (Add Directory Shadow System) Command Description	102
ADDDSTCLGE (Add Distribution Catalog Entry) Command Description	106
ADDDSTLE (Add Distribution List Entry) Command Description	116
ADDDSTQ (Add Distribution Queue) Command Description	119
ADDDSTRTE (Add Distribution Route) Command Description	123
ADDDSTSYSN (Add Distribution Secondary System Name) Command Description	127
ADDILOAUT (Add Document Library Object Authority) Command Description	129
ADDEMLCFGE (Add Emulation Configuration Entry) Command Description	131
ADDENVVAR (Add Environment Variable) Command Description	135
ADDEXITPGM (Add Exit Program) Command Description	137
ADDEWCBCDE (Add Extended Wireless Controller Bar Code Entry) Command Description	139
ADDEWCM (Add Extended Wireless Controller Member) Command Description	143
ADDEWCPTCE (Add Extended Wireless Controller PTC Entry) Command Description	145
ADDEWLM (Add Extended Wireless Line Member) Command Description	149
ADDFNNTBLE (Add Font Table Entry) Command Description	152
ADDHDBDLFM (Add Host Database to DataLink File Manager) Command Description	159
ADDIMGCLGE (Add Image Catalog Entry) Command Description	160
ADDICFDEVE (Add Intersystem Communications Function Program Device Entry) Command Description	162
ADDIPSIFC (Add IP over SNA Interface) Command Description	171
ADDIPSLOC (Add IP over SNA Location Entry) Command Description	173
ADDIPSRTE (Add IP over SNA Route) Command Description	176
ADDJOBQE (Add Job Queue Entry) Command Description	178
ADDJOBSCDE (Add Job Schedule Entry) Command Description	181
ADDJOBJS (Add Job using Job Scheduler) Command Description	187
ADDLIBLE (Add Library List Entry) Command Description	202
ADDLICCRQA (Add License CRQ Activity) Command Description	203
ADDLICENSE (Add License Key Information) Command Description	209
ADDLNK (Add Link) Command Description	213
ADDLANADPI (Add Local Area Network Adapter Information) Command Description	215
ADDLFM (Add Logical File Member) Command Description	216

ADDMEDIBRM (Add Media Information to BRM) Command Description	220
ADDMLMBRM (Add Media Library Media to BRM) Command Description	220
ADDMEDBRM (Add Media to BRM) Command Description	221
ADDMSGD (Add Message Description) Command Description	222
ADDMFS (Add Mounted File System) Command Description	234
MOUNT (Add Mounted File System) Command	240
ADDNTWAUTE (Add NetWare Authentication Entry) Command Description	240
ADDNETJOB (Add Network Job Entry) Command Description	241
ADDNWSSTGL (Add Network Server Storage Link) Command Description	244
ADDNETTBL (Add Network Table Entry) Command Description	247
ADDNCK (Add Nickname) Command Description	248
ADDNODLE (Add Node List Entry) Command Description	250
ADDOBJCRQA (Add Object CRQ Activity) Command Description	253
ADDOPTCTG (Add Optical Cartridge) Command Description	265
ADDOPTSVR (Add Optical Server) Command Description	266
ADDPEXDFN (Add Performance Explorer Definition) Command Description	267
ADDPEXFTR (Add Performance Explorer Filter) Command Description	283
ADDPFCST (Add Physical File Constraint) Command Description	290
ADDPFM (Add Physical File Member) Command Description	295
ADDPFTRG (Add Physical File Trigger) Command Description	297
ADDTCPPTP (Add Point-to-Point TCP/IP Profile) Command Description	301
ADDPJE (Add Prestart Job Entry) Command Description	307
ADDPRBACNE (Add Problem Action Entry) Command Description	312
ADDPRBSLTE (Add Problem Selection Entry) Command Description	316

Command Descriptions

ADDACC (Add Access Code) Command Description

ADDACC Command syntax diagram

Purpose

The Add Access Code (ADDACC) command defines a new access code to the system. Subsequent use of the new code is allowed when a document is filed either during a document interchange session or when the Add Document Library Authority (ADDDLOAUT), Edit Document Library Object Authority (EDTDLOAUT), Remove Document Library Object Authority (RMVDLOAUT), Grant Access Code Authority (GRTACCAUT), Revoke Access Code Authority (RVKACCAUT), or Remove Access Code (RMVACC) commands are used. The ADDACC command identifies both the added access code and the descriptive text associated with it.

Restriction: This command is shipped with public *EXCLUDE authority. Use of this command is restricted to the security officer, users with *ALLOBJ (all object) authority, and users to whom authority has been granted by using the Grant Object Authority (GRTOBJAUT) command.

Required Parameters

ACC Specifies the access code being added to the system. The access code is a decimal number ranging from 1 through 2047, and it must not be currently defined on the system.

***AVAIL:** The system chooses the next free access code and adds it to the system; the user must specify the text. The access code selected is returned in a completion message.

access-code: Specify a decimal number ranging from 1 through 2047.

TEXT Specifies the user-defined text that briefly describes the access code. More information on this parameter is in commonly used parameters.

'description': Specify no more than 50 characters of text, enclosed in apostrophes.

Examples for ADDACC

Example 1: Adding an Access Code

```
ADDACC ACC(700) TEXT('programmers')
```

This command adds access code 700 to the system. This access code is authorized to all programmers (after the GRTACCAUT command is run). It is used when filing documents to which all programmers may have access.

Example 2: Adding Next Available Access Code

```
ADDACC ACC(*AVAIL)  
TEXT('department managers')
```

This command adds the next available access code to the system. This access code is authorized to all department managers (after the GRTACCAUT command is run). It is used when placing in the document library objects to which all department managers may have access. The system returns a message containing the access code that was being used.

Error messages for ADDACC

*ESCAPE Messages

CPF897B

Mail Log Conversion failed.

CPF9001

Add access code request failed.

CPF9009

System requires file &1 in &2 be journaled.

CPF9845

Error occurred while opening file &1.

CPF9846

Error while processing file &1 in library &2.


CPF9847

Error occurred while closing file &1 in library &2.

ADDALRACNE (Add Alert Action Entry) Command Description

ADDALRACNE Command syntax diagram

Purpose

The Add Alert Action Entry (ADDALRACNE) command allows the user to add an action entry to the specified alert filter. This entry describes the actions that should be taken for an alert that has been assigned to the specified group. More information on alerts is in the Alerts Support  book.

Required Parameters

FILTER

Specifies the qualified name of the filter to which the action entry is added.

The name of the filter can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

filter-name: Specify the name of the filter.

GROUP

Specifies the group name to which the defined actions are to be applied. The group name is assigned from the selection criteria in the filter.

Optional Parameters

LOG Specifies whether the alert is logged.

***NETATR:** The ALRLOGSTS network attribute controls the logging of this alert.

***YES:** The alert is logged.

***NO:** The alert is not logged.

ASNUSER

Specifies the user assigned to the alert.

***NONE:** No user is specified.

assigned-user: Specify a user name.

SEND Specifies the destination to which the alert is to be sent. An alert cannot be sent to the local system or sent multiple times. The system checks for this action when the alert is sent.

***NONE:** The alert is not sent.

Element 1: Network Identifier

***FOCALPT:** Sends the alert to the system focal point. The focal point system is determined at send time.

***NETATR:** The LCLNETID value specified in the system network attributes is used.

network-ID: Specify the network ID of the destination node.

Element 2: Control Point Name

control-point-name: Specify the control point name of the destination system.

SNDDTAQ

Specifies the data queue in which an alert notification record is placed. Keyed data queues are supported.

***NONE:** No data queue is used.

The name of the data queue can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

Element 1: Data Queue Name

data-queue-name: Specify the name of the data queue.

Element 2: Data Queue Key

***NONE:** No key is used on the data queue.

data-queue-key: Specify the data queue key.

GENTRAP

Specifies whether the alert generates an SNMP trap.

***NO:** An SNMP trap is not generated from this alert.

***YES:** An SNMP trap is generated from this alert.

Example for ADDALRACNE

```
ADDALRACNE FILTER(MYLIB/MYFILTER) GROUP(CHICAGO)
LOG(*NETATR) ASNUSER(CHICAGOOPR)
SEND(*FOCALPT)(*NETATR.MILWKEE)
SNDDTAQ(*LIBL/ALERTDTAQ)
```

This command defines the following actions for group CHICAGO:

1. Log the alert based on the ALRLOGSTS network attribute.
2. Send the alert to this system's focal point.
3. Send the alert to the system with control point name MILWKEE and a network id based on the LCLNETID value specified in the system network attributes.
4. Place an alert notification on data queue ALERTDTAQ.
5. Assign the alert to user CHICAGOOPR.

Error messages for ADDALRACNE

***ESCAPE Messages**

CPF2150

Object information function failed.

CPF2151

Operation failed for &2 in &1 type *&3.

CPF812F

Filter damaged.

CPF91DB

Group &4 already exists.

CPF91DE

Filter &1/&2 at maximum size.

CPF91EB

Filter type &3 not correct for this operation.

CPF91EC

Internal processing error occurred.

CPF91E8

Internal processing error occurred.

CPF9802

Not authorized to object &2 in &3.

CPF9803

Cannot allocate object &2 in library &3.

CPF9807

One or more libraries in library list deleted.

CPF9808

Cannot allocate one or more libraries on library list.


CPF9830

Cannot assign library &1.

ADDALRD (Add Alert Description) Command Description

ADDALRD Command syntax diagram

Purpose

The Add Alert Description (ADDALRD) command allows the user to create the description of an alert condition for a particular message identifier. The user provides the SNA generic alert code points that are used to create an alert. More information on alerts is in the Alerts Support  book.

Required Parameters

MSGID

Specifies the message identifier to which this alert description corresponds.

ALRTBL

Specifies the alert table in which this alert description is created.

The name of the alert table can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

alert-table-name: Specify the name of the alert table that is used.

Optional Parameters

ALRTYPE

Specifies the code point (a hexadecimal number) for the alert type.

***NONE:** There is no alert type code point for this alert description.

alert-type-code-point: Specify the alert type code point.

ALRD Specifies the code point for the alert description.

***NONE:** There is no alert description code point for this alert description.

alert-description-code-point: Specify the alert description code point.

PBLCAUSE

Specifies probable causes, which are listed in order of decreasing probability. Up to 99 code points for probable cause can be listed.

***NONE:** There are no probable cause code points for this alert description.

probable-cause-code-point: Specify the probable cause code point.

CAUSE

Specifies user, install, or failure causes. Up to 99 causes can be specified.

***NONE:** There are no cause code points for this alert description.

Element 1: Type of Code Point

***USER:** A user cause code point follows.

***INSTALL:** An install cause code point follows.

***FAILURE:** A failure cause code point follows.

Element 2: Cause Code Point

cause-code-point: Specify the cause code point. Up to three detailed data qualifiers or one product identifier qualifier can be specified for each code point. A detailed data qualifier consists of a detailed data identifier code point and detailed data. Specify *NONE or *NODATA if there is no detailed data.

Element 3: First Detailed Data Identifier

***NONE**: There is no detailed data identifier code point for this cause.

detailed-data-ID: Specify the detailed data identifier code point used to identify the data. Detailed data identifiers can be specified up to three times in each session.

Element 4: Detailed Data for First Identifier

***NODATA**: There is no data for this cause.

detailed-data: Specify up to 40 characters of detail data. A substitution variable from the corresponding message description can be specified and the message data is substituted into the alert description when the alert is created.

Element 5: Product Identifier

***NONE**: There is no product identifier for this cause.

***SNDHDW**: Indicates the sender hardware, which is the iSeries 400 server.

***SNDSFW**: Indicates the sender software specified in the LICPGM keyword of the CRTALRTBL command.

***RSCHDW**: Indicates the failing resource hardware, which is determined by the resource hierarchy in the message description.

Note:

The user can specify either 0 to 3 detailed data qualifiers or one product identifier qualifier, but not both.

ACTION

Specifies a recommended action for a user, install, or failure cause. Up to 99 recommended actions can be listed.

***NONE**: There are no recommended action code points for this alert description.

Element 1: Type of Action Code Point

***USER**: A user cause code point follows.

***INSTALL**: An install cause recommended action code point follows.

***FAILURE**: A failure cause recommended action code point follows.

***UNKNOWN**: A recommended action for a 'cause undetermined' error follows.

Element 2: Action Code Point

action-code-point: Specify the recommended action code point. Up to three detailed data qualifiers or one product identifier qualifier can be specified for each code point. A detailed data qualifier consists of a detailed data ID code point and detailed data. Specify *NONE *NODATA if there is no detailed data.

Element 3: First Detailed Data Identifier

***NONE**: There is no detailed data identifier code point for this action.

detailed-data-ID: Specify the detailed data identifier code point used to identify the data. Detailed data identifiers can be specified up to three times in each session.

Element 4: Detailed Data for First Identifier

***NODATA**: There is no data for this action.

detailed-data: Specify up to 40 characters of detail data. A substitution variable from the corresponding message description can be specified and the message data is substituted into the alert description when the alert is created.

Element 5: Product Identifier

***NONE**: There is no product identifier for this action.

***SNDHDW**: Indicates the sender hardware, which is the iSeries server.

***SNDSFW**: The sender software, specified in the PRDID keyword of the CRTALRTBL command is used.

***RSCHDW**: Indicates that the failing resource hardware, which is determined by the resource hierarchy in the message description is used.

Note:

The user can specify either 0 to 3 detailed data identifiers or one product identifier qualifier, but not both.

Example for ADDALRD

```
ADDALRD MSGID(USR1234) ALRTBL(USER/USRMSG)
ALRTYPE(01) ALRD(3100)
PBLCAUSE(1000 3121)
CAUSE((*USER 6001) (*FAILURE 1000)
(*FAILURE 3121)) ACTION((*USER 1000)
(*FAILIRE 00B0 00A5 'DSPMSG QSYSOPR')
(*FAILURE F0A0 22 '&5')
(*FAILURE 00E1 *NONE *NODATA *NONE
*NODATA *NONE *NODATA *SNDHDW))
```

This command defines three recommended failure actions:

- '00B0', which requires a detailed qualifier. One detailed data qualifier is provided.
- The detailed data identifier code point is '00A5', which identifies the text Command and
- The detailed data 'DSPMSG QSYSOPR'.

Failure recommended action 'F0A0' specifies a message substitution variable ('&5') as the detailed data. When the message 'USR1234' is sent, the message data for variable '&5' is put into the alert for the detailed data.

Failure action X'00E1' references a product identifier; in this case, it is the sending hardware iSeries server. Place holders are needed for the detailed data qualifiers.

Error messages for ADDALRD

***ESCAPE Messages**

CPF1A01

Alert table &1 in &2 cannot be extended.

CPF1A02

Alert code &1 already in alert table &2.

CPF1A03

Alert identifier &1 already in alert table &2.

CPF1A05

Alert table &1 in &2 damaged.

CPF2499

Message identifier &1 not allowed.

CPF7BB0

Alert description already exists.

CPF7BB5

Alert description &1 could not be added to alert table &2 in library &3.

CPF9801

Object &2 in library &3 not found.

CPF9802

Not authorized to object &2 in &3.

CPF9803

Cannot allocate object &2 in library &3.

CPF9807

One or more libraries in library list deleted.

CPF9808

Cannot allocate one or more libraries on library list.

CPF9810

Library &1 not found.

CPF9811

Program &1 in library &2 not found.

CPF9812

File &1 in library &2 not found.

CPF9814

Device &1 not found.

CPF9820

Not authorized to use library &1.

CPF9821

Not authorized to program &1 in library &2.

CPF9822

Not authorized to file &1 in library &2.

CPF9825

Not authorized to device &1.

CPF9830

Cannot assign library &1.

CPF9831

Cannot assign device &1.

CPF9899

Error occurred during processing of command.

ADDALRSLTE (Add Alert Selection Entry) Command Description

ADDALRSLTE Command syntax diagram

Purpose

The Add Alert Selection Entry (ADDALRSLTE) command allows the user to add an alert selection entry to an alert filter. Selection entries are the criteria that categorize a group of alerts. More information on alerts

is in the Alerts Support  book.

FILTER

Specifies the qualified name of the filter being added.

The name of the filter can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

filter-name: Specify the name of the filter.

SELECT

Specifies the comparisons to be made to determine if the alert belongs in the specified group. The selection entry results in a successful match with an alert when the data in the alert satisfies the relationships specified on the SELECT parameter. Up to 10 attribute values can be compared to the alert.

***ANY:** Any alert matches this selection record. Specify the conditions under which an alert matches the selection entry. Each condition must contain the following four elements:

1. One of the logical operators *IF, *AND, or *OR
2. The attribute compared
3. One of the relational operators
4. The attribute value

Element 1: Logical Operator

***IF:** Identifies the first condition that must be satisfied.

***AND:** The conditions on both sides of the *AND must be satisfied.

***OR:** One of the conditions on each side of the *OR must be satisfied.

If there is one set or several sets of conditions, the *IF value must be specified as the first value in the first set of comparison values. If more than one set of conditions are specified, *AND or *OR must be specified as the first value in each set after the first. Each condition must be enclosed in parentheses. *AND is evaluated before *OR.

Element 2: Attribute

***ORIGIN:** Specifies whether the alert is generated or received. The valid values for this attribute are L (Locally generated) or R (Received).

***RSCNAME:** Specifies the name of the failing resource. The value for this attribute must be a 8-character name.

***RSCTYPE:** Specifies the type of the failing resource. The value for this attribute must be a 3-character resource type (for example, TAP or DKT).

***HIERNAME:** Specifies all of the resources in the alert resource hierarchy. The alert resource hierarchy is the list of resources, separated by blanks, displayed on the Work with Alerts

(WRKALR) command detailed data displays. The value for this attribute can be a list of up to 5 resource names separated by a blank, unless the value is used with the *CT relational operator. If the *CT value is used, the selection relation can test to see if the given resource name is found anywhere within the hierarchy. This attribute contains the resource names from the hierarchy only.

***HIERTYPE:** Specifies all of the resource types in the alert resource hierarchy. The resource types match the resource names specified on the *HIERNAME attribute. The value for this attribute can be a list of up to 5 resource types (1 to 3 characters in length) separated by a blank, unless the value is used with the *CT relational operator. If the *CT value is used, the selection relation can test to see if the given resource type is found anywhere within the hierarchy.

***MSGID:** Specifies the message identifier.

***MSGSEV:** Specifies the message severity. This value, >> 00 through 99, represents the severity level of the message (99 is the highest severity level) <<.

***ALRID:** Specifies the alert identifier. The alert identifier is displayed on the Work with Alerts (WRKALR) command detailed data display. The value for this attribute must be an 8-digit hexadecimal number unless it is used with the *CT relational operator. If the *CT operator or a wildcard character is used, the attribute must have an even number of digits up to a maximum of 8. The alert ID may not be a valid comparison for iSeries alerts created after problem analysis.

***ALRTYPE:** Specifies the alert type code point that is in the alert. The value for this attribute is a 2 digit hexadecimal number.

***ALRDSC:** Specifies the alert description code point that is in the alert. The value for this attribute must be an 4-digit hexadecimal number unless it is used with the *CT relational operator. If the *CT operator or a wildcard character is used, the attribute must have an even number of digits up to a maximum of 4.

***PBLCSE:** Specifies the probable cause code point that is in the alert. The value for this attribute must be an 4-digit hexadecimal number unless it is used with the *CT relational operator. If the *CT operator or a wildcard character is used, the attribute must have an even number of digits up to a maximum of 4.

***USRCSE:** Specifies the first user cause code point that is in the alert. The value for this attribute must be an 4-digit hexadecimal number unless it is used with the *CT relational operator. If the *CT operator or a wildcard character is used, the attribute must have an even number of digits up to a maximum of 4.

***INSCSE:** Specifies the first install cause code point that is in the alert. The value for this attribute must be an 4-digit hexadecimal number unless it is used with the *CT relational operator. If the *CT operator or a wildcard character is used, the attribute must have an even number of digits up to a maximum of 4.

***FLRCSE:** Specifies the first failure cause code point that is in the alert. The value for this attribute must be an 4-digit hexadecimal number unless it is used with the *CT relational operator. If the *CT operator or a wildcard character is used, the attribute must have an even number of digits up to a maximum of 4.

***RSCHDW:** Specifies the failing hardware resource information in the alert. This information is displayed on the Work with Alerts (WRKALR) command detailed data displays. Specify a value for this attribute using the following form:


```
'tttt mmm ss-sssssss'  
'tttt mmm ss-sssss'  
'tttt mmm sssssss'  
'tttt mmm sssss'
```

where `tttt` is the machine type, `mmm` is the model number, and `ssssssss` is the serial number. Use this format to match a particular hardware resource or use a part of the hardware value with the `*CT` relational operator to provide a partial match.

***SNDHDW:** Specifies the sending hardware resource information in the alert. This information is displayed on the Work with Alerts (WRKALR) command detailed data displays. Specify a value for this attribute using the following form:

```
'tttt mmm ss-sssssss'  
'tttt mmm ss-sssss'  
'tttt mmm sssssss'  
'tttt mmm sssss'
```

where `tttt` is the machine type, `mmm` is the model number, and `ssssssss` is the serial number. Use this format to match a particular hardware resource or use a part of the hardware value with the `*CT` relational operator to provide a partial match.

***RSCSFW:** Specifies the failing software resource information in the alert. This information is displayed on the Work with Alerts (WRKALR) command detailed data displays. Specify a value for this attribute using the following form:

```
'ppppppp vv rr mm'
```

where `ppppppp` is the licensed program identifier, `vv` is the version number, `rr` is the release number, and `mm` is the modification level. Use this format to match a particular software resource or use a part of the software value with the `*CT` relational operator to provide a partial match.

***SNDSFW:** Specifies the sending software resource information in the alert. This information is displayed on the Work with Alerts (WRKALR) command detailed data displays. Specify a value for this attribute using the following form:

```
'ppppppp vv rr mm'
```

where `ppppppp` is the licensed program identifier, `vv` is the version number, `rr` is the release number, and `mm` is the modification level. Use this format to match a particular software resource or use a part of the software value with the `*CT` relational operator to provide a partial match.

Element 3: Relational Operator

***EQ:** The attribute in element 2 must be equal to the value specified in element 4.

***GT:** The attribute in element 2 must be greater than the value specified in element 4.

***LT:** The attribute in element 2 must be less than the value specified in element 4.

***NE:** The attribute in element 2 must not be equal to the value specified in element 4.

***GE:** The attribute in element 2 must be greater than or equal to the value specified in element 4.

***LE:** The attribute in element 2 must be less than or equal to the value specified in element 4.

***CT:** The attribute in element 2 must contain the value specified in element 4.

Element 4: Attribute Value

attribute-value: Specify the value (a maximum of 60 characters) to be compared with the contents of the specified attribute. The value must be specified in apostrophes if it contains blanks or special characters and must be in character format. If a CL variable is specified for the value, it must be a character variable.

generic-attribute-value*: Specify the generic attribute value. Generic attribute values are only allowed with the *EQ and *NE operator. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk (*) substitutes for any valid characters. A generic name specifies all attributes with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete attribute name. If the complete attribute name is specified, and multiple libraries are searched, multiple attributes can be added only if *ALL or *ALLUSR library values can be specified for the name. See generic names for additional information.

Optional Parameters

SEQNBR

Specifies the sequence number of the alert selection entry. Selection entries are evaluated in order by sequence number.

***GEN**: Allows the system to generate the sequence number. The sequence number will be greater than all previous selection entries.

sequence-number: Specify a number from 1 through 9999.

GROUP

Specifies the group that an alert is assigned to if the alert matches the criteria specified on the SELECT parameter.

***DEFAULT**: The alert is assigned to the *DEFAULT group. The *DEFAULT group is automatically added when a filter is created.

group-name: Specify a group name to which the alert is assigned.

Example for ADDALRSLTE

```
ADDALRSLTE FILTER(MYLIB/MYFILTER)
  SELECT((*IF *RSCNAME *EQ CHICAGO1)
  (*AND *RSCTYPE *EQ CP))
  SEQNBR(*GEN) GROUP(CHICAGO)
```

This command adds selection entry 0010 to the filter MYFILTER in library MYLIB (a 0010 is generated because no entries have been added to the filter). Any alerts that have a resource name of 'CHICAGO1' and a resource type of 'CP' (control point) are assigned to group CHICAGO.

Error messages for ADDALRSLTE

*ESCAPE Messages

CPD91CB

*CT not allowed with numeric values.

CPF2150

Object information function failed.

CPF2151

Operation failed for &2 in &1 type *&3.

CPF812F

Filter damaged.

CPF91DA

Sequence number &4 already exists.

CPF91DE

Filter &1/&2 at maximum size.

CPF91D9

Sequence number cannot be automatically created.

CPF91EA

*IF relationship not in correct position.

CPF91EB

Filter type &3 not correct for this operation.

CPF91EC

Internal processing error occurred.

CPF91E6

Generic values only allowed with *EQ or *NE.

CPF91E7

Character in position &4 not valid in value specified.

CPF91E8

Internal processing error occurred.

CPF9802

Not authorized to object &2 in &3.

CPF9803

Cannot allocate object &2 in library &3.

CPF9807

One or more libraries in library list deleted.

CPF9808

Cannot allocate one or more libraries on library list.

CPF9830

Cannot assign library &1.

ADDAUTLE (Add Authorization List Entry) Command Description

ADDAUTLE Command syntax diagram

Purpose

The Add Authorization List Entry (ADDAUTLE) command allows the user to add entries to an authorization list. An entry consists of a user's name and the authorities associated with that user on the authorization list. Both the authorization list and the user profile must exist. If the specified user is already on the list, a message is issued and the user's authorities on the list are not changed.

The users who can use this command to add users to an authorization list are: the owner of the authorization list, a user with authorization list management (AUTLMGT) authority on the authorization list, or a user with all object (ALLOBJ) authority.

When the ADDAUTLE command is used to add a user to an authorization list, the user must specify the name of the authorization list, a list of authorized users, and a list of authorities specified for the list. Each user on the list is given the authorities specified on the command.

Restrictions:

1. Authorization list management authority allows a user to manage the authorization list and, therefore, to manage the authorities for all objects secured by the list.

2. Only the owner of the list or a user with *ALLOBJ authority can add a user with *AUTLMGT authority.
3. A user with *AUTLMGT authority can add users and give specific authorities only to the *AUTLMGT level.

Required Parameters

AUTL Specifies the name or generic name of the authorization list to which the users are being added. The authorization list must already exist.

authorization-list-name: Specify the name of the authorization list to which the user profile name is added.

generic-authorization-list-name*: Specify the generic name of the authorization list to which the user profile names are being added. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. Generic names provides additional information.

USER Specifies a list of user names to be added to the authorization list. Up to 50 user names can be specified. If a user profile name is already on the authorization list, a message is issued and the user's authorities are not changed.

Optional Parameter

AUT Specifies the authority given to users specified on the USER parameter. Users must have *AUTLMGT authority to manage the authorization list.

***CHANGE**: The user can perform all operations on the object except those limited to the owner or controlled by object existence authority and object management authority. The user can change and perform basic functions on the object. Change authority provides object operational authority and all data authority. If the user profile name is an authorization list, the user cannot add, change, or remove user profile names.

***ALL**: The user can perform all operations except those limited to the owner or controlled by authorization list management authority. The user can control the object's existence, specify the security for the object, change the object, and perform basic functions on the object. The user also can change ownership of the object.

***USE**: The user can perform basic operations on the object, such as running a program or reading a file. The user cannot change the object. *USE authority provides object operational authority, read authority, and execute authority.

***AUTLMGT**: Authorization list management authority provides the authority to add users to the authorization list, to change users' authorities on the authorization list, or to remove users from the authorization list, to rename an authorization list, or to create a duplicate authorization list.

***OBJALTER**: Object alter authority provides the authority needed to alter the attributes of an object. If the user has this authority on a database file, the user can add and remove triggers, add and remove referential and unique constraints, and change the attributes of the database file. If the user has this authority on an SQL package, the user can change the attributes of the SQL package. This authority is currently only used for database files and SQL packages.

***OBJEXIST**: Object existence authority provides the authority to control an object's existence and ownership. These authorities are necessary for users who want to delete an object, free storage for an object, perform save and restore operations for an object, or transfer ownership of an object. A user with special save system (*SAVSYS) authority does not need object existence authority to save or restore objects. Object existence authority is required to create an object that has been named by an authority holder.

***OBJMGT:** Object management authority provides the authority to specify the security for an object, to move or rename an object, and to add members to database files.

***OBJOPR:** Object operational authority provides authority to look at the description of an object and to use the object as determined by the data authorities held by the user.

***OBJREF:** Object reference authority provides the authority needed to reference an object from another object such that operations on that object may be restricted by the other object. If the user has this authority on a physical file, the user can add referential constraints in which the physical file is the parent. This authority is currently only used for database files.

***ADD:** Add authority provides the authority to add entries to an object (for example, job entries to a queue or records to a file).

***DLT:** Delete authority allows the user to remove entries from an object, for example, remove messages from a message queue or records from a file.

***EXECUTE:** Execute authority provides the authority needed to run a program or locate an object in a library or directory.

***READ:** Read authority provides the authority needed to show the contents of an object.

***UPD:** Update authority provides the authority needed to change the entries in an object.

Single Value

***EXCLUDE:** The user cannot access the object.

Example for ADDAUTLE

```
ADDAUTLE AUTL(PAYROLL) USER(TOM)
AUT(*ALL *AUTLMGT)
```

This command adds user TOM to the PAYROLL authorization list and gives him all authority to the objects secured by the authorization list. TOM also has authority to manage the authorization list.

Error messages for ADDAUTLE

***ESCAPE Messages**

CPF22AA

Only *AUTLMGT authority can be specified with *ALL authority.

CPF22AB

Only *AUTLMGT can be specified with *CHANGE authority.

CPF22AC

Only *AUTLMGT authority can be specified with *USE authority.

CPF2253

No objects found for &1 in library &2.

CPF2280

*PUBLIC is always on authorization list, cannot be added.

CPF2281

The users specified do not exist on the system.

CPF2282

&1 errors adding users, &2 authorization lists processed.

CPF2283

Authorization list &1 does not exist.

CPF2284

Not authorized to change authorization list &1.

CPF2289

Unable to allocate authorization list &1.

CPF2290

*EXCLUDE cannot be specified with another authority.

ADDAJE (Add Autostart Job Entry) Command Description

ADDAJE Command syntax diagram

Purpose

The Add Autostart Job Entry (ADDAJE) command adds a job entry which becomes active when the subsystem is started (the subsystem specified on the subsystem description). The user must first specify this command, then start the subsystem. The job entry identifies the job and its associated job description to the subsystem. These jobs are automatically started when the subsystem is started.

Restriction: To use this command, the user must have object operational and object management authorities for the specified autostart job entry.

Required Parameters

SBSD Specifies the qualified name of the subsystem description where the job entry that automatically starts is added.

The name of the subsystem description can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

subsystem-description-name: Specify the name of the subsystem description where the job entry is added.

JOB Specifies the simple name of the job that is automatically started when a subsystem is started by using the subsystem description specified in the SBSB parameter.

Optional Parameters

JOBD Specifies the name of the job description used for the job that is started by this autostart job entry. If the job description does not exist when the entry is added, a library qualifier must be specified because the qualified job description name is retained in the subsystem description.

***SBSB:** The job description that has the same qualified name as the subsystem description, specified by the SBSB parameter, is used for the job being started.

The name of the job description can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

job-description-name: Specify the name of the job description that is used for the job started by this job entry.

Example for ADDAJE

```
ADDAJE  SBSDBD(ACCTLIB/ACCTINT)
        JOB(ACCTINIT)  JOBD(ACCTLIB/INITSBS)
```

This command adds the job ACCTINIT as a job entry that starts automatically to the subsystem description ACCTINT in the library ACCTLIB. In this case, the job that starts automatically might be used to perform certain routines whenever the subsystem ACCTINT is started. When the subsystem is started, the job description INITSBS in ACCTLIB is used to obtain the attributes for this job and a job named ACCTINIT is automatically started in the subsystem.

Error messages for ADDAJE

*ESCAPE Messages

CPF1619

Subsystem description &1 in library &2 damaged.

CPF1697

Subsystem description &1 not changed.

ADDBNDDIRE (Add Binding Directory Entry) Command Description

ADDBNDDIRE Command syntax diagram

Purpose

The Add Binding Directory Entry (ADDBNDDIRE) command adds an entry to the binding directory.

Restrictions:

1. You must have *USE authority for the library where the binding directory is being updated.
2. You must have object operational and *ADD authority to the binding directory.
3. You must have *EXECUTE authority to the specified library when using generic processing.

Required Parameters

BNDDIR

Specifies the binding directory to which an entry is added.

The name of the binding directory can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

library-name: Specify the name of the library to be searched.

binding-directory-name: Specify the name of the binding directory to be updated.

OBJ Specifies the object name to be added to the binding directory.

Element 1: Name of the Object to be Added

The name of the binding directory can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

library-name: Specify the name of the library to be searched.

***ALL:** All objects of the specified type residing in the specified library are added.

object-name: Specify the object to be added.

generic-object-name:* Specify the generic name of the object. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. See generic names for additional information.

Element 2: Type of Object to be Added

***SRVPGM:** Indicates the object to be added is a service program.

***MODULE:** Indicates the object to be added is a module.

POSITION

Specifies the position in the binding directory where the list of objects is added.

***LAST:** The list of objects is added to the end of the binding directory entries.

***FIRST:** The list of objects is inserted prior to the first binding directory entry.

Element 1: Object Position

***AFTER:** The list of objects is added to the binding directory after the binding directory entry specified on this parameter. The entry specified must currently exist in the binding directory.

***BEFORE:** The list of objects is added to the binding directory before the binding directory entry specified on this parameter. The entry specified must currently exist in the binding directory.

***REPLACE:** The object specified on the OBJ parameter replaces the binding directory entry specified on this parameter. The entry specified must currently exist in the binding directory. Only one entry can be specified on the OBJ parameter.

Element 2: Binding Directory Entry Name

The name of the binding directory can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

library-name: Specify the name of the library to be searched.

binding-directory-entry-name: Specify a binding directory entry name that exists in the specified binding directory.

Element 3: Binding Directory Entry Type

***SRVPGM:** Indicates the directory entry is a service program.

***MODULE:** Indicates the directory entry is a module.

Examples for ADDBNDDIRE

Example 1:

```
ADDBNDDIRE  BNDDIR(TESTBNDDIR)
            OBJ((TESTOBJ))  POSITION(*LAST)
```

This command adds a binding directory entry for service program TESTOBJ in library *LIBL to the end of the list of binding directory entries found in the binding directory TESTBNDDIR.

Example 2:

```
ADDBNDDIRE  BNDDIR(TESTBNDDIR)
            OBJ((TESTLIB/TESTOBJ *MODULE))  POSITION(*FIRST)
```

This command adds a binding directory entry for module TESTOBJ in library TESTLIB to the beginning of the binding directory entries found in the binding directory TESTBNDDIR.

Example 3:

```
ADDBNDDIRE  BNDDIR(TESTBNDDIR)
            OBJ((TESTLIB/TESTOBJ *MODULE) (TESTOBJ2))
            POSITION(*FIRST)
```

This command adds a binding directory entry for module TESTOBJ in library TESTLIB followed by an entry for service program TESTOBJ2 in the library list to the beginning of the binding directory entries found in the binding directory TESTBNDDIR.

Example 4:

```
ADDBNDDIRE  BNDDIR(TESTBNDDIR)
            OBJ((TESTLIB/TESTOBJ *MODULE) (TESTOBJ2))
            POSITION(*BEFORE  TESTMOD *MODULE)
```

This command adds a binding directory entry for module TESTOBJ in library TESTLIB, followed by an entry for service program TESTOBJ2 in library *LIBL prior to the binding directory entry for module TESTMOD in library *LIBL found in the binding directory TESTBNDDIR.

The binding directory entry for module TESTMOD in library *LIBL must be found in the binding directory TESTBNDDIR for this operation to be successful.

Error messages for ADDBNDDIRE

*ESCAPE Messages

CPF5D01

Binding directory &1 in library &2 is not usable.

CPF5D09

Object &2/&1 type &3 was not found in binding directory &4 in library &5.

CPF980F

Binding directory &1 in library &2 not found.

CPF9801

Object &2 in library &3 not found.

CPF9802

Not authorized to object &2 in &3.

CPF9803

Cannot allocate object &2 in library &3.

CPF9820

Not authorized to use library &1.

CPF9830

Cannot assign library &1.

ADDBKP (Add Breakpoint) Command Description

ADDBKP Command syntax diagram

Purpose

The Add Breakpoint (ADDBKP) command sets up to 10 breakpoints in a program.

A breakpoint is a location in a program where processing stops and control is given to the user or to a specified program. The breakpoint is set when a statement number or label of a command or machine instruction is specified. The program is stopped just before processing begins on the statement (or machine instruction) on which the breakpoint is set.

This command shows the values of certain program variables when a breakpoint in the program is reached. As many as 10 variables per breakpoint can be specified, and as many as 10 breakpoints per command can be set. However, the same program variables apply to every breakpoint specified in the command. Different ADDBKP commands must be used to specify different sets of variables for each breakpoint.

This command specifies conditional breakpoints in which the program is stopped when a condition is true. This condition involves two program variables or one program variable and a constant. When using conditional breakpoints, it is possible to stop the program when a program variable becomes a certain value.

A conditional breakpoint can also be specified by specifying a skip value. The program does not stop until the breakpoint statements have been processed as many times as the skip number indicates. After that, the breakpoint causes the program to stop.

When a breakpoint is reached in the interactive debugging environment, a display is shown to the user that identifies which breakpoint has been reached and (optionally) the values of the specified program variables when the program is stopped. A message is also written to the job log when the breakpoint is reached. From the display, the user presses F10 to show the command entry display, presses F3 to exit the display and cancel the program, or presses the Enter key to allow the program to continue running.

When a breakpoint is reached in the batch debugging environment, the breakpoint information is written to a printer file and, optionally, another program is called to take action on the breakpoint condition. The name of the called program is specified in the BKPPGM parameter.

When an interactive job is debugging another job, and a breakpoint is reached in the debugged job, a breakpoint display is shown. This display appears in the debugging job, interrupting what was previously being shown. The user must press the Enter key, allowing the stopped program to continue, before returning to the previous display.

Restrictions:

1. This command is valid only in the debug mode. For more information on how to start the debug mode, refer to the description of the STRDBG (Start Debug) command.
2. This command cannot be used if the user is servicing another job, and that job is on a job queue, or is being held, suspended, or ended.
3. This command cannot be used to add breakpoints to a bound program.

Required Parameter

STMT Specifies the statement identifiers of up to 10 statements or machine instructions in the program at which breakpoints are set. The program stops before processing a statement specified as a breakpoint.

The list can contain up to 10 identifiers (statement numbers, program labels, or machine instruction numbers) that are valid for the program specified by the PGM parameter. At least one identifier is needed. If a machine instruction number is specified, a slash must be placed in front of the number and both the slash and the number must be enclosed in apostrophes; for example, STMT ('/21').

In high-level language programs, different statements and/or labels can be mapped to the same internal instruction. This happens when several statements that do not operate on variables directly (such as DO, END, and comments) follow one another in a program. The intermediate representation of a program list is used to determine which statements can be mapped to the same instruction.

Because different statements can be mapped to the same instruction, adding a breakpoint can redefine a previous breakpoint that was added for a different statement. When this occurs, the new breakpoint replaces the previously defined breakpoint.

Optional Parameters

PGMVAR

Specifies the names of up to 10 program variables shown that are in a high-level language or a machine instruction program. The name and value of each program variable is shown when any of the breakpoints specified in the STMT parameter are reached. During a run, the program stops *before* processing a statement specified as a breakpoint.

Note:

In some high-level languages such as RPG, variables that are declared but not referred to in the program cannot be specified on the PGMVAR parameter.

***NONE:** No program variables are shown for any of the breakpoints specified.

Element 1: Program Variables

***CHAR:** This special value is specified instead of a variable name if a basing pointer is also specified. This special value displays a character view of a pointer to be shown without the use of a based variable.

'program-variable': Specify the names of up to 10 program variables, separated by blanks, shown when a breakpoint is reached. The names must be enclosed in apostrophes if they contain special characters. For example, a CL variable, &VAR, must be specified as PGMVAR('&VAR').

If the program variable is an array, the subscripts representing an element in the array can be specified as a breakpoint. If an array name is specified without any subscripts, all of the array elements are recorded. A single-dimensional cross-section can also be specified. Up to 132 characters can be specified for this program variable entry. This includes any qualifiers, subscripts, embedded blanks, parentheses, and commas. It does not include the enclosing apostrophes when special characters are used.

An integer, a machine-interface object-definition-table-vector (MI ODV) number, asterisk (single-dimensional cross-section), or a numeric variable name can be specified for a subscript. For more information on testing and debugging at machine interface level and on the

program-variable value, refer to the CL Programming  book. Some examples follow:

```
PGMVAR(A)
PGMVAR('A(2,B)')
PGMVAR('B(I1,*,I3)')
PGMVAR('VAR1 OF A(I,J IN B)')
```

Element 2: Basing Pointers

'basing-pointer': Specify up to five basing pointers for the program variable being shown. In some languages, the program variable can be based on a pointer variable. Each basing pointer name must be enclosed in apostrophes if it contains special characters.

If the basing-pointer is an array, the subscripts representing an element in the array must be specified. Up to 132 characters can be specified for a basing-pointer name. This includes any qualification, subscripts, embedded blanks, parentheses, and commas. It does not include the enclosing apostrophes when special characters are used. An integer, an MI ODV number, or a numeric variable name can be specified for a subscript. For more information on the basing-pointer value, refer to parameter values used for testing and debugging. Some examples are:

```
PGMVAR(('VAR1(B,5)' 'PTR2(C,P2)'))
PGMVAR((VAR2 (BASEPTRA BASEPTRB)))
```

START

Specifies, for string variables only, the starting position in the string from which its value is shown when the breakpoint is reached. If more than one string variable is specified in the PGMVAR parameter, the same starting position value is used for each one. For a bit string, the value specifies the starting bit position; for a character string, the value specifies the starting character position.

For conditional breakpoints, the START parameter also specifies the start in the string where the comparison is made.

1: The scope of the open data path (ODP) is the job in which the program occurs. If the job is multi-threaded, only those opens from the same thread can share this ODP.

starting-position: Specify the first position of the program variable being shown.

The START value specified must not be larger than the maximum string length for any variable specified, except that START(1) is allowed if the maximum length for a string is 0. The LEN value,

plus the START position minus one, must not be greater than the maximum string length. These checks are made for each string variable specified in the PGMVAR parameter.

LEN Specifies, for string variables only, the length of the string shown when the breakpoint is reached, starting at the position specified by the START parameter. If more than one string variable is specified in the PGMVAR parameter, the same value is used for each one. For a bit string, the value specifies the number of bits shown; for a character string, the value specifies the number of characters shown.

For conditional breakpoints, the LEN parameter also specifies the length of the string where the comparison is made.

***DCL:** The string variable is shown to the end of the string or for a value of 200 bytes, whichever is less. If the string variable has a maximum length of 0, only LEN(*DCL) is allowed.

displayed-length: Specify the length of the data shown. The length (as well as the combination of START and LEN) must be no greater than the length of the shortest string specified in the PGMVAR parameter.

OUTFMT

Specifies the format in which the objects are shown.

***CHAR:** Variables are shown in character form.

***HEX:** Variables are shown in both character format and hexadecimal format.

SKIP Specifies the number of times the statement or statements on the STMT parameter must be processed before the program is stopped.

0: The program stops immediately when the statement or statements on the STMT parameter are processed. No skipping of breakpoints is done.

skip-value: Specify the number of times the statements on the STMT parameter must be processed before the program is stopped. If there is more than one statement specified, each statement will have its own independent skip value. There is a separate skip count for each statement.

BKPCOND

Specifies a defined condition that must be true before the program is stopped. The condition is tested before any statement on the STMT parameter is processed. If the condition is false, the breakpoint does not stop the program. If the condition is true, the program is stopped.

***NONE:** No breakpoint condition is specified.

Element 1: Variable

variable: Specify a variable to be used in the breakpoint condition. *PGMVAR1 indicates the first variable, *PGMVAR2 the second, and so on. Only numeric, character, or bit variables can be specified.

Element 2: Operator

operator: Specify the type of comparison to be done for a conditional breakpoint. The following comparisons are allowed:

- *EQ - equal to
- *NE - not equal to
- *GT - greater than
- *LT - less than
- *GE - greater than or equal to
- *NL - not less than (same as *GE)
- *LE - less than or equal to
- *NG - not greater than (same as *LE)
- *CT - contains

The *CT operator compares character strings to determine whether one character string contains one or more occurrences of another character string. This comparison is for an exact match, and it is case sensitive.

Element 3: Comparing a Constant or Variable

compare-value: Specify a constant or another variable to compare with the variable. If a constant is specified, it must be the same type as the variable. If the variable is numeric, the constant must be a number. If the variable is a bit, the constant must be a string containing only 1's and 0's. If the variable is a character, the compare value is treated as a character string, even if a number is specified.

If another program variable is specified, it is compared with the variable. The variables must be of the same type. If the variables are numeric, they must both be floating point or not floating point. For example, a packed number cannot be compared with a floating point number.

When comparing two non-floating point variables, or a non-floating point variable and a constant, the total number of digits needed to represent them must not exceed 31. For example, a PACKED(24,2) and a PACKED(24,20) cannot be compared. The first variable requires 22 digits to the left of the decimal point and two digits to the right. The second variable requires four digits to the left of the decimal point and 20 digits to the right. To compare these variables would require a variable with 22 digits to the left of the decimal point and 20 digits to the right. This exceeds the maximum number of allowed digits, 31.

When comparing two character strings, the shorter of the two is padded with blanks. When comparing two bit strings, they must both be of the same length.

The SKIP and BKPCOND parameters can be used together. In this case, the breakpoint condition is not evaluated until the breakpoint has skipped the number of times specified by SKIP. After that, the breakpoint condition is evaluated and the program stops if the condition is true.

BKPPGM

Specifies the qualified name of the user-supplied program (if any) to call when a breakpoint is reached in the program specified by the PGM parameter. When the program specified on the BKPPGM parameter is called, it is passed four parameters that identify: the program name, the recursion level, the high-level language statement identifier, and the machine instruction number at which the breakpoint occurred. Those four parameters have the following formats:

1. Program name (10 bytes). The name of the program in which the breakpoint was reached.
2. Recursion level (5 bytes). The recursion level number of the program in which the breakpoint was reached. This value is a 1- to 5-digit number that is padded on the right with blanks.
3. Statement identifier (10 bytes). The high-level language program statement identifier that was reached. This statement identifier is the statement identifier specified in the Add Breakpoint (ADDBKP) command that defined the breakpoint. If a machine instruction number was used to specify the breakpoint, this parameter contains a slash (/) followed by a 4-digit hexadecimal machine instruction number.
4. Instruction number (5 bytes). The machine instruction number that corresponds to the high-level language statement at which the breakpoint was reached. No slash appears in front of this machine instruction number. It consists of 1 to 4 hexadecimal characters that represent the MI instruction number, followed by one or more blanks. If a machine instruction number is passed in the third parameter, the numbers in the third and fourth parameters are the same.

All the parameter values are left-adjusted and padded with blanks. When the called program returns, the program being debugged continues processing, with the statement with the breakpoint on it.

***NONE:** No breakpoint-handling program is called when any breakpoint specified in this ADDBKP command is reached in the batch environment. The interrupted program continues processing.

The name of the program can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

program-name: Specify the name of the user-supplied program to call if any of the breakpoints on this command are reached while debugging in a batch environment. The program specified should not be the same as the program specified in the PGM parameter. If they are the same, the results are unpredictable. After the called program runs, it returns control to the interrupted program, which continues processing.

PGM Specifies the name of the program to which the breakpoints are added.

***DFTPGM:** The breakpoints are added to the program currently specified as the default program in debug mode.

program-name: Specify the name of the program to which the breakpoints are added. The program must already be in debug mode.

Examples for ADDBKP

Example 1: Adding Breakpoints in Debug Mode

```
ADDBKP STMT(150 RTN1 205)
      PGMVAR('&TEMP' '&INREC')
```

This command establishes breakpoints at CL statement numbers 150 and 205 and at the label RTN1 for the default program in debug mode. When any of these breakpoints is reached, the CL variables &TEMP and &INREC are automatically shown. Note that the CL variables must include a leading ampersand (&) and be specified within apostrophes.

Example 2: Adding Breakpoints to HLL Program

```
ADDBKP STMT(100) PGMVAR('AMOUNT(200)')
      PGM(MYPROG)
```

Assume in this example that MYPROG is a high-level language program being debugged in an interactive environment and that the program variable AMOUNT is a 250-element array in MYPROG. This command adds a breakpoint to statement 100 in MYPROG. When MYPROG is started, the program stops processing at statement 100, and the value of the 200th element of the AMOUNT array is shown. If AMOUNT had been specified without a subscript, all of the array elements would have been shown.

Example 3: Program Stops After Processing Statement 10 Times

```
ADDBKP STMT(10) SKIP(1000)
```

This command causes the default program to stop when statement 10 is processed 1000 times (the breakpoint is skipped 1000 times).

Example 4: Program Stops After Processing Multiple Statements

```
ADDBKP STMT(10 20 30) SKIP(50)
```

This command causes the default program to stop when statements 10, 20, and 30 are processed 50 times.

Example 5: Conditional Breakpoint

```
ADDBKP STMT(10) PGMVAR(X) BKPCOND(PGMVAR1 *EQ 5)
```

This command stops the default program at statement 10 when variable X is equal to five.

Example 6: Conditional Breakpoint

```
ADDBKP STMT(20) PGMVAR((S1) (S2)) SKIP(100)
BKPCOND(*PGMVAR1 *CT *PGMVAR2)
```

This command stops after statement 20 has been processed 100 times, and then only if the character string S2 occurs in the character string S1.

Error messages for ADDBKP

*ESCAPE Messages

CPF1999

Errors occurred on command.



ADDCCSCLT (Add Change Control Server Client) Command Description

Note: To use this command, you must have the 5722-MG1 (Managed System Services for iSeries) licensed program installed.

ADDCCSCLT Command syntax diagram

Purpose

The Add Change Control Server Client (ADDCCSCLT) command allows the user to define a client to be recognized by the Change Control Server. This command should be used to define the first client to the change control server.

Required Parameters

CLIENT

Specifies the name of the client being added.

client-name: Specify a maximum of 63 characters for the name of the client. The name of the client must be precisely specified. Embedded blanks are not allowed. If you specify the name of an existing client, the command is rejected. The following characters are not allowed: asterisk (*), question mark (?), and slash (/).

CPNAME

The control point name for this client. This is the control point part of the SNA distribution services address. NetView/DM refers to this field as the target address.

control-point-name: Specify the APPN control point name of the client being added. A maximum of 8 characters can be used in this field. Both letters and numbers are allowed. Embedded blanks are not allowed. If any lowercase characters are specified, the system changes them to and stores them as uppercase characters.

Optional Parameters

CLTMODE

The mode in which the change control client is configured. The possible values are:

***PUSH:** The change control client is configured with push mode. Change control and distribution operations on a change control client that is configured with push mode are controlled from a change control server in the network. A push mode change control client cannot intervene in the operations performed on it. This mode is allowed when the value of the Client Type (CLTTYPER) parameter is *CLIENT, *SERVER, or *SINGLE.

***FOCAL:** The change control client is configured as a focal point. A change control client configured in this mode receives change control reports from change control clients for which it is defined as the focal point. Reports are routed to the focal point even if operations are initiated from another change control client in the network. A focal point can be considered a focal point from a remote change control client. This value is valid when the value of the Client Type (CLTTYPER) parameter is *SERVER or *SINGLE. When the change control client is configured as a focal point, it is recognized as a remote change control client. Only one client can be identified as the focal point.

***PULL:** The change control client is configured with pull mode. Change control operations on a pull mode change control client can be controlled either by the change control client itself, or by a change control server in the network. This mode is allowed when the value of the Client Type (CLTTYPER) parameter is *CLIENT, *SERVER, or *SINGLE.

***MANAGER:** This change control client is configured as a manager. A change control client that is configured in manager mode can perform change control operations on any change control client inside its domain. It also receives change control reports from those change control clients for which it is defined as manager. This mode is allowed when the value of the Client Type (CLTTYPER) parameter is *SERVER or *SINGLE. When the change control client is configured as a manager, it is recognized as a remote change control client.

***NOMODE:** The change control client is a user interface only. This mode must be used when the value of the Client Type (CLTTYPER) parameter is *UI (user interface).

CLTTYPER

The type of client that you are defining.

***CLIENT:** The change control client is configured as a client type. A change control client is of client type when it is working in conjunction with a change control server and has the Software Distribution Client product installed. All local or remote change control clients must be configured. Remote change control clients must be configured if they are to send or receive files or change control requests. Change control operations can be performed on remote change control clients if the remote administration product option is installed on your system. You do not need to configure a remote change control client to receive distributed files or change control commands from it. Up to 2024 local change control clients can be defined for a change control server. This value is only valid when CLTMODE(*PUSH) or CLTMODE(*PULL) is specified.

***SERVER:** The change control client is configured as a server type. Change control clients that have the Software Distribution Server option installed are configured as servers. The Software Distribution catalog resides at a change control server, and change control and distribution operations for a change control domain are initiated from them. A change control server can initiate change control operations to remote change control clients if the Remote Administration product option is installed on it. This value is valid for CLTMODE(*PUSH), CLTMODE(*PULL), CLTMODE(*MANAGER), and CLTMODE(*FOCAL).

***SINGLE:** The change control client is configured as single-node type. Change control clients that are running Software Distribution and are configured as a base system can be configured as single-node change control clients. Single-node change control clients can be used as preparation sites for software or as focal points to received reports of change control operations. NetView/DM

for MVS nodes must be defined as single-node change control clients. This value is valid for CLTMODE(*PUSH), CLTMODE(*PULL), CLTMODE(*MANAGER), and CLTMODE(*FOCAL).

***UI:** The change control client is configured as user-interface type. These change control clients can only be used to run the Software Distribution user interfaces. This type of change control client is useful when you have an environment with more than one change control server. It allows an administrator to access all change control servers from the same change control client either to perform administrative tasks or to schedule distributions to change control clients. A workstation configured as a user-interface change control client is used to initiate change control on other change control clients, or to request distributions to and from the change control server. User interface change control clients cannot receive change control instructions from a change control client. This value is valid when CLTMODE(*NOMODE) is specified.

SVRNAME

Specifies the name of the change control server that the remote change control client is connected to. This parameter is required if the change control client is remote or if CLTTYTYPE(*CLIENT) is specified.

***DFT:** The name of the change control server to which the change control client is physically connected.

server-name: The name of the change control server to which the change control client is connected. If the name you specify is different from the name of the local server, the client is configured as a remote client.

The server name can be up to 63 characters. Valid server names consist of uppercase letters A through Z, numbers 0 through 9, and special characters: at sign (@), dollar sign (\$), and the number sign (#). The asterisk (*), question mark (?), and slash (/) characters are not allowed.

DMNID

Specifies the change control server domain identifier of the change control client being added. This parameter is not valid when CLTTYTYPE(*UI) is specified and the change control client is remote, or when the change control client is local. This parameter is required if CLTMODE(*MANAGER) or CLTMODE(*FOCAL) is specified. The possible values are:

***SVRCPNAME:** The control point name of the client. If CLTTYTYPE(*CLIENT) is specified, the domain identifier is set to the control point name of the change control server to which the change control client is connected. If CLTTYTYPE(*SERVER) or CLTTYTYPE(*SINGLE) is specified, the domain identifier is set to the value specified in the control point name (CPNAME) parameter.

domain-identifier: Specifies the change control server domain identifier of the change control client being added. The domain identifier can be a maximum of 8 characters. Valid domain identifiers consist of uppercase letters A through Z, numbers 0 through 9, and special characters: at sign (@), dollar sign (\$), and the number sign (#).

OPSYSTYPE

Defines the operating system type of the client being defined. Some of the products or names listed may be trademarks or service marks of other companies.

***OS/2:** Client running NetView Distribution Manager Agent/2.

***AIX:** Client running NetView Distribution Management Agent/6000.

***DOS:** Client running NetView Distribution Management Agent for DOS.

***HPUX:** Client running NetView Distribution Management Agent for HP-UX.

***IRIX:** Client running NetView Distribution Manager Agent for IRIX.

***MAC:** Client running NetView Distribution Management Agent for MAC.

***MVS:** Client running NetView Distribution Manager Agent for MVS.

***NCR:** Client running NetView Distribution Management Agent for NCR.

***NETWARE:** Client running NetView Distribution Manager Agent for Netware.

***OS/2:** Client running NetView Distribution Manager Agent/2.

***OS400:** Client running Managed System Services for iSeries.

***SCO:** Client running NetView Distribution Management Agent for SCO.

***SINIX:** Client running NetView Distribution Management Agent for SINIX. SINIX is a product of Siemens Nixdorf company.

***SOLARIS:** Client running NetView Distribution Management Agent for Sun Solaris. Sun Solaris is a SunSoft product and is a trademark of SUN Microsystems, Incorporated.

***SUNOS:** Client running NetView Distribution Management Agent for SunOS. SunOS is a trademark of SUN Microsystems, Incorporated.

***WINDOWS:** Client running NetView Distribution Management Agent for Windows.

***WINDOWS95:** Client running NetView Distribution Manager Agent for Windows 95.

***WINDOWSNT:** Client running NetView Distribution Management Agent for Windows/NT. Windows NT is a trademark of Microsoft Corporation.

TEXT A description of the client being defined.

***NONE:** No text is specified.

description: Specify a maximum of 59 characters for the description of the client.

CCPERIOD

Specifies a period of time allocated for change control requests. This parameter is not valid when CLTTYPE(*UI) is specified. This parameter is ignored when the change control client is remote. The following values can be coded in this parameter, which contains a list of two elements:

Element 1: Starting Time

One of the following is used to specify the starting time at which change control operations can be performed on the client.

***FIRST:** The change control operations can start at first time of day. For example, you can specify 24:01.

start-time: Specify the starting time at which change control operations can be started. Specify the time using hours and minutes.

The time can be specified as 4 or 6 digits (hhmm or hhmmss, where hh = hours, mm = minutes, and ss = seconds). Seconds are optional. The time can be specified with or without a time separator, such as a colon (:). With a time separator, specify a string of 5 or 8 digits (hh:mm or hh:mm:ss).

Element 2: Ending Time

One of the following is used to specify the ending time at which change control operations will stop on the client.

***LAST:** The change control operations can stop at last time of day. For example, you can specify 23:59.

stop-time: Specify the ending time at which change control operations will stop. Specify the time using hours and minutes.

The time can be specified as 4 or 6 digits (hhmm or hhmmss, where hh = hours, mm = minutes, and ss = seconds). Seconds are optional. The time can be specified with or without a time separator, such as a colon (:). With a time separator, specify a string of 5 or 8 digits (hh:mm or hh:mm:ss).

DSTPERIOD

Specifies a period of time allocated for distribution requests. This parameter is not valid when GLTTYPER(*UI) is specified. This parameter is ignored when the change control client is remote. The following values can be coded in this parameter, which contains a list of two elements:

Element 1: Starting Time

One of the following is used to specify the starting time at which distribution operations can be performed on the client.

***FIRST**: The distribution operations can start at first time of day.

start-time: Specify the starting time at which distribution operations can be started. Specify the time using hours and minutes.

The time can be specified as 4 or 6 digits (hhmm or hhmmss, where hh = hours, mm = minutes, and ss = seconds). Seconds are optional. The time can be specified with or without a time separator, such as a colon (:). With a time separator, specify a string of 5 or 8 digits (hh:mm or hh:mm:ss).

Element 2: Ending Time

One of the following is used to specify the ending time at which distribution operations will stop on the client.

***LAST**: The distribution operations will stop at last time of day.

end-time: Specify the ending time at which distribution operations will stop. Specify the time using hours and minutes.

The time can be specified as 4 or 6 digits (hhmm or hhmmss, where hh = hours, mm = minutes, and ss = seconds). Seconds are optional. The time can be specified with or without a time separator, such as a colon (:). With a time separator, specify a string of 5 or 8 digits (hh:mm or hh:mm:ss).

MSGLOGLVL

Message log level defines the log level that should be used by change control clients before they establish a connection to the change control server and discover the level configured for them. This parameter is ignored when the change control client is remote.

***NORM**: This is the default log level and includes both errors and messages about the main or normal events such as the acceptance of a change management request.

***MIN**: This value should only be selected if there are problems with excess logging on the system. At this level, error logs with the minimal amount of information are produced. Fatal errors are always logged.

***DIAG:** This value should only be selected if a collection of logs is being performed for helping to solve a problem. When you specify this value, detailed information about the change control server process is reported.

CUSTOMER

The name of the customer at the client. This name is used when it is necessary to contact the customer.

***NONE:** No customer name is specified.

customer-name: Specify a maximum of 59 characters for the name of the customer. The customer name can be specified in any format appropriate to the user.

CONTACT

Specifies the name of the person that service personnel on the change control client enterprise should contact.

***NONE:** No contact name is specified.

contact-name: Specify a maximum of 59 characters for the name of the contact. The contact name can be specified in any format appropriate to the user.

ADDRESS

Specifies the street address, city, state, country, and zip code of the change control client enterprise.

***NONE:** No address is specified.

contact-address: Specify a maximum of 199 characters for the address for the specified client. The contact address can be specified in any format appropriate to the user.

MANAGER

Specifies the name of the person who is the manager on the change control client enterprise.

***NONE:** No owning manager is specified.

owning-manager: Specify a maximum of 59 characters for the name of the owning manager. The owning manager can be specified in any format appropriate to the user.

PHONE

Specifies the complete change control client enterprise telephone number. The telephone number should include the area code, exchange, number, and the extension where the service representative can reach the person to contact about system problems.

***NONE:** No telephone number is specified.

telephone-number: Specify up to 31 characters for the telephone number of the client. The telephone number can be specified in any format appropriate to the user.

PASSWORD

This keyword is not available for this release.

ACCKEY

It is the client access key (TAK) value. If this value is specified, the access key must exist in the access key table and must be assigned to the user.

The possible values are:

***NONE:** No client access key is specified.

access-key: Specify an access key to be assigned to the client. This is a four-byte binary field.

HDWPARM

The definition of a hardware parameter that is used for checking hardware prerequisites when installing a change file. Up to 10 hardware parameters can be specified. This parameter is not valid when CLTTYPE(*UI) is specified. This parameter is ignored when the change control client is remote.

***NONE:** No hardware parameters are specified.

Element 1: Parameter

hardware-parameter: Specify a maximum of 80 characters for the hardware parameter.

Element 2: Value

value: Specify a maximum of 80 characters for the value of the hardware parameter.

INSTKN

An installation token to be used by the client during change file installation. An installation token should be the directories that will contain the files and paths that will be installed with the product on the change control client. Up to 10 installation tokens can be specified. This parameter is not valid when CLTTYPE(*UI) is specified.

***NONE:** No installation tokens are specified.

Element 1: Token

installation-token: Specify a maximum of 11 characters for the installation token.

Element 2: Value

value: Specify the a maximum of 49 characters for value of the installation token parameter.

RMTLOCNAME

Specifies the address and address type of the client to be added. This parameter is ignored when the change control client is remote.

Element 1: Client address

The client address can be an internet protocol host name, an internet address, or an SNA network ID and control point name.

The following elements can be used in this parameter:

***CLIENT:** The client address will be set to the value specified in the client name (CLIENT) keyword. If this value is specified then *IP should be specified as the address type. The value specified in the client name (CLIENT) keyword will be validated as an Internet or host name address.

***CPNAME:** The control point name will be set to value specified in the control point name (CPNAME) keyword and the network ID will be set to the control point name assigned to the change control server. If this value is specified then *SNA should be specified as the address type.

remote-location-name: Specify an Internet Protocol host name, an Internet address, or an SNA network ID and control point name.

- A host name must follow these conventions:
 - The first character must be either an English alphabetic character or a numeric character.
 - The last character must be either an English alphabetic character or a numeric character.

- Blanks () are not allowed.
- The special characters, period (.) and minus (-), are allowed.
- Parts of the name separated by periods (.) cannot exceed 63 characters in length.
- Internet address names (in the form nnn.nnn.nnn.nnn) are not allowed.
- Names must be from 1 to 255 characters in length.
- The internet address is specified in the form *nnn.nnn.nnn.nnn*, where *nnn* is a decimal number ranging from 0 through 255. An internet address is not valid if it has a value of all binary ones or all binary zeros for the network identifier (ID) portion or the host name portion of the address. If the internet address is entered from a command line, the address must be enclosed in apostrophes.
- An SNA client address is specified using the format *nnnnnnnn.ccccccc*, where *nnnnnnnn* is the network ID and *ccccccc* is the control point name. If only the control point name, *ccccccc* is specified, the network ID will be set to the control point name assigned to the Change Control Server. The SNA client address can also be specified as *nnnnnnnn.*CPNAME*, where **CPNAME* is the value specified in the control point name (CPNAME) keyword.

Element 2: Address type

The possible address type values are:

***IP:** The remote location name has an Internet Protocol (IP) address type.

***SNA:** The remote location name has a Systems Network Architecture (SNA) address type.

MODE Specify the name of the mode that define the connection with the change control client. This parameter is ignored when the value of the address type keyword on the RMTLOCNAME parameter is something other than *SNA.

The possible mode values are:

***NETATR:** The mode specified in the network attributes is used.

mode-name: Specify a maximum of 8 characters for the mode name. Embedded blanks are not allowed. If any lowercase characters are specified, the system changes them to and stores them as uppercase characters.

Examples for ADDCCSCLT

Example 1: Adding a client using default values

```
ADDCCSCLT CLIENT(KevinAldux) CPNAME(MEXGPL13)
```

Example 2: Adding client KevinAldux using hardware parameters, installation tokens, distribution period and password

```
ADDCCSCLT CLIENT(KevinAldux) CPNAME(MEXGPL13) TEXT('Client no.2') +
DSTPERIOD('12:30:00' '16:43:00') PASSWORD() +
HDWPARAM((MEM 16MB) (DISK 120MB) (KEYB LA)) +
INSTKN((TOKEN1 /USR) (TOKEN2 /USR2))
```

Example 3: Adding client CONNERY using DOS operating system, Manager, phone, password, address, SNA client address and mode

```
ADDCCSCLT CLIENT(CONNERY) CPNAME(MEXGPL16) OPSYSTYPE(*DOS) +
  TEXT('Client no.3') DSTPERIOD(*FIRST '16:43:00') +
  MANAGER(Charles) PHONE('(507)285-2884') PASSWORD() +
  ADDRESS('Patria Avenue 30') +
  HDWPARAM((MEM 16MB) (DISK 120MB) (KEYB LA)) +
  INSTKN((TOKEN1 /USR) (TOKEN2 /USR2)) +
  RMTLOCNAME(APPN.MEXGPL13 *SNA) +
  MODE(blank)
```

Example 4: Adding client Lorena with Windows/NT operating system, an internet address, message log level, hardware parameters, installation tokens, manager name, phone and password

```
ADDCCSCLT CLIENT(Lorena) CPNAME(MEXGPL13) OPSYSTYPE(*WINDOWSNT) +
  TEXT('Client no.4') MSGLOGLVL(*MIN) +
  MANAGER(Charles) PHONE('(507)112-2884') PASSWORD() +
  ADDRESS('Patria Avenue 30') +
  HDWPARAM((MEM 16MB) (DISK 120MB) (KEYB LA)) +
  INSTKN((TOKEN1 /USR) (TOKEN2 /USR2)) +
  RMTLOCNAME('9.18.1.2' *IP)
```

Error messages for ADDCCSCLT

***ESCAPE Messages**

MSS0A1A

Target access key not found.

MSS0A1F

Control point name &2 already exists.

MSS0A2B

Installation token not valid.

MSS0A2D

Client name specified already exists.

MSS0A29

Target access key table not found.

MSS0123

Internal processing error occurred.

MSS2102

Change control client not added.



ADDCRQA (Add Change Request Activity) Command Description

Note: To use this command, you must have the 5722-SM1 (System Manager for iSeries) licensed program installed.

ADDCRQA Command syntax diagram

Purpose

The Add Change Request Activity (ADDCRQA) command adds a change management activity to a change request description. The change management action that is to be performed by the added activity depends on the value specified on the action parameter.

The activity can be conditioned so that it will only run after one or more other activities have completed (successfully or unsuccessfully). The activity may also be scheduled to run at a date and time in the future.

Restrictions:

- You must have *CHANGE authority to the change request description object and *EXECUTE authority to the library.
- If a node list (NODL) value is specified, the node list can only contain entries that have a value of *SNA for the address type.
- Global names or component names must be 64 characters or less including one separator character between tokens.

Note:

If you need to cancel an installation of a NetView Distribution Manager (NetView DM) change file or other non-OS/400 object, you must use the Add Change Management Activity (QNSADDCM) API.

Required Parameters

CRQD Specifies the change request description object name.

The possible library values are:

***LIBL:** All of the libraries in your library list and in the system portions of the job's library list are searched.

***CURLIB:** The current library for the job is used to locate the object.

library-name: Specify that only the library named in this parameter is searched.

change-request-description: Specify the name of the change request description object.

ACTIVITY

Specifies the name of the activity to add to the change request description.

***GEN:** An activity name will be generated. The activity name is of the form QACTxxxxxx where xxxxxx is the first multiple of ten not already being used.

***LAST:** The activity is the last to run in the change request. When *LAST is specified for the activity (ACTIVITY) parameter, the condition (COND) parameter and the start time (STRTIME) parameter cannot be specified. Only one activity named *LAST can exist in the change request description.

activity-name: Specify a 10-character activity name.

ACTION

Specifies the change management action to be performed by the added activity.

***ACP:** Causes the managed system to accept a previously installed object. The activity relinquishes resources at a managed system required to maintain removability of a change. This cancels the removability of a change previously installed in a removable manner. The resources released are, typically, unaltered versions of components affected by the change.

***ACT:** Causes the managed system to activate all previously installed changes. Each managed system implements the activation in its own way. For example, the activity performs an initial program load (IPL) of a managed iSeries server or activates a configuration on a PS/2 and causes the PS/2 to restart.

***DLT:** Requests a delete action at one or more managed systems.

***INS:** Requests the installation of up to seven objects. The objects are treated as corequisites, which means that either all installations succeed or all do not. The activity uses an object and its corequisites, if any, to alter all components necessary to effect the change. The managed system can perform such alteration in a removable manner if required, so that a subsequent request (remove) can return all those components to their original condition prior to the alteration. Also automatic removal or automatic acceptance is possible.

***RMV:** The activity, when run at the managed system, returns (that is, removes) all components previously altered in connection with a change to their condition prior to the installation of the change. This is possible only for changes previously installed in a removable manner.

***RTV:** Retrieves an object identified by its global name, from a managed system or from another central site system, for storage at the central site system. To retrieve an object from more than one system, a global name with an *ANY token is required, so that each retrieved object has a unique global name. Global names with unspecified tokens (*ANY, *HIGHEST, or *LOWEST) are stored in the distribution repository when they are retrieved.

***RUN:** Causes a program or procedure to be run at one or more managed systems.

***SND:** Sends an object from the central site system to one or more managed systems or to another central site system.

***SNDINS:** Sends an object from the central site system to one or more managed systems and installs the object at the managed system

***SNDRUN:** Sends an object from the central site system to one or more managed systems and runs the object at the managed system.

***UNINS:** Removes (that is, uninstalls) a previously installed component from a managed system. The component may have originally been built up from the installation of multiple objects, all of which are removed.

GLBNAME

Specifies a global name, which is a series of tokens that uniquely identify an object in an SNA network. The global name represents the name that is used to locate the appropriate catalog entry on both the central site system and the managed systems. The catalog entry specifies the object that is to be used on that system. For example, if a retrieve action is specified, the global name is used to determine the object that is to be retrieved on the managed system. Also, the global name shows the location where it is to be stored on the central site system.

Special values in a token position indicate how to search for the object. By specifying *ANY in a token position, the token is ignored when searching for the correct object. If multiple objects are found matching the tokens specified, an error is returned.

If an object is sent, the global name must have been previously cataloged so that it is associated with a local object name or associated with an object in the distribution repository. Retrieved objects, for which no catalog entry exists, are placed in the distribution repository.

Special values other than *SERVER, *TARGET, *MDDATE, and *MDTIME in a token position are allowed only for the following actions: *DLT, *RTV, *SND, *SNDRUN, or *SNDINS. The special values allowed for the first token are *NETID, *SERVER, *TARGET, *MDDATE, and *MDTIME.

Element 1: Token 1

***NETID:** The first global name token value is a network ID generated by the command from the network attributes. The network ID is determined by the current value of the LCLNETID network attribute value.

***SERVER:** This token is stored within the change request activity with the value &SERVER, and is replaced by the short name of the change control server when the object is distributed.

***TARGET:** This token is stored within the change request activity with the value &TARGET, and is replaced by the short name of the target when the object is distributed.

***MDDATE:** This token is stored within the change request activity with the value &DATE, and is replaced when distributed by the date that the object was last changed.

***MDTIME:** This token is stored within the change request activity with the value &TIME, and is replaced when distributed by the time that the object was last changed.

global-name-token-1: Specify the first token of the global name. The first token is recommended to be the registered enterprise ID or network ID.

Element 2-10: Token 2-10

***ANY:** Any token value matches when searching for the object where the action is to be performed.

***HIGHEST:** The object with the highest token value has the action performed on it. The token must be ordered. This is useful when a token in a global name is used to indicate a different version of the object and you need to manipulate the object with the highest version level.

***LOWEST:** The object with the lowest token value has the action performed on it. The token must be ordered. This is useful when a token in a global name is used to indicate a different version of the object and you need to manipulate the object with the lowest version level.

***NETID:** The network ID of this system is used. The network ID is determined by the current value of the LCLNETID network attribute value.

***CPNAME:** The control point name of this system is used. The control point name is determined by the current value of the LCLCPNAME network attribute value.

***SERVER:** This token is stored within the change request activity with the value &SERVER, and is replaced by the short name of the change control server when the object is distributed.

***TARGET:** This token is stored within the change request activity with the value &TARGET, and is replaced by the short name of the target when the object is distributed.

***MDDATE:** This token is stored within the change request activity with the value &DATE, and is replaced when distributed by the date that the object was last changed.

***MDTIME:** This token is stored within the change request activity with the value &TIME, and is replaced when distributed by the time that the object was last changed.

global-name-token-n: Specify one of a series of 1 to 16 character tokens that uniquely identify the object on which the action is to be performed. Characters A through Z and 0 through 9 can be used. Other special values (@, #, and \$) can be used for tokens that represent network IDs and system names.

Note:

GLBNAME is valid only when ACTION(*ACT) is not specified.

COMPNAME

Component name, which is the set of global names previous to the REF, UPD or FIX token. The component name is used to identify the installable objects that will be uninstalled. The maximum number of tokens allowed is 7. The component name is used to identify the installable object that must be uninstalled from the managed system.

Note:

COMPNAME is only valid when ACTION(*UNINS) is specified.

Element 1: Token 1

The only special value allowed for the first token is *NETID.

***NETID:** The network ID of this system is used. The network ID is determined by the current value of the LCLNETID network attribute value.

component-name-token-1: One of a series of 1 to 16 character tokens that uniquely identifies the object on which the action is to be performed. Characters A through Z and 0 through 9 can be used. Other special values (@, #, and \$) can be used for tokens that represent network IDs and system names.

Elements 2-7: Tokens 2-7

***NETID:** The network ID of this system is used. The network ID is determined by the current value of the LCLNETID network attribute value.

***CPNAME:** The control point name of this system is used. The network ID is determined by the current value of the LCLCPNAME network attribute value.

component-name-token-n: One of a series of 1 to 16 character tokens that uniquely identifies the object on which the action is to be performed. Characters A through Z and 0 through 9 can be used. Other special values (@, #, and \$) can be used for tokens that represent network IDs and system names.

NODL Specifies that the node list parameter is the object name that contains a list of systems that are the destinations for the activity. This parameter cannot be specified if the control point name (CPNAME) parameter is also specified.

***NONE:** The systems on which this activity is to be performed are not specified by a node list. Individual control point names must be specified.

The possible library values are:

***LIBL:** All of the libraries in the user and system portions of the job's library list are searched for the node list object.

***CURLIB:** The current library for the job is used to locate the node list object.

library-name: Specify the name of the library to be searched.

node-list-name: Specify the node list object name containing the list of systems on which the activity is to be performed.

CPNAME

Specifies the APPN control point names of the managed systems on which this activity is to be performed. Control point names cannot be specified if the node list (NODL) parameter is specified.

CPNAME is required unless NODL is specified.

***NONE:** The systems on which this activity is to be performed are not identified individually. A node list must be specified.

***NETATR:** The network ID of the local system is used. This is useful when the node being specified is in the same network as the local system.

network-identifier: Specify the APPN network identifier of the managed system on which the activity is to be performed. For NetView Distribution Management Agents, the network identifier is the change control server name.

control-point-name: Specify the APPN control point name of the managed system on which the activity is to be performed. For NetView Distribution Management Agents, the control point name is the change control client which supports numeric characters (0-9) in the first position of control point names that are valid in other platforms.

TEXT Specifies the activity description.

***GEN:** A text description should be generated for the activity being added. The text description should be generated based the verb specified on the ACTION parameter. The text descriptions generated are:

- Accept object
- Restart the system
- Delete object
- Run object
- Install object
- Remove object
- Retrieve object
- Send object
- Send and run object
- Send and install object
- Uninstall object

***BLANK:** No text is specified.

text-description: Specify a 50-character textual description of the activity.

ACTFRC

Specifies whether or not the managed system should proceed with the activation based on its quiesced state.

activation force: Specify the activation force value.

***NO:** The managed system will not proceed with the activation if the quiesce check shows that the managed system is still active.

delay units: Indicates the unit of time on which the delay period is specified.

***SECONDS:** The delay period will be specified in seconds.

***MINUTES:** The delay period will be specified in minutes.

***HOURS:** The delay period will be specified in hours.

delay period: Specifies the maximum amount of time that the managed system may wait to quiesce (if not already quiesced) before taking the action specified:

3600: The delay period default is 3600 seconds.

1-65535: Delay period range.

***YES**: The managed system will proceed with the activation even if the quiesce check shows that the managed system is still active. Delay units and delay period are ignored when activation force is ***YES**.

Note: ACTFRC is valid only when ACTION(*ACT) is specified.

ACTUSEACT

Activation use on activate. Specifies which components altered by changes will be used during the activation.

***NONE**: No activation use on activate is specified.

***BOTH**: Both trial and production version.

***PROD**: Production version only.

***LAST**: Last used; either both trial and production or production only.

Note: ACTUSEACT is valid only when ACTION(*ACT) is specified.

CPRTYPE

Specifies the compression algorithm and related information associated with the compression of a particular change object. When one or both of the compression transfer state or compression storage state parameter specify ***COMPRESS**, this parameter must be present and one (and only one) of the adaptive compression, SNA compression, or user compression algorithms may be requested. This support is dependent on the implementation on the managed system.

***NONE**: No compression type is specified.

***ADAPTIVE**: Specifies whether or not adaptive compression pertains to the requested object. The iSeries does not support ***ADAPTIVE**.

***SNA**: Specifies whether or not SNA compression pertains to the requested object.

***USER**: Specifies whether or not a named user compression pertains to the requested object.

Notes:

1. CPRTYPE is valid only when ACTION(*RTV), ACTION(*SND), ACTION(*SNDRUN), or ACTION(*SNDINS) is specified
2. CPRTYPE cannot be specified when CMPSTGSTT and CPRTFRSTT are not specified.
3. CPRTYPE cannot be specified when CMPSTGSTT is ***DECOMPRESS** and CPRTFRSTT is not specified.
4. CPRTYPE cannot be specified when CPRSTGSTT is not specified and CPRTFRSTT is ***DECOMPRESS**.
5. CPRTYPE cannot be specified when CPRSTGSTT and CPRTFRSTT are ***DECOMPRESS**.

SNACPRCHR

Specifies information about the SNA compression algorithm as it pertains to the requested object. If omitted, the implication is that SNA compression does not pertain to the requested object.

***BLANK:** When SNA compression is requested, it is optionally specified with the implied default being the (X'40') character. Otherwise, it is not specified.

SNA prime character: The prime compression character to be associated with the single control byte (SCB) used by the SNA compression algorithm. Valid values are '00'X - 'FF'X

Note: SNACPRCHR is valid only when ACTION(*RTV), ACTION(*SND), ACTION(*SNDRUN) or ACTION(*SNDINS) is specified.

USRCPRINF

Specifies information about a named user compression algorithm as it pertains to the requested object. If omitted, the implication is that user compression does not pertain to the requested object.

user-compression-name: The name of the user compression algorithm that pertains to the requested object. It should be specified when user compression is requested. Otherwise, it is not specified.

user-parameters: User parameters that apply to the user compression algorithm named in the user compression name. It is optionally specified when user compression is requested. Otherwise, it is not specified.

Note: USRCPRINF is valid only when ACTION(*RTV), ACTION(*SND), ACTION(*SNDRUN) or ACTION(*SNDINS) is specified.

CPRSTGSTT

Specifies whether or not the object should be stored in compressed format at the managed system once the object is sent.

***NONE:** No compression storage state is specified.

***DECOMPRESS:** Store the object in a decompressed format at the managed system. If the object is already decompressed when it arrives to the managed system, store it as received. Otherwise, decompress the object using the FS encoded algorithm before storing it.

***COMPRESS:** Store the object in a compressed format at the managed system. When this value is specified, the compression algorithm must also be specified. If the object is already compressed when it arrives at the managed system, store the object as received. Otherwise, compress the object using the compression algorithm before storing it.

Note: CPRSTGSTT is valid only when ACTION(*RTV), ACTION(*SND), ACTION(*SNDRUN) or ACTION(*SNDINS) is specified.

CPRTFRSTT

Specifies whether or not the object should be transferred to the managed system in compressed format.

***NONE:** No compression transfer state is specified.

***DECOMPRESS:** Transfer the object in decompressed format to the managed system. If already decompressed at the source, transfer the object as stored. Otherwise, decompress the object using the compression method used to catalog the object in the managed system before sending it.

***COMPRESS:** Compress the object using the compression algorithm and transfer the object in compressed format to the managed system. When this value is specified, the compression algorithm must also be specified. If the object is already compressed using a different algorithm at the source, decompress the object using the compression method used to catalog the object in the managed system before compressing and transferring it.

Note: CPRTFRSTT is valid only when ACTION(*RTV), ACTION(*SND), ACTION(*SNDRUN) or ACTION(*SNDINS) is specified.

ACTUSEINS

Specifies whether the component to be altered by the installation process will be trial version or production version.

If activation use on install is *TRIAL, it means the object should be installed in the trial area to be tested. Removability must then be *YES.

***PROD:** Production version only.

***TRIAL:** Trial version only.

Note: ACTUSEINS is valid only when ACTION(*INS) or ACTION(*SNDINS) is specified.

ALTACTIONCOMP

Specifies whether or not the managed system is allowed to apply the component alterations to the active system. If alterations cannot be applied, then such action is to be deferred until the next activation. This parameter is only allowed for *SNDINS action.

***NONE:** No alter active component is specified.

***ALLOWED:** Managed system is allowed to apply the component alterations to the active system.

***NOTALLOWED:** Managed system is not allowed to apply the component alterations to the active system.

Note: ALTACTIONCOMP is valid only when ACTION(*INS), ACTION(*SNDINS), ACTION(*RMV) or ACTION(*UNINS) is specified.

AUTOACP

Specifies whether the managed system accepts objects automatically if installation and any tests performed are successful, in order to release resources required to maintain removability as soon as possible. Like a separate accept request, the managed system deletes the objects after successful automatic acceptance.

***NONE:** No auto accept is specified.

***NO:** Do not perform automatic acceptance.

auto accept: Auto acceptance possible values:

***YES:** Perform automatic acceptance.

***DESIRED:** Perform automatic acceptance, if the specified managed system supports it.

delay: Specifies the number of days the entry point is expected to wait before accepting the object automatically.

delay-days: Valid range of days is 0-255.

Note: AUTOACP is valid only when ACTION(*INS) or ACTION(*SNDINS) is specified.

AUTORMV

Specifies whether or not the managed system removes the object automatically if failure by either installation or a test.

***NONE:** No auto remove is specified.

***DESIRED:** Perform automatic removal if the specified managed system supports it.

***YES:** Perform automatic removal.

***NO:** Do not perform automatic removal.

Note: AUTORMV is valid only when ACTION(*INS) or ACTION(*SNDINS) is specified.

PRETEST

Specifies whether or not the entry point is to perform a test on the objects prior to installing them.

***DESIRED:** Perform a pretest if the specified managed system supports it.

***YES:** Perform a pretest.

***NO:** Do not perform a pretest.

Note: PRETEST is valid only when ACTION(*INS) or ACTION(*SNDINS) is specified.

POSTTEST

Specifies whether or not the entry point is to perform a test on the objects after installing or removing them.

***DESIRED:** Perform a posttest if the specified managed system supports it.

***YES:** Perform a posttest.

***NO:** Do not perform a posttest.

Note: POSTTEST is valid only when ACTION(*INS), ACTION(*SNDINS) or ACTION(*RMV) is specified.

ALWRMV

Specifies whether or not objects are to be installed in a removable manner (so that a subsequent remove action can be issued against the objects).

***YES:** Install the object in a removable manner.

***DESIRED:** Install the object in a removable manner if the specified managed system supports it.

***NO:** Do not install the object in a removable manner.

Note: ALWRMV is valid only when ACTION(*INS) or ACTION(*SNDINS) is specified.

PARM Specifies the parameters to be passed to the program to be run.

parameter data: Specifies the parameters to be passed when starting the program. Each parameter is a 1 to 253 character. The initial display allows 25 characters to be typed. By typing an & in position 1, the field expands to accommodate longer parameters. If the parameters include blanks or special characters, enclose them in apostrophes.

Note: PARM is valid only when ACTION(*RUN) or ACTION(*SNDRUN) is specified.

KEEPOBJ

Specifies whether or not the object should be kept or deleted after the function has been successfully performed.

***NONE:** No keep object is specified.

***YES:** The object should be kept after performing the function.

***NO:** The object should be deleted after performing the function.

Note: KEEPOBJ is valid only when ACTION(*RUN), ACTION(*INS), ACTION(*SNDRUN) or ACTION(*SNDINS) is specified.

COREQCHGNL

Specifies a list of SNA/FS global names identifying the names of objects that are to be installed by the entry point as a part of the installation of the object to be retrieved. A maximum of six corequisite change names are allowed.

The global name is a unique name you assign to the object so that it is not confused with any other object in a network. The global name represents the name that will be used to locate the appropriate catalog entry on both the central site and managed systems.

***NONE:** No corequisite change name list is specified.

global-name-token: One of a series of 1-16 character tokens that uniquely identify the object on which the action is to be performed. Characters A through Z and 0 through 9 may be used. Other special values (@, #, and \$) may be used for tokens that represent network IDs and system names.

Note: COREQCHGNL is valid only when ACTION(*INS) or ACTION(*SNDINS) is specified.

REPLACE

Specifies whether the object should be replaced if it already exists.

***NO:** The object must be added.

***YES:** The object must be replaced.

***ALLOWED:** The object should be replaced or added.

Note: REPLACE is valid only when ACTION(*RTV), ACTION(*SND), ACTION(*SNDRUN) or ACTION(*SNDINS) is specified.

TODLTNAME

Specifies the name of the object, at the managed system location, that is to be deleted.

Special values in a token position indicate how to search for the object. By specifying *ANY in a token position, the token is ignored when searching for the correct object. If multiple objects are found matching the tokens specified, an error is returned.

Special values in a token position are allowed only for actions *DLT, *RTV, *SND, *SNDRUN, or *SNDINS.

***NONE:** No to be deleted name is specified.

Element 1: Token 1

***NETID:** The first global name token value is a network ID generated by the command from the network attributes. The network ID is determined by the current value of the LCLNETID network attribute value.

***SERVER:** This token is stored within the change request activity with the value &SERVER, and is replaced by the short name of the change control server when the object is distributed.

***TARGET:** This token is stored within the change request activity with the value &TARGET, and is replaced by the short name of the target when the object is distributed.

***MDDATE:** This token is stored within the change request activity with the value &DATE, and is replaced when distributed by the date that the object was last changed.

***MDTIME:** This token is stored within the change request activity with the value &TIME, and is replaced when distributed by the time that the object was last changed.

global-name-token-1: Specify the first token of the global name. The first token is recommended to be the registered enterprise ID or network ID.

Element 2-10: Token 2-10

***ANY:** Any token value matches when searching for the object where the action is to be performed.

***HIGHEST:** The object with the highest token value has the action performed on it. The token must be ordered. This is useful when a token in a global name is used to indicate a different version of the object and you need to manipulate the object with the highest version level.

***LOWEST:** The object with the lowest token value has the action performed on it. The token must be ordered. This is useful when a token in a global name is used to indicate a different version of the object and you need to manipulate the object with the lowest version level.

***NETID:** The network ID of this system is used. The network ID is determined by the current value of the LCLNETID network attribute value.

***CPNAME:** The control point name of this system is used. The control point name is determined by the current value of the LCLCPNAME network attribute value.

***SERVER:** This token is stored within the change request activity with the value &SERVER, and is replaced by the short name of the change control server when the object is distributed.

***TARGET:** This token is stored within the change request activity with the value &TARGET, and is replaced by the short name of the target when the object is distributed.

***MDDATE:** This token is stored within the change request activity with the value &DATE, and is replaced when distributed by the date that the object was last changed.

***MDTIME:** This token is stored within the change request activity with the value &TIME, and is replaced when distributed by the time that the object was last changed.

global-name-token-n: Specify one of a series of 1 to 16 character tokens that uniquely identify the object on which the action is to be performed. Characters A through Z and 0 through 9 can be used. Other special values (@, #, and \$) can be used for tokens that represent network IDs and system names.

Notes:

1. TODLTNAME is valid only when ACTION(*SND), ACTION(*SNDRUN) or ACTION(*SNDINS) is specified.
2. The number of tokens specified in the TODLTNAME parameter must match the number of tokens specified in the GLBNAME parameter.
3. TODLTNAME cannot be specified if the REPLACE value is *NO.

REFLVL

Specifies the level of the software component to be uninstalled.

***NONE:** No refresh level is specified.

refresh-level: Specify a numeric character string up to 16 digits.

Note:

REFLVL is valid only when ACTION(*UNINS) is specified.

FRCUNINS

Specifies whether to perform the uninstallation even if the software component includes an object that is waiting to be installed or an object that is currently being installed..

***NO:** Do not allow uninstallation if there are objects, for the software component, waiting to be installed.

***YES:** Allow uninstallation even if there are objects, for the software component, waiting to be installed.

Note:

FRCUNINS is valid only when ACTION(*UNINS) is specified.

RMTSTRTIME

Specifies the date and time when the activity can begin running on the managed system. The current date and time values and next date values are determined when the activity begins running at the central site system based on the central site date and time.

Single value:

***NONE:** No start time on the managed system is specified.

Element 1: Time Zone

The time zone of the remote start time.

***LCLSYS:** The remote start time is specified in the time zone of the central site system. The start time is converted to Greenwich Mean Time.

***MGDSYS:** The remote start time is specified in the time zone of the managed system.

Element 2: Start After Time

This is the definition of the time after which the activity is to start.

***CURRENT:** This function can start on the managed system at any time on or after the time this activity is started on the central site system on the date specified in element 3.

start-after-time: Specify the time when this function can start on the managed system. The time can be entered as 4 or 6 digits (hhmm or hhmmss) where hh = hours, mm = minutes, and ss = seconds. Seconds are optional. The time can be specified with or without a time separator. With a time separator, specify a string of 5 or 8 digits (hh:mm or hh:mm:ss).

Element 3: Start After Date

***CURRENT:** This function starts on the managed system on any date on or after the activity starts on the central site system.

***NEXT:** This function starts on the managed system on any date after the activity starts on the central site system.

start-after-date: Specify the date after the functions start on the managed system. The date must be specified in the job date format.

Notes:

1. The special values *CURRENT and *NEXT cannot be specified for the date and the time when the time zone value *MGDSYS is specified.
2. This parameter is valid only when these actions are specified: *ACP, *ACT, *RUN, *INS, *RMV, *SNDRUN, *SNDINS, or *UNINS.

COND Specifies which conditions must be met before this activity can be performed. Each condition identifies an activity that must run before this activity and the value the end code from that activity must have to allow this activity to run. The default condition is that the previous activity (in alphabetical order) must complete successfully before this activity can be run.

Element 1: Conditioning Activity

The activity that must be run before this activity.

***PRV:** This activity is conditioned on the previous activity. Activities are ordered alphabetically by activity name. If the activity being added is the first activity, a previous activity does not exist and any condition with *PRV is marked as having been met.

conditioning-activity-name: Specify the name of the activity that must be run before this activity. The activity name specified in the activity (ACTIVITY) parameter cannot be specified in the conditioning activity name. An activity cannot be conditioned on itself.

generic-conditioning-activity-name:* Specify the generic name of the activities that must be run before this activity.

Element 2: Relational Operator

This element is the relational operator to use when comparing the end code from the conditioning activity.

***EQ:** Equal

***GT:** Greater than

***LT:** Less than

***NE:** Not equal

***GE:** Greater than or equal

***LE:** Less than or equal

Element 3: Condition Code

This element is the value compared to the actual end code of the conditioning activity.

***SUCCESS:** The activity ended successfully (0 <= end code <= 9). This end code can only be specified with relational operator *EQ or *NE.

***FAIL:** The activity failed (10 <= end code <= 89). This end code can only be specified with relational operator *EQ or *NE.

***NOTRUN:** The activity never started (90 <= end code <= 99). This end code is only specified with relational operator *EQ or *NE.

***ANY:** The activity ended with any end code. This end code is only specified with relational operator *EQ.

end-code: Specify an integer value (0-99) that indicates the result of an activity (success or failure). The end code ranges and descriptions are:

00 Activity completed successfully.

01-09 Activity completed with warning messages.

10-29 Activity did not complete successfully.

30-39 Activity was canceled by a user before it completed.

- 30 = Activity ended with *CNTRLD option
- 35 = Activity ended with *IMMED option
- 39 = Activity ended with *FRCFAIL option

40-49 Activity was not run due errors detected by the application.

- 40 = Activity not run for security reasons

90-99 Activity was not run because conditions or schedules were not met.

- 95 = Scheduled start time expired

- 99 = Conditions cannot be met

Element 4: Condition Mode

This element indicates which systems the conditioning activity must have completed on before this activity can be performed.

***ALLNODES:** The conditioning activity specified must complete on all nodes before this activity runs.

***SAMENODE:** When the conditioning activity specified completes for a given node, the activity specified on the ACTIVITY parameter may run for that same node even though the conditioning activity specified cannot have completed for all other nodes. In the case where this activity lists a node not in the conditioning activity, this activity may run for that node; the condition is ignored.

***NONE:** There are no conditions for this activity.

STRTIME

Specifies the date and time when this activity can be started on the central site system. The current date and time values, and next date values are determined when the change request is submitted.

Element 1: Start After Time

***CURRENT:** This activity can start any time on or after the time when the change request is submitted.

start-after-time: Specify the time when this activity can start. The time can be entered as 4 or 6 digits (hhmm or hhmmss) where hh = hours, mm = minutes, and ss = seconds. Seconds are optional. The time can be specified with or without a time separator such as a colon (:). With a time separator, specify a string of 5 or 8 digits (hh:mm or hh:mm:ss).

Element 2: Start After Date

***CURRENT:** This activity can start on or after the date on which the change request is submitted.

***NEXT:** The activity can start on any date after the date when the change request is submitted.

start-after-date: Specify the date after which this activity can start. The date must be specified in the job date format.

Element 3: Start Before Time

This element is ignored if the start before date is *ANY.

***ANY:** The activity can start at any time on or before the start before date.

***CURRENT:** The activity must start before the time when the change request was submitted on the date specified on the start before data element.

start-before-time: Specify the time before the activity must be started. If the activity cannot be started before this time, it never starts. The time can be entered as 4 or 6 digits (hhmm or hhmmss) where hh = hours, mm = minutes, and ss = seconds. Seconds are optional. The time can be specified with or without a time separator such as a colon (:). With a time separator, specify a string of 5 or 8 digits (hh:mm or hh:mm:ss).

Element 4: Start Before Date

***ANY:** The activity can start at any time after the start after time and the start after date.

***CURRENT:** The activity must start on the date the change request is submitted.

***NEXT:** The activity must start by the day after the date the change request is submitted.

start-before-date: Specify the date before the activity must start. If the activity cannot be started by this date, it never starts. The date must be specified in the job date format.

HOLD Hold the activity when the change request is submitted.

***NO:** The activity is not held and will run when all conditions are met.

***YES:** The activity is held for all nodes when the change request is submitted. It must be released by you before it runs.

Examples for ADDCRQA

The following examples illustrate how to use the ADDCRQA command to schedule activities to be performed on managed systems controlled by a NetView Distribution Manager/6000 change control server. The examples shown here are grouped according to their activity action:

- Accept
- Activate
- Delete
- Run
- Install
- Remove
- Retrieve
- Send, send and run, send and install
- Uninstall

Example 1: Accept Actions

The examples shown here match those shown for the install examples and demonstrate how some of the installed objects can be accepted.

- The install activity added to the change request description is:

```
ADDCRQA CRQD(CCLIB/CRQACPRMV)
  ACTIVITY(INSACT01)
  ACTION(*INS)
  GLBNAME(IBM 1234567 PMGRAB UPD 1 2)
  CPNAME((ROMSERV1 MARYPWS1))
  ACTUSEINS(*TRIAL)
  ALWRMV(*YES)
```

When the change request is submitted and if the activity is run successfully, object IBM 1234567 PMGRAB UPD 1 2 is installed (to be tested) in the trial area of managed system MARYPWS1. MARYPWS1 is controlled by the change control server ROMSERV1. Because the object is installed in the trial area, the object cannot be accepted.

- The install activity added to the change request description is:

```
ADDCRQA CRQD(CCLIB/CRQACPRMV)
  ACTIVITY(INSACT02)
  ACTION(*INS)
  GLBNAME(IBM 1234567 PMGRAB UPD 1 2)
  CPNAME((ROMSERV1 MARYPWS1))
  ACTUSEINS(*PROD)
  ALTACTCOMP(*ALLOWED)
  ALWRMV(*NO)
  RMTSTRTIME((*MGDSYS) (8:30:00 12/25/02))
```

When the change request is submitted and if the activity is successfully run, object IBM 1234567 PMGRAB UPD 1 2 is installed in the active area of the system MARYPWS1. MARYPWS1 is managed by the change control server ROMSERV1. The object is installed in a nonremovable manner, and thus, the object does not need to be accepted.

- The install activity added to the change request description is:

```
ADDCRQA CRQD(CCLIB/CRQACPRMV)
  ACTIVITY(INSACT03)
  ACTION(*INS)
  GLBNAME(IBM 1234567 PMGRAB UPD 1 2)
  CPNAME((ROMSERV1 MARYPWS1))
  COREQCHGNL((REXX PROC UPDATE CONFIG))
  ACTUSEINS(*PROD)
  ALTACTCOMP(*NOTALLOWED)
  AUTOACP((*YES))
  ALWRMV(*YES)
```

The objects are automatically accepted with a required activation if the following conditions are met:

- The change request is submitted
- The activity runs successfully
- The installation completes successfully (the object IBM 1234567 PMGRAB UPD 1 2 and its corequisite REXX PROC UPDATE CONFIG were installed successfully)

If the installation did not complete successfully, the objects IBM 1234567 PMGRAB UPD 1 2 and REXX PROC UPDATE CONFIG are not automatically accepted. You can accept the objects running the following activities:

```
ADDCRQA CRQD(CCLIB/CRQACPRMV)
  ACTIVITY(ACCACT01)
  ACTION(*ACP)
  GLBNAME(IBM 1234567 PMGRAB UPD 1 2)
  CPNAME((ROMSERV1 MARYPWS1))
  COND((INSACT03 *EQ 20 *SAMENODE))
```

```
ADDCRQA CRQD(CCLIB/CRQACPRMV)
  ACTIVITY(ACCACT02)
  ACTION(*ACP)
  GLBNAME(REXX PROC UPDATE CONFIG)
  CPNAME((ROMSERV1 MARYPWS1))
  COND((INSACT03 *EQ 20 *SAMENODE))
```

- The install activity added to the change request description is:

```
ADDCRQA CRQD(CCLIB/CRQACPRMV)
  ACTIVITY(INSACT04)
  ACTION(*INS)
  GLBNAME(EURO WORDPROD UPD 2 3 US)
  CPNAME((EUROITAL FREDSW))
  ALWRMV(*YES)
```

When the change request is submitted and if the activity is run successfully, an object is installed in a removable manner. The object can be accepted adding an activity to accept the previously installed object EURO WORDPROD UPD 2 3 US on managed system FREDSW. Managed system FREDSW is controlled by the change control server EUROITAL.

```
ADDCRQA CRQD(CCLIB/CRQACPRMV)
  ACTIVITY(ACCACT03)
  ACTION(*ACP)
  GLBNAME(EURO WORDPROD UPD 2 3 US)
  CPNAME((EUROITAL FREDSW))
  COND((INSACT04 *EQ *SUCCESS *SAMENODE))
```

Example 2: Activate Actions

- It is assumed that an install activity was issued as follows:


```

ADDCRQA CRQD(CCLIB/CRQINSACT)
  ACTIVITY(INSACT01)
  ACTION(*INS)
  GLBNAME(IBM 1234567 PMGRAB UPD 1 2)
  CPNAME((ROMSERV1 MARYPWS1))
  ACTUSEINS(*PROD)
  ALTACTIONCOMP(*NOTALLOWED)
  AUTOACP(*YES)
  COREQCHGNL((REXX PROC UPDATE CONFIG))

```

- When the change request is submitted and if the activity is run successfully, objects IBM 1234567 PMGRAB UPD 1 2 and REXX PROC UPDATE CONFIG are automatically accepted. To apply the component alterations to the active system, an activation is required. To activate previously installed changes, run an activity like the following:

```

ADDCRQA CRQD(CCLIB/CRQINSACT)
  ACTIVITY(ACTACT01)
  ACTION(*ACT)
  CPNAME((ROMSERV1 MARYPWS1))
  ACTFRC(*NO *HOURS 6)
  ACTUSEACT(*PROD)
  STRTIME((2:00:00 *NEXT)(8:00:00 *NEXT))

```

The activation will occur after 2 a.m. but before 8 a.m. the next morning the request is submitted. The MARYPWS1 system waits up to six hours before proceeding with the activation if the managed system is still active.

- Add an activity to schedule the activation of all previously installed objects on managed system FREDSWWS. FREDSWWS is controlled by the change control server EUROITAL. The activation occurs as soon as possible after 11 p.m. on 15 April 2002 in the time zone where FREDSWWS is located. The activation occurs even if the managed system FREDSWWS is still active.

```

ADDCRQA CRQD(CCLIB/CRQINSACT)
  ACTIVITY(ACTACT02)
  ACTION(*ACT)
  CPNAME((EUROITAL FREDSWWS))
  ACTFRC(*YES)
  RMTSTRTIME(*MGDSYS (23:00:00 04/15/02))

```

Example 3: Delete Actions

- Add an activity to delete the object EURO WORDPROD UPD 2 3 US on managed system FREDSWWS. FREDSWWS is controlled by the change control server EUROITAL.

```

ADDCRQA CRQD(CCLIB/CRQDLT)
  ACTIVITY(DLTACT01)
  ACTION(*DLT)
  GLBNAME(EURO WORDPROD UPD 2 3 US)
  CPNAME((EUROITAL FREDSWWS))

```

- Add an activity to delete all test files from systems MARYPWS1, MARYPWS2, MARYPWS3, and MARYPWS4. These systems are controlled by the change control server ROMSERV1. The test files on these systems are cataloged as EURO SPELLCHECK TEST file-name.

```

ADDCRQA CRQD(CCLIB/CRQSNDLT)
  ACTIVITY(DLTACT01)
  ACTION(*DLT)
  GLBNAME(EURO SPELLCHECK TEST *ANY)
  CPNAME((ROMSERV1 MARYPWS1)
         (ROMSERV1 MARYPWS2)
         (ROMSERV1 MARYPWS3)
         (ROMSERV1 MARYPWS4))
  COND((SNDACT01 *EQ *SUCCESS *ALLNODES))

```

The delete activity is performed only if the object EURO SPELLCHECK EXE 1 US is successfully sent to those systems:

```

ADDCRQA CRQD(CCLIB/CRQSNDLT)
ACTIVITY(SNDACT01)
ACTION(*SND)
GLBNAME(EURO SPELLCHECK EXE 1 US)
CPNAME((ROMSERV1 MARYPWS1)
        (ROMSERV1 MARYPWS2)
        (ROMSERV1 MARYPWS3)
        (ROMSERV1 MARYPWS4))
REPLACE(*ALLOWED)

```

Example 4: Run Actions

- Add an activity to run the program or script known by the global name EURO VIRUSCHK EXE 1 US on managed system FREDWS. Pass the parameter /usr/bin to the program or script. The activity is run as soon as possible.

```

ADDCRQA CRQD(CCLIB/CRQRUN)
ACTIVITY(RUNACT01)
ACTION(*RUN)
GLBNAME(EURO VIRUSCHK EXE 1 US)
CPNAME((EUROITAL FREDWS))
PARM(("/usr/bin"))

```

- Add an activity to run the program known by the global name EURO WORDPROC EXE 2 US on managed system JOHNSWS. This activity should be run immediately by the PS/2. The object should be kept in the NetView/DM2 catalog. The change control server is in the same network as the local system.

```

ADDCRQA CRQD(CCLIB/CRQRUN)
ACTIVITY(RUNACT02)
ACTION(*RUN)
GLBNAME(EURO WORDPROC EXE 2 US)
CPNAME((*NETATR JOHNSWS))
KEEPOBJ(*YES)

```

Example 5: Install Actions

The first four examples shown here match those shown for the accept and remove examples.

- Add an activity to install the object identified by the global name IBM 1234567 PMGRAB UPD 1 2 on the change control client machine (managed system) called MARYPWS1. MARYPWS1 is controlled by the change control server ROMSERV1. The object is to be installed in the trial area.

```

ADDCRQA CRQD(CCLIB/CRQACPRMV)
ACTIVITY(INSACT01)
ACTION(*INS)
GLBNAME(IBM 1234567 PMGRAB UPD 1 2)
CPNAME((ROMSERV1 MARYPWS1))
ACTUSEINS(*TRIAL)
ALWRMV(*YES)

```

- Add an activity to install the object identified by the global name IBM 1234567 PMGRAB UPD 1 2 on the change control client machine (managed system) called MARYPWS1. MARYPWS1 is controlled by the change control server ROMSERV1. The object is to be installed permanently in the active area. The activity is to be processed at 8:30 a.m. on 25 December 2002.

```

ADDCRQA CRQD(CCLIB/CRQACPRMV)
ACTIVITY(INSACT02)
ACTION(*INS)
GLBNAME(IBM 1234567 PMGRAB UPD 1 2)
CPNAME((ROMSERV1 MARYPWS1))
ACTUSEINS(*PROD)
ALACTCOMP(*ALLOWED)
ALWRMV(*NO)
RMTSTRTIME((*MGDSYS) (8:30:00 12/25/02))

```

- Add an activity to install the objects identified by the global names IBM 1234567 PMGRAB UPD 1 2 and its corequisite REXX PROC UPDATE CONFIG on the change control client (managed system)

MARYPWS1. MARYPWS1 is controlled by the change control server ROMSERV1. The objects are to be installed in a nonremovable manner and are to be automatically accepted. An activation is required to apply the component alterations to the active system. The activity is to be processed immediately. If the installation of one of the objects fails, the other installation fails also.

```
ADDCRQA CRQD(CCLIB/CRQACPRMV)
  ACTIVITY(INSACT03)
  ACTION(*INS)
  GLBNAME(IBM 1234567 PMGRAB UPD 1 2)
  CPNAME((ROMSERV1 MARYPWS1))
  ACTUSEINS(*PROD)
  ALTACTIONCOMP(*NOTALLOWED)
  AUTOACP((*YES))
  ALWRMV(*YES)
  COREQCHGNL((REXX PROC UPDATE CONFIG))
```

- Add an activity to install the object identified by the global name EURO WORDPROD UPD 2 3 US on the change control client machine (managed system) called FREDSW. FREDSW is controlled by the change control server EUROITAL. The object is to be installed in a removable manner.

```
ADDCRQA CRQD(CCLIB/CRQACPRMV)
  ACTIVITY(INSACT04)
  ACTION(*INS)
  GLBNAME(EURO WORDPROD UPD 2 3 US)
  CPNAME((EUROITAL FREDSW))
  ALWRMV(*YES)
```

- Add an activity to schedule the install of the objects EURO.WORDPROC.REF.2.US and EURO.WORDPROC.UPD.2.3.US on managed system FREDSW. FREDSW is controlled by the change control server EUROITAL. The object is installed in the active area in a removable manner. The installation is not automatically accepted. The installation is scheduled for 3 p.m. on 1 January 2003.

```
ADDCRQA CRQD(CCLIB/CRQRTVINS)
  ACTIVITY(INSACT05)
  ACTION(*INS)
  GLBNAME(EURO WORDPROC REF 2 US)
  CPNAME((EUROITAL FREDSW))
  ALTACTIONCOMP(*ALLOWED)
  COREQCHGNL(EURO WORDPROC UPD 2 3 US)
  RMTSTRTIME(*MGDSYS (15:00:00 1/01/03))
  COND((RTVACT01 *EQ *SUCCESS *ALLNODES))
```

The activity can be performed only if the object EURO WORDPROC UPD 2 3 US is successfully retrieved from the change control client BRIGSW.

```
ADDCRQA CRQD(CCLIB/CR7)
  ACTIVITY(RTVACT01)
  ACTION(*RTV)
  GLBNAME(EURO WORDPROC UPD 2 3 US)
  CPNAME((EUROITAL BRIGSW))
  STRTIME((22:00:00 12/31/02) (06:00:00 1/01/03))
```

- Add an activity to install the object identified by the global name IBM 1234567 WINDMB UPD 1 2 on the production area of managed system MARYPWS1. If the installation fails, the object is removed automatically.

```
ADDCRQA CRQD(CCLIB/CRQINS)
  ACTIVITY(INSACT06)
  ACTION(*INS)
  GLBNAME(IBM 1234567 WINDMB UPD 1 2)
  CPNAME((ROMSERV1 MARYPWS1))
  ACTUSEINS(*PROD)
  AUTORMV(*YES)
  ALWRMV(*YES)
```

Example 6: Remove Actions

The examples shown here match those shown for the install examples and demonstrate how the installed objects can be removed.

- The install activity added to the change request description is:

```
ADDCRQA CRQD(CCLIB/CRQACPRMV)
  ACTIVITY(INSACT01)
  ACTION(*INS)
  GLBNAME(IBM 1234567 PMGRAB UPD 1 2)
  CPNAME((ROMSERV1 MARYPWS1))
  ACTUSEINS(*TRIAL)
  ALWRMV(*YES)
```

When the change request is submitted and if the activity is run successfully, object IBM 1234567 PMGRAB UPD 1 2 is installed (to be tested) in the trial area of managed system MARYPWS1. MARYPWS1 is controlled by the change control server ROMSERV1. The object IBM 1234567 PMGRAB UPD 1 2 can be removed when you run the following activity:

```
ADDCRQA CRQD(CCLIB/CRQACPRMV)
  ACTIVITY(RMVACT01)
  ACTION(*RMV)
  GLBNAME(IBM 1234567 PMGRAB UPD 1 2)
  CPNAME((ROMSERV1 MARYPWS1))
  POSTST(*NO)
```

A posttest is not required.

- The install activity added to the change request description is:

```
ADDCRQA CRQD(CCLIB/CRQACPRMV)
  ACTIVITY(INSACT02)
  ACTION(*INS)
  GLBNAME(IBM 1234567 PMGRAB UPD 1 2)
  CPNAME((ROMSERV1 MARYPWS1))
  ACTUSEINS(*PROD)
  ALTACTIONCOMP(*ALLOWED)
  ALWRMV(*NO)
  RMTSTRTIME((*MGDSYS) (8:30:00 12/25/02))
```

When the change request is submitted and if the activity is run successfully, object IBM 1234567 PMGRAB UPD 1 2 is installed in the active area of the system MARYPWS1. MARYPWS1 is managed by the change control server ROMSERV1. The object is installed in a nonremovable manner.

- The install activity added to the change request description is:

```
ADDCRQA CRQD(CCLIB/CRQACPRMV)
  ACTIVITY(INSACT03)
  ACTION(*INS)
  GLBNAME(IBM 1234567 PMGRAB UPD 1 2)
  CPNAME((ROMSERV1 MARYPWS1))
  ACTUSEINS(*PROD)
  ALTACTIONCOMP(*NOTALLOWED)
  AUTOACP((*YES))
  ALWRMV(*YES)
  COREQCHGNL((REXX PROC UPDATE CONFIG))
```

- The objects are automatically accepted if the following conditions are met:
 - The change request is submitted
 - The activity is run
 - The installation completes successfully (the object IBM 1234567 PMGRAB UPD 1 2 and its corequisite REXX PROC UPDATE CONFIG were installed successfully)

All objects are installed in a removable manner, but an activation is required. The objects can be removed by adding and running activities like the following:

```

ADDCRQA CRQD(CCLIB/CRQACPRMV)
ACTIVITY(RMVA02)
ACTION(*RMV)
GLBNAME(IBM 1234567 PMGRAB UPD 1 2)
CPNAME((ROMSERV1 MARYPWS1))
COND((INSACT03 *EQ *SUCCESS *SAMENODE))

```

```

ADDCRQA CRQD(CCLIB/CRQACPRMV)
ACTIVITY(RMVA03)
ACTION(*RMV)
GLBNAME(REXX PROC UPDATE CONFIG)
CPNAME((ROMSERV1 MARYPWS1))
COND((INSACT03 *EQ *SUCCESS *SAMENODE))

```

- The install activity added to the change request description is:

```

ADDCRQA CRQD(CCLIB/CRQACPRMV)
ACTIVITY(INSACT04)
ACTION(*INS)
GLBNAME(EURO WORDPROD UPD 2 3 US)
CPNAME((EUROITAL FREDSW))
ALWRMV(*YES)

```

When the change request is submitted and if the activity is run successfully, the object is installed in a removable manner. The object can be removed when you add an activity to remove the previously installed object EURO WORDPROD UPD 2 3 US on managed system FREDSW. FREDSW is controlled by the change control server EUROITAL.

```

ADDCRQA CRQD(CCLIB/CRQINSRMV)
ACTIVITY(RMVA04)
ACTION(*RMV)
GLBNAME(EURO WORDPROD UPD 2 3 US)
CPNAME((EUROITAL FREDSW))
COND((INSACT04 *EQ *SUCCESS *SAMENODE))

```

Example 7: Retrieve Actions

- Add an activity that causes the file EURO WORDPROC UPD 2 3 US to be retrieved from the managed system JOHNSWS. The file is transferred in a compressed format and then stored in a decompressed format. The SNA compression type is used.

```

ADDCRQA CRQD(CCLIB/CRQRTV)
ACTIVITY(RTVA01)
ACTION(*RTV)
GLBNAME(EURO WORDPROC UPD 2 3 US)
CPNAME((EUROITAL JOHNSWS))
CPRTYPE(*SNA)
CPRSTGTT(*DECOMPRESS)
CPRTFRSTT(*COMPRESS)
REPLACE(*ALLOWED)

```

- Add an activity that causes the object EURO PCSOFT UPD 2 3 US to be retrieved from the managed system JOHNSWS. The file is transferred and stored in a compressed format. The user compression name NVDMLZW is used. This user compression type is supported if the object is not decompressed at the central site system.

```

ADDCRQA CRQD(CCLIB/CRQRTV)
ACTIVITY(RTVA02)
ACTION(*RTV)
GLBNAME(EURO PCSOFT UPD 2 3 US)
CPNAME((EUROITAL JOHNSWS))
CPRTYPE(*USER)
CPRSTGTT(*COMPRESS)
CPRTFRSTT(*COMPRESS)
USRCPRINF(NVDMLZW 37 '/D')
REPLACE(*ALLOWED)

```

- Add an activity to retrieve the most recent sales file from each system. The files are cataloged as EURO SALES system-name date-created. The file is sent and stored in a compressed format. The SNA

compression algorithm is used. The file is retrieved after 10 p.m. on the day the request is submitted but before 6 a.m. the next morning when the stores open. All files retrieved are added to the distribution repository when they are retrieved.

```
ADDCRQA CRQD(CCLIB/CRQRTV)
ACTIVITY(RTVACT03)
GLBNAME(EURO SALES *ANY *HIGHEST)
NODL(STORES)
CPRTYPE(*SNA)
CPRSTGSTT(*COMPRESS)
CPRTFRSTT(*COMPRESS)
REPLACE(*NO)
STRTIME((22:00:00 *CURRENT) (06:00:00 *NEXT))
```

Example 8: Send, Send and Run, and Send and Install Actions

- Add an activity to send the file EURO SPELLCHECK EXE 1 US to the managed system FREDSW. FREDSW is controlled by the change control server EUROITAL. The file is not compressed. The file is replaced or added at the managed system.

```
ADDCRQA CRQD(CCLIB/CRQSND)
ACTIVITY(SNDCT01)
ACTION(*SND)
GLBNAME(EURO SPELLCHECK EXE 1 US)
CPNAME((EUROITAL FREDSW))
REPLACE(*ALLOWED)
```

- Add an activity to send and install the two objects EURO WORDPROC REF 2 US and EURO WORDPROC UPD 2 3 US on managed system FREDSW with these specified attributes. The installation is done to the active area in a removable manner. The installation is not automatically accepted. It is scheduled for 3 a.m. on 13 April 2002 in the time zone where the managed system is located.

```
ADDCRQA CRQD(CCLIB/CRQSND)
ACTIVITY(SNDCT02)
ACTION(*SNDINS)
GLBNAME(EURO WORDPROC REF 2 US)
CPNAME((EUROITAL FREDSW))
ALTACTCOMP(*ALLOWED)
COREQCHGNL((EURO WORDPROC UPD 2 3 US))
RMTSTRTIME(*MGDSYS (15:00:00 04/13/02))
```

- Add an activity to send and run the program known by the global name EURO.VIRUSCHK.EXE.1.US on managed system FREDSW. Pass the parameter /usr/bin to the program. The program is run as soon as possible.

```
ADDCRQA CRQD(CCLIB/CRQSND)
ACTIVITY(SNDCT03)
ACTION(*SNDRUN)
GLBNAME(EURO VIRUSCHK EXE 1 US)
CPNAME((EUROITAL FREDSW))
PARM(("/usr/bin"))
```

- Add activities to retrieve the program identified by the global name CUSTNET PCSOFT WDWAPP VER3 941128 from the PS/2 DEVPS2, which is controlled by the central site CUSTNET. Send and run on all of the PS/2s in the southeast area. The central site system runs the program at 11 p.m. in the time zone where the PS/2 is located only if the retrieve from the PS/2 DEVPS2 was successful. The activity names are generated.

```
ADDCRQA CRQD(CCLIB/CRQRTVSND)
ACTIVITY(*GEN)
ACTION(*RTV)
GLBNAME(CUSTNET PCSOFT WDWAPP VER5 021230)
CPNAME((CUSTNET DEVPS2))
```

```
ADDCRQA CRQD(CCLIB/CRQRTVSND)
ACTIVITY(*GEN)
ACTION(*SNDRUN)
```

```
GLBNAME(CUSTNET PCSOFT WDWAPP VER5 021230)
NODL(PS2SE)
RMTSTRTIME(*MGDSYS (23:00:00 12/30/02))
COND((*PRV *EQ *SUCCES *SAMENODE))
```

Example 9: Uninstall Actions

- Add an activity to uninstall the component EURO WORDPROC from managed system FREDWS at the time and date specified. The system must be in a state of inactivity at that time. Do not perform the uninstall action if there are objects waiting to be installed.

```
ADDCRQA CRQD(CCLIB/CRQUNINS)
ACTIVITY(UNACT01)
ACTION(*UNINS)
COMPNAME(EURO WORDPROC)
CPNAME((EUROITAL FREDWS))
ALTACTCOMP(*NOTALLOWED)
FRCUNINS(*NO)
REFLVL("2")
RMTSTRTIME(*MGDSYS (14:00:00 05/20/02))
```

Error messages for ADDCRQA

*ESCAPE Messages

None <>>

ADDCLUNODE (Add Cluster Node Entry) Command Description

ADDCLUNODE Command syntax diagram

Purpose

The Add Cluster Node Entry (ADDCLUNODE) command is used to add a node to the membership list of an existing cluster.

If the START parameter is set to *NO, the node that is being added will have a status of New and Cluster Resource Services will not be started on that node. The Start Cluster Node (STRCLUNOD) command can be called from a program running on one of the active nodes in the cluster to start Cluster Resource Services on a node that does not have a status of Active.

If the START parameter on this command is set to *YES, Cluster Resource Services will be started on the node that is being added. If Cluster Resource Services is successfully started, the status for the added node will be set to Active. Successful completion of this command results in Cluster Resource Service jobs (QCSTCTL, QCSTCRGM, and a job for each cluster resource group object in the cluster) started in the QSYSWRK subsystem. If the Cluster Resource Services cannot be started, the status of the added node will be set to New.

During the activation of Cluster Resource Services, the allow add to cluster (ALWADDCLU) network attribute is checked to see whether the node being added should be part of the cluster and whether to validate the cluster request through the use of X.509 digital certificates. If validation is required, the requesting node and the node being added must have the following installed on the systems:

- OS/400 option 34 (Digital Certificate Manager)
- Cryptographic Access Provider Product (AC2 or AC3)

Restrictions

1. To use this command you must have *IOSYSCFG authority.
2. You cannot call this command from a cluster resource group exit program.

3. This command cannot be issued on the node being added. You must issue the command from a node in the cluster that has a status of Active. If Cluster Resource Services has not been started on any of the nodes in the cluster, you must issue this command from the node where the cluster was originally created, and the START parameter must be set to *NO.
4. The node being added to the cluster must not already be a member of this or any other cluster. A node can be a member of only one cluster.
5. If the START parameter is set to *YES, the node must be IP reachable (TCP/IP active and the INETD server started).
6. The command will fail if any node in the cluster has a status of Partition.
7. If the START parameter is set to *YES, the potential node version of the node being added must be equal to the current cluster version or up to one level higher than the current cluster version. The potential node version and the current cluster version can be retrieved by using the Display Cluster Information (DSPCLUINF) command.

Required Parameters

CLUSTER

Specifies the name of the cluster to which the node is being added.

cluster-name: Specify the name of the cluster to which the node is being added.

NODE Specifies information about the node which will be placed in the cluster membership list.

Element 1: Node Identifier

A name that uniquely identifies a node.

node-identifier: Specify a name for the system.

Element 2: Interface Address

The cluster interface address is an IP address that is used by Cluster Resource Services to communicate with other nodes in the cluster. A maximum of 2 interface addresses per node can be specified.

Interface-address: Specify an IP address to be used to communicate with other nodes in the cluster. The address is in dotted decimal format.

Optional Parameter

START

Specifies whether or not Cluster Resource Services is to be started on the node being added.

***YES**: Cluster Resource Services will be started on the node.

***NO**: Cluster Resource Services will not be started on the node.

Example for ADDCLUNODE

```
ADDCLUNODE CLUSTER(MYCLUSTER) NODE(NODE1 ('2.5.35.117')) START(*YES)
```

This command adds node NODE1 to the cluster MYCLUSTER. Node NODE1 is added to the cluster membership list and Cluster Resource Services is started on NODE1. Interface address 2.5.35.117 will be used by Cluster Resource Services to communicate with the new node.

Error messages for ADDCLUNODE

*ESCAPE Messages

CPF0001

Error found on &1 command.



ADDCRGDEVE (Add Cluster Resource Group Device Entry) Command Description

ADDCRGDEVE Command syntax diagram

Purpose

The Add Cluster Resource Group Device Entry (ADDCRGDEVE) command is used to add one or more configuration objects representing resilient hardware devices to a device cluster resource group. All devices being added must be able to be switched from one cluster node to another. When the cluster resource group is switched over or failed over to the first backup system, all devices in the device list are moved to the backup system.

If the cluster resource group contains any members of an auxiliary storage pool group, it must contain **all** members before the cluster resource group can be started. All members do not have to be specified at once. Additional members can be added later. If the auxiliary storage pool group has previously been created and clustering can determine which members are in the group, a warning message is sent if some members of the group were not added.

If an exit program is specified for the cluster resource group, the cluster resource group exit program is called with an action code of ADD DEVICE ENTRY on all active nodes in the recovery domain. The cluster resource group status is set to Add Device Entry Pending. If the exit program completes successfully, the cluster resource group status is reset to its value at the time the command was called. If the exit program fails and the cluster resource group cannot be restored to its original condition, the cluster resource group status is set to Indoubt.

Restrictions

1. To use this command you must have *IOSYSCFG authority.
2. This command cannot be called from a cluster resource group exit program.
3. Cluster Resource Services must be active on the node processing the request.
4. The number of configuration objects being added plus the number of configuration objects already in the cluster resource group cannot exceed 256.
5. The configuration objects being added cannot be specified in another cluster resource group.
6. If the primary node does not currently own the specified devices, the command fails with an error message.
7. All nodes in the recovery domain must be active.
8. For the configuration objects specified:
 - a. Only auxiliary storage pool devices are supported.
 - b. You must create the configuration object for the devices being added on all nodes in the recovery domain of the cluster resource group.
 - c. The resource name specified in the configuration object must be the same on all nodes in the recovery domain.
 - d. Devices attached to the same IOP or high-speed link I/O bridge can be specified for only one cluster resource group.
 - e. If devices attached to different IOPs or high-speed link I/O bridges are grouped such as for an auxiliary storage pool, all devices for the affected IOPs or high-speed link I/O bridges must be specified in the same cluster resource group.
 - f. The IOP or high-speed link I/O bridge controlling the devices specified in a cluster resource group must be accessible by all nodes in the cluster resource group's recovery domain. This is verified if

sufficient hardware configuration has been performed so that all nodes are aware of the new hardware. If hardware configuration is incomplete, this is verified when the Start Cluster Resource Group (STRCRG) command is called.

- g. If a data base name is specified in the configuration object, it must be the same on all nodes in the recovery domain.
- h. If a new auxiliary storage pool group is added to an active cluster resource group, all members of the auxiliary storage pool group must be specified.
- i. If a server takeover IP address is specified, it must exist on all nodes in the recovery domain if the cluster resource group is active. The server takeover IP address must be unique within the cluster. It can only be associated with a primary auxiliary storage pool.

Required Parameters

CLUSTER

Specifies the name of the cluster to which the cluster resource group belongs.

cluster-name: Specify the name of the cluster which contains the cluster resource group.

CRG

Specifies the name of the cluster resource group which is to contain the device entries.

cluster-resource-group-name: Specify the name of the cluster resource group.

CFGOBJ

Specifies detailed information about the resilient devices to be added to the cluster resource group.

Element 1: Configuration Object

The name of the auxiliary storage pool device description object which can be switched between the nodes in the recovery domain. An auxiliary storage pool device description can be specified in only one cluster resource group.

configuration-object: Specify the device description name.

Element 2: Configuration Object Type

Specifies the type of configuration object specified with configuration object.

***DEV**: Device description.

Element 3: Configuration Object Online

Specifies what configuration action to take when the cluster resource group is switched over or failed over to a backup node. The configuration object can be varied on and start the server takeover IP address or leave the configuration object varied off and the server takeover IP address inactive and does **not** start or end the server takeover IP address when device ownership is moved to another node. This attribute does **not** vary the device on or off when the cluster resource group is started or ended.

***OFFLINE**: Do not vary the configuration object on and do not start the server takeover IP address.

***ONLINE**: Vary the configuration object on and start the server takeover IP address.

***PRIMARY**: This is a secondary ASP. Vary on processing is determined by the corresponding primary ASP for this ASP group.

Element 4: Server Takeover IP Address

Specifies a takeover IP address for servers associated with the relational database name in the device description for an auxiliary storage pool. This element is optional and can only be specified for a primary auxiliary storage pool. If specified, the address must be presented in dotted decimal

format. The specified address must exist on all nodes in the recovery domain if the cluster resource group is active. If not specified, or for a secondary or UDFS auxiliary storage pool, the element must be set to *NONE.

***NONE:** There is no server takeover IP address associated with the relational database name in the device description for an auxiliary storage pool.

server-takeover-ip-address: Specify the server takeover IP address for the relational database name.

Example for ADDCRGDEVE

```
ADDCRGDEVE CLUSTER(MYCLUSTER) CRG(MYCRG) CFGOBJ((IASP01 *DEVD *ONLINE *NONE))
```

This command adds an ASP device description object, IASP01, to the device list of the resilient device cluster resource group MYCRG in the cluster MYCLUSTER. If MYCRG is switched over or failed over to a backup node, the device will be varied on to the new primary.

Error messages for ADDCRGDEVE

*ESCAPE Messages

CPF0001

Error found on &1 command.



ADDCRGNODE (Add Cluster Resource Group Node Entry) Command Description

ADDCRGNODE Command syntax diagram

Purpose

The Add Cluster Resource Group Node Entry (ADDCRGNODE) command is used to add a new node to the recovery domain of an existing cluster resource group. The node can be added as another backup node, as a replicate node, or as the new primary node in the cluster resource group's recovery domain. A node can only be added as a primary node if the cluster resource group has a status of Inactive. When a new node is added as the primary, the old primary node becomes the last backup. If the cluster resource group has a status of Active, a node can be added as either a backup or a replicate. This command causes the roles of all nodes to be updated in the preferred recovery domain and the current recovery domain.

A node can be added to a resilient device cluster resource group even if it has no device entries. Device entries must be added using the Add Cluster Resource Group Device Entry (ADDCRGDEVE) command before the cluster resource group can be started. If the node being added to a device cluster resource group is to become the new primary node, ownership of the devices specified are switched from the current primary to the new primary if none of the devices are varied on for the current primary. If any devices are varied on, an error message is returned. Devices are not varied on after the ownership is switched.

Restrictions

1. To use this command you must have *IOSYSCFG authority.
2. This command cannot be called from a cluster resource group exit program.
3. The node being added to the recovery domain must not already be a member of this cluster resource group's recovery domain.
4. To add a node as the primary, the cluster resource group must be Inactive.

5. Cluster Resource Services must be active on the node processing the request.
6. There must be at least one active node in the recovery domain.
7. The node being added must be active in the cluster.
8. The cluster resource group exit program must exist on each node in the recovery domain, including the node being added.
9. If a node is being added to a resilient device cluster resource group:
 - a. A node can be added to a cluster resource group even if it has no device entries. Device entries must be added using the Add Cluster Resource Group Device Entry API before the cluster resource group can be started.
 - b. The node must be in the same device domain as the other nodes in the recovery domain.
 - c. The configuration objects for the device resources in the cluster resource group must exist on the node being added and the resource names in the configuration objects must be the same as the resource names used by the configuration objects on the existing nodes in the recovery domain. The node being added must be able to access the hardware resources represented by the configuration objects in the cluster resource group.
 - d. If a data base name is specified in the configuration objects in the cluster resource group, it must be the same on the node being added.
 - e. All members of an auxiliary storage pool group must be configured in the cluster resource group before ownership of the auxiliary storage pool can be changed.
 - f. If a server takeover IP address is specified in the cluster resource group and the cluster resource group is active, the server takeover IP address must exist.

Required Parameters

CLUSTER

Specifies the name of the cluster where the cluster resource group exists.

cluster-name: Specify the name of the cluster where the cluster resource group exists.

CRG Specifies the name of the cluster resource group that will have the new node added to its recovery domain.

cluster-resource-group-name: Specify the name of the cluster resource group.

NODE Specifies the node being added to the recovery domain of the cluster resource group specified. The node specified must be in the cluster and must be unique in the recovery domain of the cluster resource group specified.

node-identifier: Specify the name of the cluster node.

Optional Parameter

RCYDMN

Specifies the role of the node in the recovery domain of the cluster resource group.

Element 1: Node Role

Specifies what role the node will have in the recovery domain.

***BACKUP**: The new node will be added as a backup node.

***PRIMARY**: The new node will be added as the primary node. The cluster resource group must have a status of Inactive.

***REPLICATE**: The new node will be added as a replicate node. Replicate nodes are not ordered.

Element 2: Backup Sequence Number

Specifies the backup order for a node with role of *BACKUP. If there is already a node

with the same backup order, the new node is inserted in the position requested. At the completion of the request, the nodes with backup roles will be sequentially renumbered from the first backup to the last. The first backup will always be 1.

***LAST:** The new node will be added as the last backup in the recovery domain.

backup-sequence-number: Specify the backup sequence number.

Example for ADDCRGNODE

```
ADDCRGNODE CLUSTER(MYCLUSTER) CRG(MYCRG) NODE(NODE1) ROLE(*BACKUP 3)
```

This command adds node NODE1 to the recovery domain of cluster resource group MYCRG in the cluster called MYCLUSTER. The node is added as the third backup node. Any existing backup nodes will be renumbered sequentially.

Error messages for ADDCRGNODE

*ESCAPE Messages

CPF0001

Error found on &1 command.



ADDCMDCRQA (Add Command Change Request Activity) Command Description

Note: To use this command, you must have the 5722-SM1 (System Manager for iSeries) licensed program installed.

ADDCMDCRQA Command syntax diagram

Purpose

The Add Command Change Request Activity (ADDCMDCRQA) command adds an activity to a change request description to run a command on one or more managed systems.

The activity can be conditioned so that it only runs after one or more other activities have completed (successfully or unsuccessfully). The activity can also be scheduled to run at a date and time in the future.

Restrictions:

1. You must have *CHANGE authority to the change request description and *EXECUTE authority to the library.
2. If a node list (NODL) value is specified, the node list can only contain entries that have a value of *SNA for the address type.

Notes:

The following notes provide information on how the command works.

1. Authorization to the product specified on the activity is not verified until the activity runs.
2. All conditions must be satisfied before the activity can run.
3. The start times indicate when the activity can be started. Actual start times can be later due to network and system delays.

Required Parameters

CRQD Specifies the change request description object name.

The possible library values are:

***LIBL:** All of the libraries in your library list and in the system portions of the job's library list are searched.

***CURLIB:** The current library for the job is used to locate the object.

library-name: Specify that only the library named in this parameter is searched.

change-request-description: Specify the name of the change request description object.

ACTIVITY

Specifies the name of the activity to add to the change request description.

***GEN:** An activity name is generated. The activity name is of the form QACTxxxxxx where xxxxxx is the first multiple of ten not already being used.

***LAST:** The activity is the last to run in the change request. When *LAST is specified for the activity (ACTIVITY) parameter, the condition (COND) parameter and the start time (STRTIME) parameter cannot be specified. Only one activity named *LAST can exist in the change request description.

activity-name: Specify a 10-character activity name.

CMD Specifies the CL command to run. The command can be any command that can be run in batch. The command must follow the OS/400 CL command format of 1 to 10 characters with the first character in alphabetical order and the other nine characters alphanumeric.

command-string: Specify the command to run on the managed system. Command prompting support is available if you press F4.

Optional Parameters

NODL Specifies that the node list parameter is the object name that contains a list of systems that are the destinations for the activity. This parameter cannot be specified if the control point name (CPNAME) parameter is specified.

***NONE:** The systems on which this activity is to be performed are not specified by a node list. Individual control point names must be specified.

The possible library values are:

***LIBL:** All of the libraries in the user and system portions of the job's library list are searched for the node list object.

***CURLIB:** The current library for the job is used to locate the node list object.

library-name: Specify the name of the library to be searched.

node-list-name: Specify the node list object name containing the list of systems on which the activity is to be performed.

CPNAME

Specifies the APPN control point names of the managed systems on which this activity is to be performed. Control point names cannot be specified if the node list (NODL) parameter is specified.

The possible values are:

***LOCAL:** The local system is identified as the target system. If *LOCAL is specified, the command is run on the local system. Any spooled files created remain on the system.

***NONE:** The systems on which this activity is to be performed are not identified individually. A node list must be specified.

***NETATR:** The network ID of the local system is used. This is useful when the node being specified is in the same network as the local system.

network-identifier: Specify the APPN network identifier of the managed system on which the activity is to be performed.

control-point-name: Specify the APPN control point name of the managed system on which the activity is to be performed.

COND Specifies which conditions must be met before this activity can be performed. Each condition identifies an activity that must run before this activity and the value the end code from that activity must have to allow this activity to run. The default condition is that the previous activity (in alphabetical order) must complete successfully before this activity can be run.

Element 1: Conditioning Activity

The activity that must be run before this activity.

***PRV:** This activity is conditioned on the previous activity. Activities are ordered alphabetically by activity name. If the activity being added is the first activity, a previous activity does not exist and any condition with *PRV is marked as having been met.

conditioning-activity-name: Specify the name of the activity that must run before this activity. The activity name specified in the activity (ACTIVITY) parameter cannot be specified in the conditioning activity name. An activity cannot be conditioned on itself.

generic-conditioning-activity-name:* Specify the generic name of the activities that must run before this activity.

Element 2: Relational Operator

This element is the relational operator to use when comparing the end code from the conditioning activity.

***EQ:** Equal

***GT:** Greater than

***LT:** Less than

***NE:** Not equal

***GE:** Greater than or equal

***LE:** Less than or equal

Element 3: Condition Code

This element is the value compared to the actual end code of the conditioning activity.

***SUCCESS:** The activity ended successfully (0 <= end code <= 9). This end code can only be specified with relational operator *EQ or *NE.

***FAIL:** The activity failed (10 <= end code <= 89). This end code can only be specified with relational operator *EQ or *NE.

***NOTRUN:** The activity never started (90 <= end code <= 99). This end code is only specified with relational operator *EQ or *NE.

***ANY:** The activity ended with any end code. This end code is only specified with relational operator *EQ.

end-code: Specify an integer value (0-99) that indicates the result of an activity (success or failure). The end code ranges and descriptions are:

00 Activity completed successfully.

01-09 Activity completed with warning messages.

10-29 Activity did not complete successfully.

30-39 Activity was canceled by a user before it completed.

- 30 = Activity ended with *CNTRLD option
- 35 = Activity ended with *IMMED option
- 39 = Activity ended with *FRCFAIL option

40-49 Activity was not run due to errors detected by the application.

- 40 = Activity not run for security reasons

90-99 Activity was not run because conditions or schedules were not met.

- 95 = Scheduled start time expired
- 99 = Conditions cannot be met

Element 4: Condition Mode

This element indicates which systems the conditioning activity must have completed on before this activity can be performed.

***ALLNODES:** The conditioning activity specified must complete on all nodes before this activity can run.

***SAMENODE:** When the conditioning activity specified completes for a given node, the activity specified on the ACTIVITY parameter may run for that same node even though the conditioning activity specified may not have completed for all other nodes. In the case where this activity lists a node not in the conditioning activity, this activity may run for that node; the condition is ignored.

***NONE:** There are no conditions for this activity.

STRTIME

Specifies the date and time when this activity can be started on the central site system. The current date and time values and next date values are determined when the change request is submitted.

Element 1: Start After Time

***CURRENT:** This activity can start any time on or after the time when the change request is submitted.

start-after-time: Specify the time when this activity can start. The time can be entered as 4 or 6 digits (hhmm or hhmmss), where hh = hours, mm = minutes, and ss = seconds. Seconds are optional. The time can be specified with or without a time separator such as a colon (:). With a time separator, specify a string of 5 or 8 digits (hh:mm or hh:mm:ss).

Element 2: Start After Date

***CURRENT:** This activity can start on or after the date on which the change request is submitted.

***NEXT:** The activity can start on any date after the date the change request is submitted.

start-after-date: Specify the date after this activity can start. The date must be specified in the job date format.

Element 3: Start Before Time

This element is ignored if the start before date is *ANY.

***ANY:** The activity can start at any time on or before the start before date.

***CURRENT:** The activity must start before the time at which the change request was submitted on the date specified on the start before date element.

start-before-time: Specify the time before which the activity must start. If the activity cannot be started before this time, it never starts. The time can be entered as 4 or 6 digits (hhmm or hhmmss) where hh = hours, mm = minutes, and ss = seconds. Seconds are optional. The time can be specified with or without a time separator such as a colon (:). With a time separator, specify a string of 5 or 8 digits (hh:mm or hh:mm:ss).

Element 4: Start Before Date

***ANY:** The activity can start at any time after the start after time and the start after date.

***CURRENT:** The activity must start on the date the change request is submitted.

***NEXT:** The activity must start by the day after the date the change request is submitted.

start-before-date: Specify the date before the activity must start. If the activity cannot be started by this date, it never starts. The date must be specified in the job date format.

RMTSTRTIME

Specifies the date and time when the activity can begin running on the managed system. The current date and time values, and the next date values are determined when the activity begins running at the central site system based on the central site date and time.

Element 1: Time Zone

The time zone of the remote start time.

***LCLSYS:** The remote start time is specified in the time zone of the central site system.

***MGDSYS:** The remote start time is specified in the time zone of the managed system.

Element 2: Start After Time

This is the definition of the time after which the activity is to start.

***CURRENT:** This function can start on the managed system at any time on or after the time this activity is started on the central site system on the date specified in element 3.

start-after-time: Specify the time after which this function can start on the managed system. The time can be entered as 4 or 6 digits (hhmm or hhmmss) where, hh = hours, mm = minutes, and ss = seconds. Seconds are optional. The time can be specified with or without a time separator. With a time separator, specify a string of 5 or 8 digits (hh:mm or hh:mm:ss).

Element 3: Start After Date

***CURRENT:** This function starts on the managed system on any date on or after the date this activity starts on the central site system.

***NEXT:** This function starts on the managed system on any date after the date the activity starts on the central site system.

start-after-date: Specify the date after the functions start on the managed system. The date must be specified in the job date format.

Element 4: Start Before Time

This element is ignored if the start before date is *ANY.

***ANY:** The activity can start at any time on or after the start time.

***CURRENT:** The activity must start before the time at which the change request was submitted on the date specified on the start before date element.

start-before-time: Specify the time before which the activity must be started. If the activity cannot be started before this time, it never starts. The time can be entered as 4 or 6 digits (hhmm or hhmmss) where hh = hours, mm = minutes, and ss = seconds. Seconds are optional. The time can be specified with or without a time separator. With a time separator, specify a string of 5 or 8 digits (hh:mm or hh:mm:ss).

Element 5: Start Before Date

***ANY:** The activity can start at any date on or after the start date.

***CURRENT:** The activity must start on the date the change request was submitted.

***NEXT:** The activity must start by the day after the date the change request was submitted.

start-before-date: The date before which the activity must be started. If the activity cannot be started by this date, it never starts. The date must be specified in the job date format.

Note:

The special values *CURRENT and *NEXT cannot be specified for the date and the time when the time zone value *MGDSYS is specified.

RTNSPLF

Specifies whether the output spooled file from the remote command is returned.

***YES:** The spooled files created from the remote command are returned from the managed system. The spooled files from all of the nodes are combined into one spooled file that can be viewed by displaying the command activity details of the change request.

***NO:** The output data is not returned from the managed system.

***FAIL:** The spooled file job log is returned from the managed system if the command fails when running.

USRPRF

Specifies the user profile under which the command runs at the remote systems. If the managed system is running the NetView Remote Operations Agent/400 product, this parameter is ignored at the managed systems.

***NONE:** No user profile is specified. The default user profile is used on each managed system.

user-profile: Specify the name of the user profile.

PASSWORD

Specifies the password for the remote user profile.

- Managed systems at releases prior to V5R1M0 accept only uppercased passwords up to 10 characters long. If a longer password is entered, SMU18A2 message with 100B0007 SNA sense code is returned, indicating that the request was rejected.
- Managed systems at release V5R1M0 and later, running with QPWLVL system value:
 - 0 or 1 — truncate the received passwords to 10 characters.
 - 2 or 3 — accept passwords up to 128 characters.

***NONE:** No password is specified.

***USRPRF:** The password is the same as the user profile.

password: Specify the password for the user profile.

TEXT Specifies the activity description.

***GEN:** A description is generated based on the action specified.

text-description: Specify a 50-character description of the activity.

ENCODE

Specifies whether or not the command, user profile, and password are encoded when sent to the managed systems. If the managed system is running the NetView Remote Operations Agent/400 product, this parameter must be set to *NO.

***NO:** The command, user profile, and password are not encoded when the request is sent to the managed systems.

***YES:** The command, user profile, and password are encoded when the request is sent to the managed systems. The remote command key, which is a managed system attribute, must be specified on both the central site system and the managed system. This attribute can be changed using the Change Managed System Attributes (CHGMGDSYSA) command. The remote command key must be the same on the central site system and the managed system.

HOLD Specifies that the activity be held when the change request is submitted.

***NO:** The activity is not held. It runs when all conditions are met at the start time.

***YES:** The activity is held for all nodes when the change request is submitted. It must be released by you before it runs.

Examples for ADDCMDCRQA

Example 1: Adding an Activity

```
ADDCMDCRQA CRQD(MYLIB/CR1) ACTIVITY(ACT01)
CMD(STRSBS QCMN)
```

This example shows how activity ACT01 is added to change request description CR1 in library MYLIB to start the QCMN subsystem on the central site system. The activity runs as soon as the change request is submitted.

Example 2: Adding an Activity for a Node list

```
ADDCMDCRQA CRQD(MYLIB/CR2) ACTIVITY(ACT01)
  CMD(STRSBS QSNADS) NODL(MYLIB/STORES)
  CPNAME(*NONE) RTNSPLF(*FAIL)
```

This example shows how activity ACT01 starts the subsystem QSNADS on the systems identified in the STORES node list in the library MYLIB. The example also asks for the spooled file to be returned to the central site system if the command fails.

Example 3: Adding an Activity for Two Systems

```
ADDCMDCRQA CRQD(MYLIB/CR3) ACTIVITY(ACT01)
  CMD(PRODLIB/RUNREPORTS) CPNAME((STORENET STOREA)
  (STORENET STOREB)) USRPRF(REPORTOPER)
  PASSWORD(OPERPASS) ENCODE(*YES)
```

This example shows how activity ACT01 runs the detail reports for STOREA and STOREB in the STORENET network. The report runs under the REPORTOPER user profile on the managed systems. Because the user profile and a password are supplied, the request will be encoded when sent to the managed system.

Error messages for ADDCMDCRQA

*ESCAPE Messages

CPF9681 E

Activity &1 already exists.

CPF9684 E

Start after time of start time not valid.

CPF9685 E

Start before time of start time not valid.

CPF968A E

Activity name &1 not valid.

CPF968E E

Condition list or start time cannot be specified.

CPF9691 E

Start after date of start time not valid.

CPF9692 E

Start before date of start time not valid.

CPF9696 E

Generated activity limit exceeded.

CPF9697 E

Condition activity cannot equal activity name.

CPF9698 E

Maximum size of change request description exceeded.

CPF9699 E

Start time not valid.

CPF9801 E

Object &2 in library &1 not found.

CPF9802 E

Not authorized to object &2/&3 type &5.

CPF9803 E

Unable to allocate object &2/&3 type &5.

CPF9810 E

Library &1 not found.

CPF9838 E

User profile storage limit exceeded.

SMU1681 C

Activity &3 added.

SMU168C D

Node list or control point names required.

SMU16xx E

Start after time of remote start time not valid.

SMU16xx E

Start before time of remote start time not valid.

SMU16xx E

Start after date of remote start time not valid.

SMU16xx E

Start before date of remote start time not valid.d

SMU16xx E

Remote start time not valid.



ADDCMNE (Add Communications Entry) Command Description

ADDCMNE Command syntax diagram

Purpose

The Add Communications Entry (ADDCMNE) command adds a communications entry to an existing subsystem description.

Each entry specifies a communications device, device type, or remote location that can enter work into the subsystem. The subsystem can allocate a communications device, if the device is not currently allocated to another subsystem or job. A communications device that is currently allocated may eventually be de-allocated, making it available to other subsystems. For information regarding reallocation of

communications devices to other subsystems, see the Work Management  book.

Restrictions:

1. The user must have object operational and object management authorities for the communications entry.
2. A subsystem cannot allocate a communications device that is currently allocated to another subsystem or job.

Required Parameters

SBSD Specifies the qualified name of the subsystem description to which the communications entry is being added or in which it is being changed.

The name of the subsystem description can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

subsystem-description-name: Specify the name of the subsystem description to which the communications entry is added.

DEV Specifies the name of the device description or the type of the device being used with this communications entry.

Note: A user must specify either this parameter or the RMTLOCNAME parameter, but not both.

***ALL:** All communications devices can be used with this communications entry.

***APPC:** All advanced program-to-program communications devices can be used with this communications entry. The devices created with the CRTDEVAPPC command are used by the iSeries 400 server APPN support.

***ASYNC:** All asynchronous communications devices, including those created with the CRTDEVASC command, can be used with this communications entry. This value is valid only if MODE(*ANY) is specified.

***BSCSEL:** All binary synchronous equivalency link communications devices, including those created with the CRTDEVBSC command, can be used with this communications entry. This value is valid only if MODE(*ANY) is specified.

***FINANCE:** All FINANCE communications devices, including those created with the CRTDEVFNC command, are used with this communications entry. This value is valid only if MODE(*ANY) is specified.

***INTRA:** All INTRA communications devices, including those created with the CRTDEVINTR command, are used with this communications entry. This value is valid only if MODE(*ANY) is specified.

***RETAIL:** All RETAIL communications devices, including those created with the CRTDEVRTL command, are used with this communications entry. This value is valid only if MODE(*ANY) is specified.

***SNUF:** All SNA upline facility communications devices, including those created with the CRTDEVSNUF command, can be used with this communications entry. This value is valid only if MODE(*ANY) is specified.

device-name: Specify the device description name or the type of device to use with this communications device entry. The name specified on the CRTDEVxxx command associated with this device description name is used.

generic-device-name*: Specify the generic name of the device. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. See generic names for additional information.

RMTLOCNAME

Specifies the name of the remote location that is used with this object.

Note:

The remote location name specified in the CRTDEVXXX command can be used here. No validity checking is done on the remote location name.

The user must specify either this parameter or the DEV parameter, but not both.

Optional Parameters

JOB Specifies the qualified name of the job description used for jobs created as a result of program start requests and processed through this communications entry. If the job description does not exist when the communications entry is added, a library qualifier must be specified because the qualified job description name is kept in the subsystem description.

***USRPRF**: The job description name specified in the user profile of the user that made the program start request is used for jobs that are processed through this communications entry.

***SBSD**: The job description having the same name as the subsystem description (specified by the SBSDB parameter) is used for jobs processed through this communications entry.

The name of the job description can be qualified by one of the following library values:

***LIBL**: All libraries in the job's library list are searched until the first match is found.

***CURLIB**: The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

job-description-name: Specify the qualified name of the job description that is used for the jobs processed through this communications entry.

DFTUSR

Specifies the default user profile used for a program start request that contains no password (either all blanks or a zero length password) and user profile name. This user profile is not used for program start requests that contain a password or specify a user profile (either valid or not valid).

***NONE**: No user profile is specified as a default.

***SYS:** User program start requests are treated the same as DFTUSR(*NONE). For system-generated program start requests, the correct user profile is used.

user-profile-name: Specify the name of the user profile that is used for all program start requests that enter the system through this communications entry and that contain no password or user profile name.

Note:

The names QSECOFR, QSPL, QDOC, QDBSHR, QRJE, and QSYS are not valid entries on this parameter.

MODE Specifies the mode name of the communications device or remote location name used with this communications entry.

***ANY:** Any available modes defined to the communications device or remote location are allocated to the subsystem. If the communications device does not have defined modes, the communications device itself is allocated to the subsystem.

mode-name: Specify a mode name of the communications device or remote location used with this communications entry.

MAXACT

Specifies the maximum number of program start requests that can be active at the same time through this communications entry. More information is in Commonly used parameters.

***NOMAX:** There is no maximum number of jobs that can be active at the same time.

maximum-active-jobs: Specify the maximum number of jobs that can be active at the same time through this communications entry.

Example for ADDCMNE

```
ADDCMNE  SBS1(ALIB/SBS1)  DEV(COMDEV)
```

This command adds a communications entry for the APPC device named COMDEV to the subsystem description SBS1, which resides in library ALIB. The DFTUSR parameter defaults to *NONE, which means that no jobs may enter the system through this entry unless valid security information is supplied on the program start request.

Error messages for ADDCMNE

***ESCAPE Messages**

CPF1619

Subsystem description &1 in library &2 damaged.

CPF1691

Active subsystem description may or may not have changed.

CPF1697

Subsystem description &1 not changed.

ADDCOMSNMP (Add Community for SNMP) Command Description

ADDCOMSNMP Command syntax diagram

Purpose

The Add Community for SNMP (ADDCOMSNMP) command defines an SNMP community profile and adds it to the SNMP agent community list. An SNMP agent uses a community profile to determine whether or not to honor a request sent by an SNMP manager. The community profile consists of a community name,

an object access specification, and a list of the SNMP managers that are part of the community. The combination of the community name (COM) and the translate to ASCII community (ASCII.COM) parameters defines a community.

Multiple community profiles, each having a unique community name may exist in the SNMP agent community list at one time. Similarly, the same internet address may appear in more than one community profile.

The OS/400* SNMP agent does not support community views. A view is a subset of the objects in the management information base (MIB). Each OS/400 community consists of all of the objects in the MIB.

Restrictions: An SNMP manager sends three types of requests: get, get-next, and set. Get and get-next requests are used to read management information base (MIB) variables, and a set request is used to modify MIB variables. For a request from an SNMP manager to be accepted by the iSeries 400 SNMP agent, all of the following must be true:

1. The community name in the SNMP manager request specifies a defined community.
2. The internet address of the manager that sent the request must be listed in the community profile.
3. For a set request, the community object access must allow write operations to occur. For a get request or get-next request, read operations must be allowed.
4. For a set request, the object specified in the request must be able to be changed. For a get request or get-next request, the object must be readable.

Required Parameter

COM Specifies the name of the SNMP community being added. Each SNMP community name must be unique.

community-name: Specify the name of the SNMP community being added. The name may contain characters that cannot be displayed (for example, X'60619E').

Optional Parameters

ASCII.COM

Specifies whether the community name is translated to ASCII characters when the community profile is added to the SNMP agent community list.

***YES:** The community name is translated to ASCII characters when the community profile is added to the SNMP agent community list. This value should be specified if the SNMP manager system defines its community names entirely of ASCII characters. An error message is sent if the community name cannot be translated to ASCII characters.

***NO:** The community name is not translated to ASCII characters when the community profile is added to the SNMP agent community list. This value should be specified if the SNMP manager system defines its community names using EBCDIC characters or characters that cannot be displayed.

INTNETADR

Specifies the internet addresses of the SNMP managers that are part of this community.

***ANY:** Allow any SNMP manager to be part of this community.

manager-internet-address: Specify the internet address of the SNMP manager. The internet address is specified in the form *nnn.nnn.nnn.nnn*, where *nnn* is a decimal number ranging from 0 through 255. An internet address is not valid if it has a value of all binary ones or all binary zeros for the network identifier (ID) portion or the host ID portion of the address. If the internet address is entered from a command line, the address must be enclosed in apostrophes. Up to 300 unique internet addresses may be specified. The same internet address may appear in more than one community profile.

OBJACC

Specifies the object access for the community.

***SNMPATR:** The object access defined with the Change SNMP Attributes (CHGSNMPA) command is used for this community.

***READ:** Allow SNMP managers that are part of this community to read all management information base (MIB) objects with get or get-next requests. Modification of MIB objects by SNMP managers is not permitted.

***WRITE:** Allow SNMP managers that are part of this community to change all MIB objects that are able to change with set requests. Specifying *WRITE implies *READ access.

***NONE:** Do not allow SNMP managers that are part of this community any access to MIB objects.

LOGSET

Specifies whether set requests from SNMP managers in this community are logged in journal QSNMP in library QUSRSYS.

***SNMPATR:** The value defined with the Change SNMP Attributes (CHGSNMPA) command is used for this community.

***YES:** Set requests are logged.

***NO:** Set requests are not logged.

LOGGET

Specifies whether get requests and get-next requests from SNMP managers in this community are logged in journal QSNMP in library QUSRSYS.

***SNMPATR:** The value defined with the Change SNMP Attributes (CHGSNMPA) command is used for this community.

***YES:** Get requests and get-next requests are logged.

***NO:** Get requests and get-next requests are not logged.

Example for ADDCOMSNMP

```
ADDCOMSNMP  COM(ROCHESTER)
             INTNETADR('8.6.5.4' '8.6.5.3')
             OBJACC(*WRITE)
```

This command adds the community ROCHESTER to the SNMP agent community list. SNMP managers with internet addresses 8.6.5.4 and 8.6.5.3 are the only managers in the community and are able to change all MIB objects.

Error messages for ADDCOMSNMP

*ESCAPE Messages

TCP4001

Error occurred accessing SNMP configuration information.

TCP4008

Community already exists. Reason code &3.

TCP8050

*IOSYSCFG authority required to use &1.

ADDCFGLE (Add Configuration List Entries) Command Description

ADDCFGLE Command syntax diagram

Purpose

The Add Configuration List Entries (ADDCFGLE) command adds entries to a configuration list.

Note:

You can also use an option on the full screen entry display of the Work with Configuration Lists (WRKCFGL) command to add, change, or remove entries in an existing list except for the configuration lists of type *APPNDIR, *APPNSSL, and *SNAPASTHR.

Required Parameters

TYPE Specifies the type of configuration list entry being added.

***APPNDIR:** An advanced peer-to-peer networking* (APPN*) directory search filter configuration list is used.

***APPNLCL:** An APPN local location list is used. Up to 476 APPN local location entries are allowed in the configuration list (using the CHGCFGL and CRTCFGL commands).

***APPNRMT:** An APPN remote location list is used. Up to 1898 APPN remote location entries are allowed in the configuration list (using the CHGCFGL and CRTCFGL commands).

***APPNSSL:** An APPN session end point filter configuration list is used.

***ASYNCADR:** An asynchronous network address list is used. Up to 294 asynchronous network address entries are allowed in the configuration list.

***ASYNCLC:** An asynchronous remote location list is used. Up to 32000 asynchronous remote location entries are allowed in the configuration list.

***RTLPASTR:** A retail pass-through list is used. Up to 450 retail pass-through entries can be specified in the configuration list.

***SNAPASTHR:** An SNA pass-through list is used. Up to 254 SNA pass-through entries can be specified in the configuration list.

CFGL Specifies the name of the configuration list. This parameter is valid only when *ASYNCADR is specified on the TYPE parameter. Only one of the other configuration list types is allowed on a system. The list types have system-supplied names: QAPPNDIR, QAPPNLCL, QAPPNRMT, QAPPNSSL, QASYNCADR, QASYNCLC, QRTLPASTR, QSNAPASTHR.

Optional Parameters

APPNLCL

Specifies the APPN local location entry. This value is required if *APPNLCL is specified for the TYPE parameter.

Up to 476 entries can be specified for this parameter, but only 50 entries can be specified at a time. An entry consists of a value from the local location name and the entry description.

Element 1: Local Location Name

local-location-name: Specify the location name of the local system. This name is used by APPN support to determine whether the request being received in is for this system or another system in the network. The local location name must be unique; it cannot already exist as a remote location name that is used by configuration list QAPPNRMT and it cannot be specified on another system as a local location in the same APPN network.

Element 2: Text Description

***BLANK:** Text is not specified.

'entry-description': Specify a short description of 20 characters or less for each local entry.

APPNRMT

Specifies the APPN remote location entry. This value is required if *APPNRMT is specified for the

TYPE parameter. Up to 1898 entries can be specified for this parameter, but only 50 entries can be specified at a time. An entry consists of a value from each of the following elements. A value must be specified for each of the 11 elements for each entry.

Element 1: Remote Location Name

***ANY:** The system potentially accepts all requests sent to it.

generic-remote-location-name:* Specify the generic name (part of a name followed by an asterisk) of the remote location(s) to be changed. The generic location name allows one directory entry to be defined for all locations, on a single control point, with a name that matches the characters preceding an *.

remote-location-name: Specify the full name of a remote location.

Element 2: Remote Network Identifier

***NETATR:** The LCLNETID value specified in the system network attributes is used.

***NONE:** The remote network identifier is not specified.

remote-network-identifier: Specify the network identifier of the network in which the remote location resides.

Element 3: Local Location Name

***NETATR:** The LCLLOCNAME value specified in the system network attributes is used.

local-location-name: Specify the location name on the local system. This name is used by APPN support to match a local/remote location pair entry.

Element 4: Control Point Name

***NONE:** There is no control point name.

control-point-name: Specify the name of the control point that provides network functions for the remote location. This field is required if the entry is either a generic name or *ANY.

Element 5: Control Point Network Identifier

***NETATR:** The LCLNETID value specified in the system network attributes is used.

control-point-network-identifier: Specify the network identifier in which the control point resides.

Element 6: Location Password

***NONE:** There is no location password.

location-password: Specify the password that is used when establishing sessions on the local location/remote location name pair. It must be an even number of hexadecimal characters.

Element 7: Location Security

Specifies how security information is handled for program start requests received from remote systems. The value is sent to the remote system when sessions are established. It is used in determining how allocate or evoke requests should be built. The value only applies to conversations started with the SECURITY(SAME) level of security.

***NO:** The remote system is not a secure location. Security validation done by the remote system is not accepted. SECURITY(SAME) conversations are treated as SECURITY(NONE). No security information will be sent with allocate or evoke requests.

***YES:** The remote system is a secure location and the local system will accept security validation done by remote systems. For SECURITY(SAME) conversations, the local system allows the remote system to verify user passwords. On the remote system, user IDs are retrieved from the operating system. The user IDs are then sent with an already verified indicator in the allocate or evoke requests.

***VFYENCPWD:** The remote system is not a secure location. For SECURITY(SAME) conversations, the remote system is not allowed to send the already verified indicator. On the remote system, user IDs and passwords are retrieved from the operating system. Passwords are then encrypted and sent with the user IDs in the allocate or evoke requests, to be verified by the local system. This value should only be used if the remote system is using OS/400 V3R2M0 or later. If the remote system does not support password protection then session establishment will not be allowed. For remote systems that support password protection, but do not support verification of encrypted passwords (VFYENCPWD), conversations will be treated as SECURITY(NONE).

Element 8: Single Session Location

This value specifies whether the connection between the local location and remote location is a single session connection.

***NO:** A single session connection is not made between the local and the remote location.

***YES:** A single session connection is made between the local location and the remote location.

Element 9: Locally Controlled Session

This value specifies whether the single session connection between the local location and remote location is locally controlled.

***NO:** The single session connection does not have to be locally controlled.

***YES:** The single session connection is locally controlled.

Element 10: Pre-established Session

This value specifies whether the single session is automatically bound when the mode is started between the local location and remote location.

***NO:** The single session connection is not automatically made between the local and remote location.

***YES:** The single session connection is automatically made between the local and remote location.

Element 11: Remote Entry Description

***BLANK:** Text is not specified.

'entry-description': Specify a short description of 20 characters or less, enclosed in apostrophes, for each remote entry.

Element 12: Number of Single-Session Conversations

10: The number of single session conversations allowed for this device description is ten.

single-session-conversations: Specify the number of conversations allowed for a single session of this device description. Valid values range from 1 through 512.

Note:

The combination of remote location name, remote network identifier, and local location name must be unique. Also, the remote location name cannot already exist as a local location in configuration list QAPPNLCL, or as the current value for either the LCLLOCNAME or the CPNAME network attribute.

ASYNCADRE

Specifies the asynchronous network address entry. This value is required if *ASYNCADR is specified for the TYPE parameter. Up to 50 entries can be specified for this parameter.

The following values make up an asynchronous network address entry.

Element 1: Network Address

network-address: Specify the asynchronous remote network address.

Element 2: Dial Retries

2: The LZ algorithm with the 12-bit code for repeated substrings in the data stream is used. These codes refer to entries in a common dictionary, created as the data flows between the sender and receiver. The LZ algorithms require storage and extra processing time. The LZ12 requires the most storage and processing time of the LZ algorithms; however, it compresses the data stream the most.

dial-retry: Specify the number of times dialing is retried (because errors occur) before the next number on the list is dialed. Valid values range from 1 through 255.

Element 3: Text Description

***BLANK**: Text is not specified.

'entry-description': Specify a short description of 20 characters or less for each network address entry.

ASYNCCLOC

Specifies the asynchronous remote location entry. This value is required if *ASYNCCLOC is specified for the TYPE parameter. Up to 50 entries can be specified for this parameter.

The following values make up an asynchronous network address entry.

Element 1: Remote Location Name

remote-location-name: Specify the name of the remote location. This name, when combined with the remote location identifier, determines whether an incoming call is accepted. The specified name must be unique.

Element 2: Remote Location Identifier

remote-location-identifier: Specify the identifier of the remote location. When this identifier is combined with the remote location name, it determines whether an incoming call is accepted. This identifier must be the same as the remote system has for its local identifier.

Element 3: Description of Remote Location Entry

***BLANK**: Text is not specified.

'entry-description': Specify a short description of 20 characters or less for each remote location entry.

RTLPASTRHRE

Specifies the retail pass-through entry. This value is required if TYPE(*RTLPASTRHRE) is specified. Up to 50 entries can be specified for this parameter.

The following values make up a retail pass-through entry:

Element 1: Retail Device Name

retail-device-name: Specify the name of the retail device to use for the pass-through session. This must be a unique value.

Element 2: SNA Upline Facility Device Name

SNUF-device-name: Specify the name of the host device to use for the pass-through session. This must be a unique value.

Element 3: Default Host Program Name

default-host-program-name: Specify the name of the program to be started on the host if a program name was not specified by the retail controller.

Element 4: Text Description

***BLANK:** Text is not specified.

'entry-description': Specify a short description of 20 characters or less for each retail pass-through entry.

FTRCPNAME

Specifies the control point name of the adjacent control point that is being filtered by the local system when a directory search request is made.

Note: This parameter is valid only if TYPE(*APPNDIR) is specified.

***ANY:** Any control point name is filtered.

generic-filtered-CP-name:* Specify the generic control point name (part of a name followed by an asterisk) of the adjacent control point(s) being filtered. The generic control point name allows one directory entry to be defined for all control points, in a single network, with a name that matches the characters preceding an *.

filtered-CP-name: Specify the control point name of the adjacent control point being filtered.

FTRCPNETID

Specifies the control point network identifier of the adjacent control point being filtered by the local system when a directory search request is made.

Note: This parameter is valid only if TYPE(*APPNDIR) is specified.

***NETATR:** The LCLNETID value specified in the system network attributes is used.

filtered-CP-network-ID: Specify the control point network identifier of the adjacent control point being filtered by the local system.

LCLLOCNAME

Specifies the local location name being supplied by the caller that is being filtered by the local system. When the local system is initiating a session, this is the local location name being used. When a bind is received from another system, this is the Secondary Logical Unit (SLU) name being used.

Note: This parameter is valid only if TYPE(*APPNSSN) is specified.

***ANY:** Any local location name will be filtered by the local system.

generic-local-location-name:* Specify the generic local location name (part of a name followed by an asterisk) of the local location(s) being filtered. The generic local location name allows one entry to be defined for all local location names, on the system, with a name that matches the characters preceding an *.

local-location-name: Specify the local location name that is being filtered by the local system.

FTRACN

Specifies the filter action for APPN requests being handled by the local system.

Note:

This parameter is valid only if TYPE(*APPNDIR) or TYPE(*APPNSSN) is specified.

***ACCEPT:** The request is accepted.

***REJECT:** The request is rejected.

APPNDIRE

Specifies the APPN directory search entry being filtered by the local system. This parameter may be specified when *APPNDIR is specified for the TYPE parameter. Up to 300 entries may be specified at a time.

Element 1: Filtered Location Name

***ANY:** Any control point location will be filtered.

generic-filtered-CP-loc-name:* Specify the generic name (part of a name followed by an asterisk) of the control point location(s) to be filtered. The generic name allows one name to be specified for all control point locations with a name that matches the characters preceding an *.

filtered-CP-location-name: Specify the control point location name to be filtered. This is the name of the location that is owned by the adjacent control point being filtered if the adjacent CP is an end node or LEN node. Or, the name of some location that accesses the local network via the adjacent control point (a non-native network node). This location name represents the name of the session partner attempting to establish a session with the remote location name (the location that exists in the local system's network).

Element 2: Filtered CP Location Network ID

***NETATR:** The LCLNETID value specified in the system network attributes is used.

filtered-CP-location-network-ID: Specify the network identifier associated with the CP location name to be filtered.

Element 3: Partner Location Name

***ANY:** Any remote location will be filtered.

generic-partner-location-name:* Specify the generic name (part of a name followed by an asterisk) of the partner location(s) to be filtered. The generic name allows one name to be specified for all partner locations with a name that matches the characters preceding an *.

partner-location-name: Specify the name of the partner location to be filtered.

Element 4: Partner Network Identifier

***NETATR:** The LCLNETID value specified in the system network attributes is used.

partner-network-identifier: Specify the network identifier associated with the partner location to be filtered.

Element 5: Entry Description

***BLANK:** Text is not specified.

'entry-description': Specify a short description of 20 characters or less for each entry.

APPNSSNE

Specifies the APPN session endpoint entry being filtered by the local system. This parameter may be specified when *APPNSSN is specified for the TYPE parameter. Up to 300 entries may be specified at a time.

Element 1: Remote Location Name

***ANY:** Any remote location will be filtered.

generic-remote-location-name:* Specify the generic name (part of a name followed by an asterisk) of the remote location(s) to be filtered. The generic name allows one name to be specified for all remote locations with a name that matches the characters preceding an *.

Element 2: Remote Network Identifier

***NETATR:** The LCLNETID value specified in the system network attributes is used.

Remote-network-identifier: Specify the remote network identifier associated with the remote location to be filtered.

Element 3: Text Description

***BLANK:** Text is not specified.

'entry-description': Specify a short description of 20 characters or less for each entry.

GRPNAME

Specifies the SNA pass-through group name of the configuration list entry being added. The group name has upstream SNA pass-through device names associated with it (DEV parameter) and must exist in the configuration list.

Note: This parameter is valid only if TYPE(*SNAPASTHR) is specified.

DEV Specifies the names of the upstream devices associated with the SNA pass-through group (GRPNAME parameter).

Note: This parameter is valid only if TYPE(*SNAPASTHR) is specified.

TEXT Specifies the text that briefly describes the SNA pass-through group. More information is in Commonly used parameters.

Note: This parameter is valid only if *APPNDIR, *APPNSSN, or *SNAPASTHR is specified for the TYPE parameter.

***BLANK:** Text is not specified.

'entry-description': Specify a description of up to 50 characters for the SNA pass-through entry being added.

SNAPASTHRE

Specifies the SNA pass-through entry. This parameter can be specified if TYPE(*SNAPASTHR) is specified. However, because this parameter may be removed in a later release, whenever possible use GRPNAME, DEV, and TEXT parameters.

Example for ADDCFGLE

```
ADDCFGLE TYPE(*APPNLCL) APPNLCL((LOC1 'location one')
(LOC2 'location two'))
```

This command adds local locations LOC1 and LOC2 to configuration list QAPPNLCL.

Error messages for ADDCFGLE

*ESCAPE Messages

CPF260F

Configuration list &1 not found.

CPF261C

Index for configuration list &1 not changed.

CPF261D

Index for configuration list &1 not changed.

CPF2613

Too many entries were added.

CPF2625

Not able to allocate object &1.

CPF263A

CFGLE type &1 does not match existing type &2.

CPF2634

Not authorized to object &1.

CPF2663

Configuration list &1 previously deleted.

CPF9838

User profile storage limit exceeded.

ADDCNNLE (Add Connection List Entry) Command Description

ADDCNNLE Command syntax diagram

Purpose

The Add Connection List Entry (ADDCNNLE) command is used to add an entry to a connection list.

Required Parameters

CNNL Specifies the name of the connection list to which this entry is added.

ENTRY

Specifies the name of the entry to be added to the connection list. Each entry in the connection list must have a unique name.

RMTNBR

Specifies the number of the remote system in the Integrated Services Digital Network (ISDN).

***ANY:** Any value, including no value, specified in the received Called Party Number information element (IE) (encoded on the call by the system) is acceptable for incoming calls. For outgoing calls, the system requires the target number to be supplied so that a Called Party Number or Keypad Facility IE can be encoded on the outgoing call. Therefore, if *ANY is specified for an outgoing call, the call out attempt fails.

'remote-number': Specify the remote number (up to 32 characters enclosed in apostrophes). Extra characters, such as parentheses, can be used to aid in readability, if they are specified on the remove character (RMVCHR) parameter when the connection list was created using the Create Connection List (CRTCNL) command.

Optional Parameters

RMTNBRTYPE

Specifies the type of remote number.

***NETTYPE:** The system determines the local number type by using the value specified on the NETTYPE parameter.

***UNKNOWN:** The local number type is not known.

***INTERNATL:** The local number is an international number type.

***SUBSCRIPTION:** The local number is a subscription number type.

***NATIONAL:** The local number is a national address type.

***NETSPECIFIC:** The local number type is specific to the network.

***ABR:** The local number type is abbreviated.

RMTNBRPLAN

Specifies the numbering plan used for the remote number.

***NETTYPE:** The numbering plan is determined by the value specified on the NETTYPE parameter.

***UNKNOWN:** The numbering plan is not known.

***ISDN:** The ISDN E.164 numbering plan is used.

***DATA:** The data numbering plan is used.

***NATIONAL:** The national numbering plan is used.

***PRIVATE:** A private numbering plan is used.

RMTSUBADR

Specifies the subaddress of the remote system.

***ANY:** Any value, including none, specified in the received Called Party Subaddress IE is acceptable for incoming calls. For outgoing calls, no Called Party Subaddress IE is encoded.

remote-subaddress: Specify the subaddress of the remote system; up to 40 hex characters.

RMTSUBTYPE

Specifies the type of remote subaddress.

***NETTYPE:** A default value is used based on the network type specified on the NETTYPE parameter when the connection list was created using the CRTCNL command.

***NSAP:** The remote subaddress type is NSAP (X.213).

***USER:** Some of the common OS/400 objects and database physical files are included.

LCLNBR

Specifies information about the local number that is called for an incoming call. If the entry is used for an outgoing call this parameter is ignored.

***ANY:** Any value, including no value, specified in the received Called Party Number IE is acceptable.

***NWID:** Any value, including no value, specified in the received Called Party Number IE is acceptable. The number used for outgoing calls is determined by the network interface description.

'local-number': Specify the local number (up to 32 characters enclosed in apostrophes). For this entry, only calls directed at this local number are accepted. Extra characters, such as parentheses, can be used if they are specified on the remove character parameter (RMVCHR parameter).

LCLNBRTYPE

Specifies the type of local number specified on the LCLNBR parameter.

***NETTYPE**: The system determines the local number type by using the value specified on the NETTYPE parameter.

***UNKNOWN**: The local number type is not known.

***INTERNATL**: The local number is an international number type.

***SUBSCRIPTION**: The local number is a subscription number type.

***NATIONAL**: The local number is a national address type.

***NETSPECIFIC**: The local number type is specific to the network.

***ABR**: The local number type is abbreviated.

LCLNBRPLAN

Specifies the numbering plan used for the local number.

***NETTYPE**: The numbering plan is determined by the value specified on the NETTYPE parameter.

***UNKNOWN**: The numbering plan is not known.

***ISDN**: The ISDN E.164 numbering plan is used.

***DATA**: The data numbering plan is used.

***NATIONAL**: The national numbering plan is used.

***PRIVATE**: A private numbering plan is used.

LCLNBRPSN

Specifies the intention of the calling user for the presentation of the local number to the called user. This parameter applies only to outgoing calls.

***NONE**: The local number presentation is not encoded. The network determines whether to present the local number to the called user.

***ALLOW**: The local number is presented to the called user.

***RESTRICT**: The presentation of the local number to the called user is restricted by the network.

LCLSUBADR

Specifies the subaddress of the local system.

***ANY**: Any value, including no value, specified in the Received Calling Party Subaddress IE is acceptable for incoming calls. For outgoing calls, no Called Party Subaddress IE is encoded.

***NWID**: Any value, including no value, specified in the received Called Party Subaddress IE is acceptable for incoming calls. For outgoing calls, the Called Party Subaddress IE is encoded on the network interface description.

local-subaddress: Specify the local subaddress; up to 40 hex characters.

LCLSUBTYPE

Specifies the type of subaddress used by the local system.

***NETTYPE**: A default value is used based on the network type specified on the NETTYPE parameter when the connection list was created using the CRTCNL command.

***NSAP**: The subaddress type is NSAP (X.213).

***USER**: The subaddress type is user-specified.

INFTRFTYPE

Specifies the information transfer type. The information transfer type determines the layer-1 protocol.

***UNRESTRICTED:** The data-channel traffic appears as digital information; no physical transformation is required and each B-channel operates at capacity (64k bits per second (bps)).

***V110:** The transfer type is V-series Recommendation 110. Each B-channel operates at 56k bps.

***DOV** Allows Data Over Voice (DOV) digital data to be transferred over an ISDN voice call. Also, this is referred to as Data Over Voice Bearer Service (DOVBS), Data Over Speech Bearer Service (DOSBS), TollSaver, or TollMizer. This option should only be used if an ISDN voice call is less expensive than an ISDN data call or if a bearer service for data is not available. The remote location must also support this feature. Data is transferred at 56Kbps in each direction.

***FAXMODEM:** Allows Facsimile (FAX) data from the integrated fax modem to be transferred over an ISDN voice call. This option should be used to connect to a remote location that is using a fax device on an analog telephone line or to another ISDN device that has Group 3 FAX modem capabilities. Data is transferred at fax speeds up to 14.4Kbps.

***ASYNCMODEM:** Allows data from the integrated asynchronous modem to be transferred over an ISDN voice call. This option should be used to connect to a remote location that is using an asynchronous modem on an analog telephone line. Data is transferred at modem speeds up to 33.6Kbps from the remote analog device to this digital connection and up to 56Kbps from this digital connection to the remote analog device.

***SYNCMODEM:** Allows data from the integrated synchronous modem to be transferred over an ISDN voice call. This option should be used to connect to a remote location that is using a synchronous modem on an analog telephone line. Data is transferred at modem speeds up to 33.6Kbps from the remote analog device to this digital connection and up to 56Kbps from this digital connection to the remote analog device.

MDMINZCMD

Specifies the command string to send to set the modem.

***NONE:** No command string is sent to the modem.

***LIND:** The command string from the line description is used.

command-string: Specifies up to 60 characters that represent the command string sent to the modem. Valid characters are upper case A thru Z, lower case a thru z, numbers 0 thru 9, and special characters:

Table 1. Special characters

Character	Description
.	Period
<	Less than sign
(Left parenthesis
+	Plus sign
&	Ampersand
*	Asterisk
)	Right parenthesis
;	Semicolon
-	Minus sign
/	Slash
,	Comma
_	Underline
>	Greater than sign
?	Question mark
:	Colon

Character	Description
=	Equal sign
	Spaces
#	Number sign
"	Double quote
!	Exclamation mark
@	At sign
^	Circumflex
%	Percent
[Left square bracket
]	Right square bracket
\	Back slash
\$	Dollar sign

TEXT Specifies the text that briefly describes the entry in the connection list. More information is in Commonly used parameters.

***BLANK:** Text is not specified.

'description': Specify a description of the entry. Specify up to 50 characters of text, enclosed in apostrophes.

Example for ADDCNNLE

```
ADDCNNLE CNNL(CHICAGO) ENTRY(CORPORATE)
RMTNBR(' (896) 999-5555')
```

This command adds entry CORPORATE in connection list CHICAGO. The entry will have (896) 999-5555 as a remote number.

Error messages for ADDCNNLE

*ESCAPE Messages

CPF266C

Connection list &1 not found.

CPF266F

Entry &2 not added to connection list &1.

ADDDTADFN (Add Data Definition) Command Description

ADDDTADFN Command syntax diagram

Purpose

The Add Data Definition (ADDDTADFN) command allows the user to copy file, record format, and field definitions from an externally described database file to a data dictionary.

When definitions are added to a dictionary, the system does a search to find out if the dictionary contains a definition with the same name as the one being added. If an exact match of the definition is found, the existing definition is used. If an exact match is not found, a new version of the definition is created.

Database files using the following functions are not added to a dictionary:

- Access path sharing
- Alternate collating sequence
- Program-described file

- Join logical file
- Logical file
 - with select/omit specifications
 - based on more than one file

Only format and field definitions of database files using the following functions are added to a dictionary:

- Field default values
- Field validity check codes
- Key fields defined using names based on physical files
- Derived fields

Note: When adding a file that is already linked, the current link is ended and then the definition is added and linked.

Required Parameters

FILE Specifies the qualified name of the externally described database file from which the data definition is copied to the specified dictionary. A database file name must be specified.

The name of the file can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

database-file-name: Specify the name of the database file from which the definition is copied to the specified dictionary.

DTADCT

Specifies the name of the dictionary to which the definitions are added.

Optional Parameter

DFN Specifies the name given to the file definition when it is copied into the dictionary.

***FILE:** The name of the file definition is the same as the database file specified in the FILE parameter.

file-definition-name: Specify the name given to the file definition when copied to the data dictionary.

Example for ADDDTADFN

```
ADDDTADFN FILE(MYLIB/MYFILE)
DTADCT(MINE) DFN(*FILE)
```

This command copies the definitions from MYFILE located in library MYLIB to the dictionary MINE. The file definition has the same name as the database file specified in the FILE parameter.

Error messages for ADDDTADFN

*ESCAPE Messages

CPF2E9E

Not enough space to add to dictionary &1.

CPF2FE0

Error occurred while opening dictionary &1.

CPF2FE1

Error occurred while closing dictionary &1.

CPF2FE2

Dictionary &1 currently in use.

CPF2F02

Not authorized to use dictionary &1.

CPF2F07

Dictionary &1 in error.

CPF2F08

Dictionary &1 not found.

CPF2F61

File &2 in &3 currently in use.

CPF2F72

File &2 in &3 not valid for ADDDTADFN.

CPF2F73

ADDDTADFN not allowed for SQL database files.

CPF2F74

Attributes of file &2 in &3 not supported.

CPF9812

File &1 in library &2 not found.

CPF9820

Not authorized to use library &1.

CPF9822

Not authorized to file &1 in library &2.



ADDDEVDMNE (Add Device Domain Entry) Command Description

ADDDEVDMNE Command syntax diagram

Purpose

The Add Device Domain Entry (ADDDEVDMNE) command is used to add a cluster node to the membership list of a device domain. There is no CL command to create a device domain. The device domain will be created when the first cluster node is added to it. Once the node is a member of a device domain, it can be added to the recovery domain of device cluster resource groups.

This CL command can be called from a program running on any node in the cluster which has a status of Active.

This CL command requires that OS/400 option 41, HA Switchable Resources, is installed and a valid license key exists on all cluster nodes that will be in the device domain.

Restrictions

1. To use this command you must have *IOSYSCFG authority.
2. This CL command cannot be called from a cluster resource group exit program.
3. The node to be added and at least one current member of the device domain must be Active. On certain conditions, all current members of the device domain must be Active.
4. A node can only be a member of one device domain.
5. This CL command will fail if any member of the device domain to which the node is being added has a status of Partition.
6. The CL command will fail if any node in the cluster has a status of Partition and this is the first node being added to the device domain.

Required Parameters

CLUSTER

Specifies the name of the cluster that contains the node.

cluster-name: Specify the name of the cluster which contains the node.

DEVDMN

Specifies the device domain to which the node is being added. If the device domain does not currently exist, it will be created.

device-domain-name: Specify the name of the device domain to which the node is being added.

NODE This parameter identifies a cluster node to be added to the device domain.

node-identifier: Specify the name of the cluster node.

Example for ADDDEVDMNE

```
ADDDEVDMNE CLUSTER(MYCLUSTER) DEVDMN(MYDOMAIN) NODE(NODE01)
```

Consider a cluster with a cluster membership list of NODE01, NODE02, and NODE03 and no existing device domains. This command creates the device domain MYDOMAIN and adds node NODE01 to the device domain membership list. Nodes NODE02 and NODE03 do not belong to any device domain.

Error messages for ADDDEVDMNE

*ESCAPE Messages

CPF0001

Error found on &1 command.



ADDDIRE (Add Directory Entry) Command Description

ADDDIRE Command syntax diagram

Purpose

The Add Directory Entry (ADDDIRE) command is used to add new entries to the system distribution directory. The directory contains the user's user identifier (ID), profile name, system name, system address, mailing address, telephone numbers, and other user information. The ADDDIRE command provides a parameter for each of the fields contained in the directory.

The ADDDIRE command does not provide interactive display support. This is provided by the Work with Directory Entries (WRKDIRE) command.

An X.400 originator/recipient (O/R) name can be added to the directory with this command. X.400 is an international standard for communications and the O/R name is the addressing information used in X.400 communications. The X.400 O/R name must be in character set 1169 and code page 500. This set includes A through Z, 0 through 9, and some special characters. Additional information on characters allowed is in the Globalization topic in the Information Center.

Notes:

1. To prevent the system from changing lowercase characters to uppercase, enclose the values in apostrophes. This does not apply to user ID/address, system name/group, department, or X.400 O/R name.
2. Only the user ID/address, system name/group, department, and X.400 O/R name are translated from the graphic character identifier (GCID) specified by the CMDCHRID parameter. All other parameters are stored exactly as they are entered and the GCID is stored with them. The default GCID value is taken from the QCHRID system value. The user can override the defaults by specifying a character set and code page or by specifying *DEV D for the display device description.
3. Double-byte character set (DBCS) characters can be entered for the following system directory entry parameters:

USR D	LOCATION
LSTNAM	BLDG
FSTNAM	OFC
MIDNAM	ADDR1
PREFNAM	ADDR2
FULNAM	ADDR3
DEPT	ADDR4
TITLE	TEXT
CMPNY	USRDFNFLD

Restriction: The user of this command must have security administrator (*SECADM) authority.

Required Parameters

USRID

Specifies the user ID and address of the user for whom the directory entry is made. Both elements must be specified. If lowercase characters are specified, the system translates and stores them as uppercase characters. Further information about specifying the user ID and address is in the SNA

Distribution Services  book.

Element 1: User ID

***ANY:** Any user ID at the address specified on Element 2 of this parameter is used. Only one *ANY is allowed for each address. This value is used to resolve a distribution that does not match a specific user ID but matches an address.

user-ID: Specify the user ID for this directory entry. Up to 8 characters can be specified. If this value is specified, an address must be specified for Element 2.

Element 2: User Address

***ANY:** Any address for the user ID specified on Element 1 is used. One **USRID(*ANY *ANY)** entry is allowed in the directory. This value is used to resolve distributions that do not match any other directory entries.

user-address: Specify the address for this directory entry. Up to 8 characters can be specified.

USR D Specifies the description associated with the user ID and address. For example, the description can contain the user's full name, department number, or position. Specify up to 50 characters for the description.

USER Specifies the user profile of the user being added to the directory. If the user being added is a local user, a valid profile must exist on the local system.

A user profile is required to define a local user whose mail is sent to the remote system specified in the system name/group.

***NONE:** The user being added to the directory is a remote user and has no local profile. If ***NONE** is specified and the SYSNAME parameter indicates the local system, an error message is returned.

user-profile-name: Specify a valid system user profile name up to 10 characters in length. The user profile name is required for all local users. If a profile name is specified for a user whose mail is sent to a remote system, the profile name must be valid on the local system.

Optional Parameters

SYSNAME

Specifies the one- or two-part name of the system on which the user works. If a two-part system name is specified on the command line, up to 8 characters make up both the system name and the system group name. The parts should be separated by at least one space.

A remote user can be added to the directory before the system name and system group are defined in the network tables, but distributions cannot be sent to that user until the remote system name and system group are defined. The remote system name and system group name are defined by using the Configure Distribution Services (CFGDSTSRV) command. Additional information on defining a remote system name and group name is in the SNA Distribution Services



book.

***LCL:** The system name is its local name. All local users being added to the directory should have ***LCL** specified as the system name.

***PC:** This system name is for distributed systems node executive (DSNX) users with a personal computer (PC) attached to the system.

***ERROR:** This value is used when the user's network contains a central system that receives all unresolved distributions. In this type of network, distribution looping may be encountered when a distribution cannot find a specific user ID on the intended system and the intended system has an ***ANY *ANY** entry directing distributions to the central system. The central system also has a default ***ANY address** entry directing unresolved distributions to the intended system. To prevent distribution looping, specify ***ERROR** as the system name for the default entry being added. When a distribution cannot find a specific user ID, but matches this default entry, the distribution is handled as a user that is not valid, just as if no directory match were found.

***ERROR** is valid only when **USRID(*ANY address)** or **USRID(*ANY *ANY)** is specified.

Element 1: System Name

system-name: Specify the name of the system on which the user works.

Element 2: System Group Name

group: Specify the system group name of the system on which the user works.

LSTNAM

Specifies the user's last name. If no names are provided (last, first, middle, preferred, or full), but a value is specified on the DEPT parameter, the last name will default to an asterisk (*). This is because the directory department function requires a non-blank full name when a department value is specified.

***NONE:** No last name is specified.

last-name: Specify up to 40 characters for the user's last name.

FSTNAM

Specifies the user's first name.

***NONE:** No first name is specified.

first-name: Specify up to 20 characters for the user's first name.

MIDNAM

Specifies the user's middle name.

***NONE**: No middle name is specified.

middle-name: Specify up to 20 characters for the user's middle name.

PREFNAM

Specifies the user's preferred name. For example, "Jonathan" likes to be called "Jon".

***NONE**: No preferred name is specified.

preferred-name: Specify up to 20 characters for the user's preferred name.

FULNAM

Specifies the user's full name. Directory entries are shown in the full name format when using the search and department functions. It is recommended that the user institute a consistent naming convention for the full name. Note that uppercase and lowercase alphabetic characters have different sorting sequences. Making the first character of each name uppercase and the rest that follow lowercase is the preferred format.

If FULNAM(*DFT) is specified, the following format is used to create the full name:

LAST NAME, FIRST NAME MIDDLE NAME (PREFERRED NAME)

The preferred name is always enclosed in parentheses. If no values are specified for the last, first, and middle names, but the DEPT parameter contains a value, the last name defaults to an asterisk (*). If the user specifies FULNAM(*DFT), the full name defaults to an asterisk because it is built from the last name.

***DFT**: The full name is created from the user-defined values specified on the LSTNAM, FSTNAM, MIDNAM, and PREFNAM parameters.

full-name: Specify up to 50 characters for the user's full name.

DEPT Specifies the name of the department of which the user is a member.

***NONE**: The user is not defined as a member of a department.

department-name: Specify up to 10 characters for the name of the user's department.

TITLE Specifies the user's job title.

***NONE**: No job title is added.

job-title: Specify up to 40 characters for the user's job title.

CMPNY

Specifies the name of the user's company.

***NONE**: No company name is added.

company-name: Specify up to 50 characters for the name of the user's company.

TELNBR1

Specifies the primary telephone number of the user. The telephone number can be specified in any arrangement appropriate for the user, including an international telephone number.

***NONE**: No primary telephone number is specified.

telephone-number-1: Specify up to 26 characters for the primary telephone number of the user.

TELNBR2

Specifies a second telephone number for the user. The telephone number can be specified in any arrangement appropriate for the user, including an international telephone number.

***NONE:** No second telephone number is specified.

telephone-number-2: Specify up to 26 characters for the second telephone number of the user.

FAXTELNR

Specifies a facsimile telephone number for the user. The facsimile telephone number can be specified in any arrangement appropriate for the user, including an international telephone number.

***NONE:** No facsimile telephone number is specified.

facsimile-telephone-number: Specify up to 32 characters for the user facsimile telephone number.

LOC Specifies the location of the user. For example, the location can specify a building and floor, a department, or a remote site.

***NONE:** No location is specified.

location: Specify up to 40 characters for the location of the user.

BLDG Specifies the name of the building in which the user works.

***NONE:** No building name is added.

building-name: Specify up to 20 characters for the name of the building in which the user works.

OFC Specifies the name of the office in which the user works.

***NONE:** No office name is added.

office-name: Specify up to 16 characters for the name of the office in which the user works.

ADDR1-ADDR4

These four parameters specify the mailing address of the user. Up to 40 characters of data can be specified in each of these fields.

***NONE:** No address lines are specified.

address-line-1-4: Specify the user's mailing address in any arrangement, up to 40 characters per line.

INDUSR

Specifies whether the user being added to the directory is an indirect user. An indirect user is a local user who does not sign on the system to receive mail but receives printed mail. An indirect user is a local user and must have a profile on the local system.

***NO:** The user is not an indirect user.

***YES:** The user is an indirect user.

PRTPEERS

Specifies whether private mail for an indirect user is printed. Consideration should be given to restricting public access to the printer when private mail is printed.

***NO:** No private mail is printed for this indirect user.

***YES:** Private mail is printed for this indirect user.

PRTCOVER

Specifies whether a cover page is printed when the user's mail is printed.

***YES:** The cover page is printed.

***NO:** The cover page is not printed.

NFYMAIL

Specifies whether the user is notified of the arrival of mail.

***SPECIFIC:** The user is notified of the arrival of specific types of mail. The types of mail are specified on the NFYPTYPERS parameter and the NFYMSGSGS parameter.

***ALLMAIL:** The user is notified of the arrival of all types of mail.

***NOMAIL:** The user is not notified of the arrival of mail.

NFYPTYPERS

Specifies whether the user is notified of the arrival of priority, private, and important mail.

Note:

This parameter is valid only if NFYMAIL(*SPECIFIC) is specified.

***YES:** The user is notified of the arrival of priority, private, and important mail.

***NO:** The user is not notified of the arrival of priority, private, and important mail.

NFYMSGS

Specifies whether the user is notified of the arrival of messages.

***YES:** The user is notified of the arrival of messages.

***NO:** The user is not notified of the arrival of messages.

NETUSRID

Specifies the unique network user ID for this directory entry. This ID is used during directory shadowing to uniquely identify a user in a network.

***USRID:** Set the network user ID to the user ID and address associated with this directory entry.

network-user-ID: Specify the network user ID for the user. A maximum of 47 characters can be specified.

TEXT Specifies any additional information to describe the directory entry. More information is in Commonly used parameters.

***NONE:** No text is specified.

'description': Specify up to 50 characters of text to describe additional information about the user.

CMDCHRID

Specifies the character identifier (graphic character set and code page) for data being specified as parameter values on this command. This character identifier (CHRID) is related to the display device used to specify the command. More information about CHRID processing is in the

Application Display Programming  book.

***SYSVAL:** The system determines the graphic character set and code page values for the command parameters from the QCHRID system values.

***DEVD:** The system determines the graphic character set and code page values for the command parameters from the display device description where this command is entered. This option is supported only when the command is entered from an interactive job. If this option is specified in a batch job, an error message is returned.

Element 1: Character Set

graphic-character-set: Specify the character set used to create the command parameters. Valid values range from 1 through 9999 characters.

Element 2: Code Page

code-page: Specify the code page. Valid values range from 1 through 9999.

COUNTRY

Specifies the country or region code part of the X.400 O/R name.

***NONE:** No country or region code is specified.

country-code: Specify an ISO 3166 Alpha-2 code or a CCITT country or region code. More information on this parameter is in Commonly used parameters.

ADMD Specifies the administrative management domain part of the X.400 O/R name.

***NONE:** No administrative management domain is specified.

administrative-management-domain: Specify a 1- to 16-character administrative management domain name. An administrative management domain is a public organization that handles a management domain. A management domain is a set of message transfer agents and user agents that comprise a system capable of handling messages.

PRMD Specifies the private management domain part of the X.400 O/R name.

***NONE:** No private management domain is specified.

private-management-domain: Specify a 1- to 16-character description of the private management domain. A private management domain is a private company or a non-commercial organization that handles a management domain. A management domain is a set of message transfer agents and user agents that comprise a system capable of handling messages.

SURNAM

Specifies the X.400 last name part of the personal name within the X.400 O/R name.

***NONE:** No surname is specified.

***LSTNAM:** The last name of the user specified in the directory entry is used as the surname.

surname: Specify up to 40 characters for the surname.

GIVENNAM

Specifies the X.400 user first name part of the personal name within the X.400 O/R name.

***NONE:** No given name is specified.

***FSTNAM:** The user first name specified in the directory entry is used as the given name. It is truncated to 16 characters.

given-name: Specify up to 16 characters for the given name.

INITIALS

Specifies the initials part of the personal name within the X.400 O/R name. For example, the initials for John Henry Smith are JH.

***NONE:** No initials are specified.

initials: Specify up to 5 characters for the initials.

GENQUAL

Specifies the generation qualifier part of the personal name within the X.400 O/R name.

***NONE:** No generation qualifier is specified.

generation-qualifier: Specify up to 3 characters for the generation qualifier.

ORG Specifies the organization part of the X.400 O/R name.

***NONE:** No organization name is specified.

'organization': Specify an organization name of up to 64 characters.

ORGUNIT

Specifies the organization-defined unit part of the X.400 O/R name.

***NONE:** No organizational unit is specified.

'organizational-unit': Specify up to 32 characters for the name of an organizational unit. Up to 4 organizational units can be listed in order of descending significance.

DMNDFNATR

Specifies the type and value of a domain-defined attribute not specified by X.400 standards but allowed in the X.400 O/R name to accommodate existing message handling systems. Up to 4 sets of attributes can be specified.

Element 1: Domain-Defined Attribute Type

***NONE:** No domain-defined attribute type is specified.

'attribute-type': Specify up to 8 characters for the description of the domain-defined attribute type.

Element 2: Domain-Defined Attribute Value

***NONE:** No domain-defined attribute value is specified.

'value': Specify up to 128 characters for the description of the domain-defined attribute value.

MSFSRVLVL

Specifies the mail server framework service level for a local user. This value is ignored for a remote user. It indicates where mail is stored on the system.

***USRIDX:** The mail is stored in a user index. OfficeVision* mail users should specify this option.

***SYSMS:** The mail is stored on the system supported message store. The system message store can be accessed by Ultimedia Mail/400, or by industry standard mail APIs that are used on the client. Ultimedia Mail/400 users should specify this option.

***DOMINO:** The mail is stored in the Lotus Domino mail database.

Element 1: Mail service level field-name

field-name: Specify the field name of another mail service for this user, if one is used. Specify up to 10 characters for the field name. This value should contain a user-defined field in the system directory that has been defined by the Change System Directory Attributes (CHGSYSDIRA) command in the USRDFNFLD parameter with a field type of *MSFSRVLVL. The user-defined field specified here should then contain information needed by the mail server framework user exit program when the program is determining where to store the mail. The address resolution exit point name is QIBM_QZMFMSF_ADR_RSL. See the AnyMail/400 Mail Server Framework Support



book for more information. This field could just be used as an indicator and the value does not have to be a user-defined field. Whenever possible, the value specified here should be a user-defined field.

Element 2: Mail service level product-ID

***NONE:** No user-defined field product ID is specified.

Product-ID: Specify up to 7 characters for the user-defined field product ID.

PREFADR

Specifies the preferred address for a user. This tells the mail server framework what fields to use in the system distribution directory for the preferred address of a user. Specify *USERID for OfficeVision and SNADS. SNADS handles all the mail that goes to a user index including the gateway for X.400 O/R names and Simple Mail Transfer Protocol (SMTP) names.

***USRID:** The user ID/address is the preferred address for this user.

***ORNAME:** The X.400 O/R name is the preferred address for this user.

***SMTP:** The SMTP name is the preferred address for this user.

Element 1: Preferred address field-name

field-name: Specify up to 10 characters for the field name. This value should contain an IBM-defined or a user-defined field in the system directory that has been defined by the Change System Directory Attributes (CHGSYSDIRA) command in the USRDFNFLD parameter with a field type of *ADDRESS. The field specified here should then contain information needed by the mail server framework user exit program when the program is determining what address to use when sending the mail. This field could just be used as an indicator and the value does not have to be an IBM-defined or user-defined field. Whenever possible, the value specified here should be an IBM-defined or a user-defined field.

Element 2: Preferred address product-ID

***NONE**: No user-defined field product ID is specified.

***IBM**: The field name is an IBM-defined field in the system distribution directory. Allowed IBM-defined field names are:

- USER (user profile)
- CCMailADR (cc*Mail** address)
- FULNAM (full name)
- NETUSRID (network user ID)
- TELNBR1 (telephone number 1)
- TELNBR2 (telephone number 2)
- FAXTELNBR (facsimile telephone number)

Product-ID: Specify up to 7 characters for the user-defined field product ID.

Element 3: Preferred address address-type

address-type: Specify up to 8 characters for the address type. The address type is a mail server framework type name that is specified on the Add Mail Framework Type Configuration (QzmfAddMailCfg) API. Whenever possible, this value should be one of the mail server framework configuration type names. When an address type is specified for a preferred address that is a special value, specify *N for the product ID.

CCMAILADR

Specifies the cc*Mail** address for this user.

***NONE**: No cc*Mail** address is specified.

*'cc*Mail-address'*: Specify the cc*Mail** address. The address field has a maximum of 126 characters. If the address includes both a remote post office name and an alias name, the limit is 126 characters for each, with a space separating them (total 253 characters). If the remote post office name contains spaces, the name must be enclosed in quotation marks. This adds two characters to the limit for a total of 128 characters or 255 characters with the alias name.

CCMAILCMT

Specifies the cc*Mail** comment for this user.

***NONE**: No cc*Mail** comment is specified.

*'cc*Mail-comment'*: Specify up to 126 characters for the cc*Mail** comment field.

USRDFNFLD

Specifies the user-defined field names and values. A list of these user-defined field names can be displayed using CHGSYSDIRA and prompting with the F4 key. Up to 100 user-defined fields can be specified.

Note:

The following SMTP user-defined fields are not always displayed when the CHGSYSDIRA command is prompted, but they can still be used in the user-defined field (USRDFNFLD) parameter to add SMTP information to the system distribution directory.

SMTPAUSRID SMTP

SMTPDMN SMTP

SMPTRTE SMTP

***NONE:** No user-defined fields are specified.

Element 1: User-Defined Field Name

field-name: Specify up to 10 characters for the user-defined field name.

Element 2: User-Defined Field Product ID

***NONE:** No user-defined field product ID is specified.

product-ID: Specify up to 7 characters for the user-defined field product ID.

Element 3: User-Defined Field Value

'value': Specify up to 512 characters for the value of the user-defined field value. Blanks are padded on the right.

ALWSYNC

Specifies whether synchronization of this entry with other directories should be allowed.

***YES:** Synchronization is allowed.

***NO:** Synchronization is not allowed.

DLOWN

Specifies if the user profile or the group profile will be assigned the ownership of the document library objects (DLOs) for this directory entry.

***USRPRF:** The user profile associated with this directory entry is the owner of the newly created Document Library Objects (DLOs).

***GRPPRF:** The group profile specified in the user profile associated with this directory entry is made the owner of newly created DLOs and has all authority to the DLOs. If the group profile value is *NONE in the user profile, then the owner of the DLO is the user profile.

Examples for ADDDIRE

Example 1: Adding a Local User

```
ADDDIRE  USRID(HURST PAYROLL)
         USRD('Manager of Payroll')
         USER(ABHURST)
         LSTNAM(Hurst)
         FSTNAM(Arthur)
```

```

PREFNAM(Art)
DEPT(55K)
ADDR1('Dept55K/025-3')
ADDR2('IBM Rochester')
ADDR3('Highway 52 North')
ADDR4('Rochester, MN 55904')
LOC('Main Office')
BLDG(025-3)
OFC(E219)
TELNBR1('435-422-2120')
TELNBR2('435-422-1012')
FAXTELNBR('435-422-3296')
DLOOWN(*GRPPRF)

```

This command adds a local user to the distribution directory by allowing the system name parameter to default to *LCL. Since this is a local user, the user profile is specified. Address lines, location, and telephone numbers have been specified. Since the TEXT parameter is not used, it defaults to *NONE. This user is not an indirect user since the INDUSR parameter defaulted to *NO.

The user's last, first, and preferred names are specified. The full name was not specified, so FULNAM(*DFT) is used and will be created as, 'Hurst, Arthur (Art)'. This user has been added as a member of the department named 55K. If this department is searched, then 'Hurst, Arthur (Art)' will be included on the search list.

Any newly created DLOs associated with this directory entry, HURST PAYROLL, will be owned by the group profile specified in the Group Profile field in user profile ABHURST. The user entry is added to the directory if each one of the following is true:

1. A user ID and address HURST PAYROLL is not already in the directory.
2. The user profile name ABHURST is not already in the directory.

Example 2: Adding a Remote User

```

ADDIRE  USERID(BYRD NEWYORK)
        USRD('Arthur J. Byrd')
        USER(*NONE) SYSNAME(BOCA)
        LOC('Boca Raton, Florida')
        DEPT(61Q)

```

This command adds a remote user entry to the distribution directory. Since this is a remote user, the USER(*NONE) parameter is specified. The system name without the system group is specified. Except for the location, all of the parameters use default values. If the user-ID and address are unique, the user entry is added to the directory as a remote user.

If you are using directory shadowing, you do not need to add remote users as these users can be shadowed to your system.

If a department value is specified for this user, but no user name is specified, the last name is set to '*'. The full name is also '*' because it is created from the last name. This is done because the directory requires a non-blank name with department.

Additional Considerations

Multiple descriptions can be associated with a given user ID and address. For example, HURST DEPT48K can have an entry with the description Arthur B. Hurst and an entry with the description Manager of Dept. 48K. The ADDIRE command does not support adding another description to a user-ID. The Work with Directory Entries (WRKDIRE) is used for adding multiple descriptions for a user.

For local users, there is a one-to-one correspondence between the user ID and address and the user profile. Only one user ID and address can be associated with a user profile name, and only one user

profile name can be associated with a user ID and address. If a user profile name is specified on the ADDDIRE command that is already associated with an existing user ID and address in the directory, an error message is returned.

This should not cause a problem for remote users since the user profile name is not specified. However, if the profile is specified, it is verified to determine that the profile name is not already in the directory. If the profile is in the directory, an error message is returned.

Error messages for ADDDIRE

***ESCAPE Messages**

CPF8360

Not enough storage for commitment control operation.

CPF89A3

Operation not successful due to authority reasons.

CPF89A4

Operation not successful due to data validation reasons.

CPF8AA1

Library QUSRSYS not completely installed.

CPF90A8

*SECADM special authority required to do requested operation.

CPF9009

System requires file &1 in &2 be journaled.

CPF9024

System cannot get correct record to finish operation.

CPF905C

Error occurred trying to find a translation table.

CPF9082

User ID and address &1 &2 not added to directory.

CPF9096

Cannot use CMDCHRID(*DEV D), DOCCHRID(*DEV D) in batch job.

CPF9810

Library &1 not found.

CPF9838

User profile storage limit exceeded.

CPF9845

Error occurred while opening file &1.

CPF9846

Error while processing file &1 in library &2.

ADDDIRSHD (Add Directory Shadow System) Command Description

ADDDIRSHD Command syntax diagram

Purpose

The Add Directory Shadow System (ADDDIRSHD) command adds a supplier system to supply system distribution directory data to your system through directory shadowing.

Restriction: To use this command, you must have security administrator (*SECADM) authority.

Required Parameters

SYSNAME

Specifies a maximum of 8 characters for the name of the supplier system you are adding. You can specify uppercase letters A through Z, numbers 0 through 9, and special characters @, #, \$, and embedded blanks. Embedded blanks must be enclosed in single quotation marks ('). Leading blanks are not allowed. The @, #, and \$ characters are not recommended because they are not part of an invariant character set and are not available on all keyboards.

SCD Specifies the date and time at which the system you are adding begins supplying data to your system.

***CURRENT:** The system begins supplying data at the current date and time.

Element 1: Shadow Date

scheduled-shadow-date: Specify the date on which the system begins supplying data to your system. The date must be specified in the job date format.

Element 2: Shadow Time

scheduled-shadow-time: Specify the time at which the system begins supplying data to your system.

The time is specified in 24-hour format with or without a time separator as follows:

- With a time separator, specify a string of 5 or 8 digits where the time separator separates the hours, minutes, and seconds. If this command is entered from the command line, the string must be enclosed in apostrophes. If a time separator other than the separator specified for your job is used, this command fails.
- Without a time separator, specify a string of 4 or 6 digits (hhmm or hhmmss) where **hh** = hours, **mm** = minutes, and **ss** = seconds. Valid values for **hh** range from 00 through 23. Valid values for **mm** and **ss** range from 00 through 59.

Optional Parameters

FRQ Specifies the frequency with which the supplier system you are adding shadows data to your system, based on the value specified on the SCD parameter.

***WEEKLY:** Shadowing occurs once a week.

***DAILY:** Shadowing occurs once a day.

***BIWEEKLY:** Shadowing occurs every other week.

***MONTHLY:** Shadowing occurs on the same date every month.

***MONTHLYREL:** Shadowing occurs on the same relative day of the same relative week of every month, such as the first Monday of the month.

***HOURS:** Shadowing occurs in the interval specified on the HOURS parameter.

HOURS

Specifies the number of hours between shadows from the supplier system you are adding.

SKIPDAY

Specifies, when FRQ(*DAILY) is specified, the days of the week when shadowing does not occur. A maximum of five values, other than *NONE, can be specified.

***NONE:** No days are skipped.

***SUN:** Sundays are skipped.

***MON:** Mondays are skipped.

- ***TUE**: Tuesdays are skipped.
- ***WED**: Wednesdays are skipped.
- ***THU**: Thursdays are skipped.
- ***FRI**: Fridays are skipped.
- ***SAT**: Saturdays are skipped.

MONTHWK

Specifies whether shadowing that occurs on the same relative day of the month is scheduled to occur in the fourth week or the last week of the month.

4: Shadowing occurs on the same relative day in the fourth week of the month.

***LAST**: Shadowing occurs on the same relative day in the last week of the month, whether or not the month has four or five weeks.

INZ

Specifies the method used for the first shadow from the supplier system. The first shadow duplicates all of the local and shadowed data in the supplier system's distribution directory. Remote users are optionally supplied when the supplier specifies RMTSHD(*YES) on the Change System Directory Attributes (CHGSYSDIRA) command. Subsequent shadows include only data that has changed since the previous shadow.

Element 1: Automatic Shadow

***APPC**: The first shadow occurs when this command is run using advanced program-to-program (APPC) communications. If you are adding a supplier system with a large directory, you may want to specify *NONAPPC to prevent the first shadow from tying up your communications lines.

When *APPC is specified, Element 2 allows you to specify whether the data in the fields of a directory entry on your system is replaced by shadowed data if the same entry also exists in the supplier system's directory.

Element 2: Replace Data

***NO**: The data in the fields of existing directory entries on your system is not replaced with data from the supplier system.

***YES**: All shadowed data is added to your system distribution directory. The data in the fields of existing directory entries on your system is replaced with shadowed data if the same entry also exists in the supplier system's directory.

***NONAPPC**: The Copy to Directory (CPYTODIR) command is used for the first shadow. It is recommended that you run the CPYTODIR command before running this command. If shadowing from the supplier system you are adding starts before CPYTODIR is run, you may lose data.

***COMPLETED**: The initial shadow has already been done using the CPYTODIR command.

RMTLOCNAME

Specifies the remote location name of the supplier system you are adding.

***SYSNAME**: The value specified on the SYSNAME parameter is used for the remote location name.

remote-location-name: Specify the full name of a remote location.

A maximum of 8 characters can be specified. The first character must be an uppercase letter A through Z, or special character \$, #, or @. The name cannot contain a blank, plus sign (+), period (.), or an underscore (_). For more information, see the APPC, APPN, and HPR topic in the Information Center.

MODE Specifies the name of the mode that defines the sessions on the device used when shadowing data from the supplier system.

***NETATR**: The mode name specified in the network attributes is used.

mode-name: Specify the mode name.

A maximum of 8 characters can be specified. The first character must be an uppercase letter A through Z, or special character \$, #, or @. The name cannot contain a blank, plus sign (+), period (.), or an underscore (_). For more information, see the APPC, APPN, and HPR topic in the Information Center.

RMTNETID

Specifies the supplier system's remote network identifier (ID).

***LOC:** The remote network identifier (ID) associated with the remote location is used. If several remote network IDs are associated with the remote location, the system determines which remote network ID is used.

***NETATR:** The RMTNETID value specified in the system network attributes is used.

***NONE:** No remote network identifier (ID) is used.

remote-network-ID: Specify a maximum of 8 characters for the remote network ID.

The first character must be an uppercase letter A through Z, or special character \$, #, or @. The name cannot contain a blank, plus sign (+), period (.), or an underscore (_). For more information, see the APPC, APPN, and HPR topic in the Information Center.

LCLLOCNAME

Specifies the local location name. The local location name is used to identify your system to the supplier system you are adding.

***LOC:** The local location name associated with the remote location is used.

***NETATR:** The LCLLOCNAME value specified in the system network attributes is used.

local-location-name: Specify a maximum of 8 characters for the local location name.

The first character must be an uppercase letter A through Z, or special character \$, #, or @. The name cannot contain a blank, plus sign (+), period (.), or an underscore (_). For more information, see the APPC, APPN, and HPR topic in the Information Center.

TEXT Specifies the text that briefly describes the shadow system distribution directory. More information is in Commonly used parameters.

***SYSNAME:** The name specified on the SYSNAME parameter is used for the description.

'description': Specify no more than 50 characters of text, enclosed in apostrophes.

Examples for ADDDIRSHD

Example 1: Shadowing a System Weekly

```
ADDDIRSHD  SYSNAME(NYCITY)
           SCD( '92/05/01' '17:00:00' )
           FRQ(*WEEKLY) TEXT('Shadow New York System')
```

This command adds the supplier system NYCITY, which starts shadowing directory data to the local system on May 1, 1992 at 5:00 p.m. The shadow frequency is once a week. The description of the shadow system is 'Shadow New York System'.

Example 2: Shadowing a Remote System Hourly

```
ADDDIRSHD  SYSNAME(CHICAGO)
           SCD( '92/04/01' '20:00:00' )
           FRQ(*HOURS)  HOURS(12)
           RMTLOCNAME(CHIC01)
           LCLLOCNAME(CHICAG01)
           TEXT('Shadow Chicago System')
```

This command adds the supplier system CHICAGO, which starts shadowing directory data to the local system on April 1, 1992 at 8:00 p.m. The frequency of shadows is every 12 hours. The description of the shadow system is 'Shadow Chicago System'. The remote location name of the CHICAGO system is CHIC01 and the local location name is CHICAGO1.

Error messages for ADDDIRSHD

*ESCAPE Messages

CPF90A8

*SECADM special authority required to do requested operation.

CPF90FE

Add or change of shadow supplier &1 was not successful.

CPF905C

Error occurred trying to find a translation table.

CPF9838

User profile storage limit exceeded.

CPF9845

Error occurred while opening file &1.

CPF9846

Error while processing file &1 in library &2.

CPF9847

Error occurred while closing file &1 in library &2.



ADDSTCLGE (Add Distribution Catalog Entry) Command Description

Note: To use this command, you must have the 5722-MG1 (Managed System Services for iSeries) licensed program installed.

ADDSTCLGE Command syntax diagram

Purpose

The Add Distribution Catalog Entry (ADDSTCLGE) command is used to add an entry to the distribution catalog. A data object can optionally be loaded into the distribution repository from an iSeries library, a folder, or any of the integrated file systems.

The distribution catalog contains a list of objects that are eligible for distribution. Each catalog entry is identified by a network-wide unique name called a global name. Each catalog entry describes where the object to be distributed is located for retrieval or can be stored when received.

The catalog entry consists of the global name of the object, the name of the object's storage location (if the object exists), and attributes of the object. The data object associated with the global name can be stored as a standard iSeries object in a library, folder, stream file, or distribution repository.

Notes:

1. When the object type *LICKY is specified, it is assumed that the file that is to be cataloged is a license key file.
2. A specific format to catalog a license key file or a stream file for a global name is not required.
3. A specific global name format is required to catalog an installable object.

Required Parameters

GLBNAME

Specifies the token values of the global name. The global name is the name by which the object is known in a system network architecture (SNA) network. The global name can be a maximum of 65- n characters in length, where n is the number of tokens. A maximum of 10 tokens can be specified and each token can be a maximum of 16 characters in length.

Valid tokens consist of uppercase letters A-Z and numbers 0-9. The special characters #, \$, or @ are valid only when VERSIONATR(*CPNAME) or VERSIONATR(*NETID) is specified. In multi-lingual networks, language translation may make the value not valid when the special characters are used. Use of these characters is not recommended.

Notes:

1. The global name must contain at least one token with VERSIONATR(*STI) specified when adding a distribution catalog entry for a product.
2. If OBJTYPE(*PRODUCT) is specified, global names can have up to a maximum of 7 tokens before VERSIONATR(*STI) is specified, and a maximum of 2 tokens after VERSIONATR(*STI) is specified.
3. At least two tokens must be specified in the global name.
4. *SERVER, *TARGET, *MDDATE, and *MDTIME values are valid for token 1 or token 2. See the descriptions in the section for Element 2.

Element 1: Token 1

***NETID:** The first global name token value is a network ID generated by the command from the network attributes. VERSIONATR(*UNSPEC) or VERSIONATR(*NETID) must be specified.

***MDDATE:** This token is stored within the change request activity with the value &DATE, and replaced when distributed by the date the object was last modified.

***MDTIME:** This token is stored within the change request activity with the value &TIME, and replaced when distributed by the time the object was last modified.

***SERVER:** This token is stored within the change request activity with the value &SERVER, and replaced by the short name of the change control server when the object is distributed.

***TARGET:** This token is stored within the change request activity with the value &TARGET, and replaced by the short name of the target when the object is distributed.

global-name-token-1: Specify the first token of the global name. The first token is recommended to be the registered enterprise ID or network ID.

Element 2-10: Token 2-10

***NETID:** Identifies the global name token n value as a network ID. This value is generated from the network attributes. VERSIONATR(*UNSPEC) or VERSIONATR(*NETID) must be specified for the corresponding token.

***CPNAME:** Identifies the global name token value as a control point name. This value is generated from the network attributes. VERSIONATR(*UNSPEC) or VERSIONATR(*CPNAME) must be specified for the corresponding token.

***DATE:** Identifies the global name token value as the current date. This value is generated from the system value with the format Y1992M04D10. VERSIONATR(*UNSPEC), VERSIONATR(*ORDDATE), or VERSIONATR(*ORDCHAR) must be specified for the corresponding token.

***TIME:** Identifies the global name token value as the current time. This value is generated from the system value with the format H13M30S20. VERSIONATR(*UNSPEC), VERSIONATR(*ORDTIME), or VERSIONATR(*ORDCHAR) must be specified for the corresponding token.

***MDDATE:** This token is stored within the change request activity with the value &DATE, and replaced when distributed by the date the object was last modified.

***MDTIME:** This token is stored within the change request activity with the value &TIME, and replaced when distributed by the time the object was last modified.

***SERVER:** This token is stored within the change request activity with the value &SERVER, and replaced by the short name of the change control server when the object is distributed.

***TARGET:** This token is stored within the change request activity with the value &TARGET, and replaced by the short name of the target when the object is distributed.

global-name-token-n: Specify a token of the global name.

OBJTYPE

Specifies the type of object used.

***FILEDATA:** A file member containing data is transferred without attributes. The MBR parameter is required when OBJTYPE(*FILEDATA) is specified and OBJ(*NONE) is not specified.

***PRODUCT:** A save file containing a product packaged with the OS/400 product packaging support.

***LICKEY:** A license key file generated through the Display License Key (DSPLICKEY) command. The MBR parameter is required when *LICKEY is specified and the OBJ parameter is not *NONE.

***STMF:** A stream file containing a continuous stream of data.

***INSOBJ:** An installable object containing a combination of objects to be installed and a list of those objects associated with the name of the target library, folder, or path where the objects must be created when the installable object is installed.

object-type: Specify the OS/400 object type to be used.

Optional Parameters

OBJ Specifies the qualified name of the object to be used when adding the catalog entry.

The name of the object can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

object-name: Specify the name of the local object.

***NONE**: A data object is not loaded into the repository. OBJ(*NONE) is only valid when STGLOC(*DSTRPS) is specified.

MBR Specifies the file member to be used when OBJTYPE(*FILEDATA), OBJTYPE(*FILE), or OBJTYPE(*LICKY) is specified.

***ALL**: When STGLOC(*DSTRPS) is specified, the entire file is loaded into the distribution repository and a catalog entry is added to the catalog for the repository object. When STGLOC(*STD) is specified, a catalog entry is added for the file. MBR(*ALL) is only valid when OBJTYPE(*FILE) is specified.

***FIRST**: When STGLOC(*DSTRPS) is specified, the first file member is loaded into the distribution repository and a catalog entry is added to the catalog for the repository object. When STGLOC(*STD) is specified, a catalog entry is added for the first member.

member-name: Specify the name of the file member to be used.

DATATYPE

Specifies the type of data contained in a member that is handled as data only, without attributes. This parameter is only valid when OBJTYPE(*FILEDATA) is specified.

***UNSPEC** The data type is not specified.

***CL** The data type is control language (CL).

***REXX** The data type is REXX.

DLO Specifies the document library object name being cataloged. This parameter is required when OBJTYPE(*DOC) is specified.

***NONE**: A document is not loaded into the distribution repository. DLO(*NONE) is only valid when STGLOC(*DSTRPS) is specified.

document-name: Specify the name of the document to be cataloged. Valid document names can be up to eight characters in length and optionally qualified with a period and a one to three character extension.

FLR Specifies the folder name in which the document specified on the DLO parameter is located. This parameter is required when OBJTYPE(*DOC) or OBJTYPE(*FLR) is specified.

***NONE**: No folder or document is loaded into the repository. This parameter is only valid when STGLOC(*DSTRPS) is specified.

folder-name: Specifies the name of the folder in which the document is located or the folder name is being cataloged. The folder name can be a maximum of 63 characters in length.

STMF Specifies the stream file to be cataloged. This parameter is required when OBJTYPE(*STMF) is specified.

***NONE**: No stream file is loaded into the distribution repository. This parameter is only valid when STGLOC(*DSTRPS) is specified.

object-path-name: Specifies the path name of the stream file. This can be a path name of 1980 characters. Additional information about path names is in the Integrated File System Introduction topic in the Information Center.

STGLOC

Specifies the storage location of the data object to which the global name points.

***STD**: The global name refers to the object specified on the OBJ name parameter.

***DSTRPS**: The global name refers to an object in the distribution repository. If an object is specified on the OBJ parameter, it is loaded into the repository.

TGTRLS

Specifies the release of the operating system on which you intend to use the object. This parameter is ignored when OBJTYPE(*FILEDATA), OBJTYPE(*PRODUCT), OBJTYPE(*LICKEY), OBJTYPE(*STMF), or OBJTYPE(*INSOBJ) is specified, or when the object is a save file.

***CURRENT:** The object is used on the release of the operating system currently running on your system. If V5R2M0 is running on your system, *CURRENT means that you intend to use the object on a system with V5R2M0 installed. The object can also be used on a system with any later release of the operating system installed.

release-level: Specify the release level in the format VxRxMx. The object is used on a system with the specified release or with any later release of the operating system installed.

Valid values depend on the current version, release, and modification level, and they change with each new release.

DTACPR

Specifies the compression method used to load a member into the distribution repository.

***NONE:** The object is not compressed.

***SNA:** The object is compressed using SNA compression.

SNACPRCHR

Specifies the prime compression character used by the SNA compression algorithm when DTACPR(*SNA) is specified. Valid values range from hexadecimal '00'X through 'FF'X. This parameter is ignored when DTACPR(*NONE) is specified.

***BLANK:** The prime compression character is a blank (hexadecimal value '40'X).

character: Specify the hexadecimal value of the prime compression character to be used.

DTAACKEY

Specifies the data access key (DAK) associated with the file. If specified, the data access key name must exist in the access key table. This access key will protect the catalog entry when it is used in the change control server/client environment.

***NONE:** No access key is assigned to the catalog entry.

access-key: Specify the name of the data access key associated with the file.

AUTL Specifies the name of the authorization list of the repository object. This parameter is required when STGLOC(*DSTRPS) is specified.

QCQRPSAUTL: The default authorization list is used.

authorization-list-name: Specify the name of the authorization list to be used.

TEXT Specifies the text description of the catalog entry.

***OBJTEXT:** The text description of the local object is used. This value is not valid when OBJ(*NONE) is specified.

Note: When cataloging a stream file, and *OBJTEXT is used for the text parameter, the catalog entry description for the stream file being cataloged is generated with the text *Catalog entry for a stream file*.

***BLANK:** No text is specified.

'description': Specify no more than 50 characters of text, enclosed in apostrophes.

VERSIONATR

Specifies the version attributes of the tokens in the global name. The attributes identify the kind of token.

Element 1: Token 1

***UNSPEC:** The version attribute is not specified.

Element 2-10: Token 2-10

***UNSPEC:** The version attribute is not specified.

***NETID:** The corresponding token is a network ID.

***CPNAME:** The corresponding token is control point name.

***SYSTYPE:** The corresponding token is a system type.

***STI:** The corresponding token is a subtree indicator.

The tokens with the following attributes may be used in partial matching:

***ORDCHAR:** The corresponding token is an ordered character.

***ORDDATE:** The corresponding token is an ordered date with the format Y1989M05D03.

***ORDDEC:** The corresponding token is an ordered decimal.

***ORDTIME:** The corresponding token is an ordered time with the format H13M30S20.

MATCHATR

Specifies whether each token in the global name must match the corresponding token on the global name in the request when partial global name matching is done.

Element 1: Token 1

***MATCH:** The requested token must match this token.

Element 2-10: Token 2-10

***NOMATCH:** The requested token is not required to match this token. This match attribute should be specified for all tokens that have VERSIONATR(*ORDDATE, *ORDTIME, *ORDCHAR, or *ORDDEC) specified.

***MATCH:** The requested token must match this token.

Examples for ADDDSTCLGE

Example 1: Adding a Catalog Entry for a File

```
ADDDSTCLGE  GLBNAME(ENTID FINANCE PAYROLL DEDUCT)
             OBJTYPE(*FILE) OBJ(*LIBL/PAYDED)
             STGLOC(*STD)
```

This command adds a catalog entry for the payroll deductions file. All requests received referring to global name ENTID FINANCE PAYROLL DEDUCT are performed on the file PAYDED.

Example 2: Loading a File into the Repository for Filedata

```
ADDDSTCLGE  GLBNAME(ENTID INVENTORY *DATE)
             OBJTYPE(*FILEDATA) OBJ(APPLZ/CURINV)
             MBR(*FIRST) STGLOC(*DSTRPS)
             DTACPR(*SNA) AUTL(MYLIST)
             VERSIONATR(*UNSPEC *ORDCHAR *ORDDATE)
             MATCHATR(*MATCH *NOMATCH *NOMATCH)
```

This command loads the current inventory file APPLZ in library CURINV into the distribution repository in compressed form. The file in the distribution repository can be referred to by the global name ENTID INVENTORY Y1993M09D15. The file is secured by authorization list MYLIST.

Example 3: Loading a Document into the Repository for a Document

```

ADDSTCLGE  GLBNAME(ENTID NYPS1 SALESRPT)
            OBJTYPE(*DOC) DLO(STATUS)
            FLR(NY/SALES.RPT/APRIL)
            STGLOC(*DSTRPS)

```

This command loads document STATUS in folder path NY/SALES.RPT/APRIL into the distribution repository.

Example 4: Adding a Distribution Catalog Entry into the Repository for a Product

```

ADDSTCLGE  GLBNAME(I3IBM1 AS400 ACCOUNT V1R1M0 BASE ALL
            2924 REF 001 V5R2M0)
OBJTYPE(*PRODUCT) OBJ(ACCLIB/ACCPKG) STGLOC(*DSTRPS)
VERSIONATR(*UNSPEC *ORDCHAR *ORDCHAR *ORDCHAR *ORDCHAR
            *ORDCHAR *ORDCHAR *STI *ORDDEC *ORDCHAR)
MATCHATR(*MATCH *NOMATCH *NOMATCH *NOMATCH *NOMATCH *NOMATCH
            *NOMATCH *NOMATCH *MATCH *NOMATCH *NOMATCH)
TEXT('Accounting Base Product V1R1M0')

```

This command adds an entry to the Distribution Catalog and points to a product stored in the distribution repository.

Example 5: Adding an Entry into the Distribution Catalog for a Product

```

ADDSTCLGE  GLBNAME(I3IBM1 AS400 DISTSYS V1R2M0 BASE ALL
            2924 REF 001 V5R2M0)
OBJTYPE(*PRODUCT) OBJ(DSTSYSLIB/DSTLP) STGLOC(*STD)
VERSIONATR(*UNSPEC *ORDCHAR *ORDCHAR *ORDCHAR *ORDCHAR
            *ORDCHAR *ORDCHAR *STI *ORDDEC *ORDCHAR)
MATCHATR(*MATCH *NOMATCH *NOMATCH *NOMATCH *NOMATCH
            *NOMATCH *NOMATCH *MATCH *NOMATCH *NOMATCH)
TEXT('Distribution System Base Product V1R2M0')

```

This command adds an entry for a product to the distribution catalog and points to the save file DSTLP in the library DSTSYSLIB.

Example 6: Adding a Distribution Catalog Entry for License File Pointing to Distribution Repository

```

ADDSTCLGE  GLBNAME(COMPANY1 ACCOUNT LICENSES)
            OBJTYPE(*LICKY) OBJ(ACCNTLIB/ACCLICF)
            MBR(*FIRST) STGLOC(*DSTRPS)
            TEXT('License file for ACCOUNT product')

```

This command adds a distribution catalog entry that is stored in the distribution repository. The license file was previously generated running the DSPLICKEY command to obtain the existing licenses keys for the product ACCOUNT from the license repository.

Example 7: Adding a Distribution Catalog Entry for License File Pointing to OS/400 Standard Location

```

ADDSTCLGE  GLBNAME(COMPANY2 PURCHASE V3 LICKEYS)
            OBJTYPE(*LICKY) OBJ(PURCHLIB/PURCHLICF)
            STGLOC(*STD) MBR(V3LIC)

```

This command adds an entry in the distribution catalog pointing to the PURCHLIB/PURCHLICF in OS/400 standard location. The license file can or cannot exist at this time.

Example 8: Adding a Distribution Catalog Entry for Stream File

```

ADDSTCLGE  GLBNAME(STREAM FILE CATALOG EXAMPLE)
            OBJTYPE(*STMF) STMF('/Dir1/Dir2/Dir3/UsrFile')
            STGLOC(*DSTRPS)
            TEXT('User file cataloged as stream file')

```

This command adds a distribution catalog entry to the repository that points to a stream file that is to be loaded in the distribution repository.

Example 9: Adding a Distribution Catalog Entry for Stream File with Symbolic Link

```
ADDSTCLGE
  GLBNAME(STREAM FILE WITH SYMBOLIC LINKS CATALOG EXAMPLE)
  OBJTYPE(*STMF) STMF('FileLink')
  STGLOC(*DSTRPS)
  TEXT('Catalog a stream file using symbolic links')
```

This command adds a distribution catalog entry that points to a stream file stored in the distribution repository. However, in this example, the stream file is referred with a symbolic link. An assumption made in this example is that the current directory contains the following symbolic link:

```
FileLink = /SomeDirectory/SomeFile
```

For information about symbolic links, see the Integrated File System Introduction topic in the Information Center.

Example 10: Adding a Distribution Catalog Entry for a Stream File using *MDDATE and *MDTIME on a remote change control server.

These tokens are useful when you catalog files that are distributed regularly, such as a transaction logs or sales data. For example, if you want to retrieve a transaction log from the remote change control client (remote target) on a daily basis, you can catalog it on the remote change control server where the remote change control client is attached to with a global name such as:

```
ADDSTCLGE GLBNAME(EURO TRANSACT LOG *MDDATE *MDTIME)
  OBJTYPE(*STMF) OBJ('/transact/logfile')
  STGLOC(*STD)
  VERSIONATR(*UNSPEC *ORDCHAR *ORDCHAR *ORDDATE
    *ORDTIME)
  TEXT('Transaction log file')
```

When displaying the catalog entry it will appear as:

```
EURO.TRANSACT.LOG.&DATE.&TIME
```

To retrieve the file, you can add the next change request activity in the local system:

```
ADDCRQA CRQD(QGPL/CRQDRTV)
  ACTIVITY(ACT01)
  ACTION(*RTV)
  GLBNAME(EURO TRANSACT LOG *MDDATE *MDTIME)
  CPNAME(RMTSRV RMTCLT)
```

When the change request is submitted in the local system, the remote change control client returns the global name with the *MDDATE and *MDTIME tokens expanded to the actual date and time the file was last modified. For example, if the local file was last modified on 27 February 2001 at 15:40:00, the global name returned from the change control client is:

```
EURO.TRANSACT.LOG.Y2001M02D27.H15M40S00
```

The local system (the one to which you retrieved the file) catalogs the file under this name. This way, you can retrieve the same file every day, and each copy is cataloged under a different name. You can see the date and time of the file easily from the catalog entry.

Example 11: Adding a Distribution Catalog Entry for a Stream File using *SERVER and *TARGET tokens.

These tokens are used so that one catalog entry can be used for a file on multiple change control clients (targets). The following catalog entry is added in the change control server named "FREDSWS FREDSWS":

```
ADDSTCLGE GLBNAME(EURO SALES FILE *SERVER *TARGET)
           OBJTYPE(*STMF) STMF('/targetdir/sales/sales.file')
           STGLOC(*STD) TEXT('Clients sale file')
```

The change control server "FREDSWS FREDSWS" has two change control client (targets) attached: the change control client "FREDSWS ROSE" and "FREDSWS NORA", and the file '/targetdir/sales/sales.file' exists on both targets.

If a user at another SNA distribution services node retrieved the sales files for all the targets attached to the change control server FREDSWS by requesting the global name EURO SALES FILE *SERVER *TARGET, that user would receive the global names EURO SALES FILE FREDSWS ROSE and EURO SALES FILE FREDSWS NORA

If a user at another SNA distribution services node retrieved it from the change control server "FREDSWS FREDSWS", he would receive the global name EURO SALES FILE FREDSWS FREDSWS. Also, sending the file pointed by EURO SALES FILE *SERVER *TARGET to a focal point from the OS/400 change control server "FREDSWS FREDSWS" would result in catalog entry EURO SALES FILE FREDSWS FREDSWS sent.

Error messages for ADDSTCLGE

***ESCAPE Messages**

MSS005B

Storage limit exceeded.

MSS0116

Maximum global name length exceeded.

MSS0117

Global name token &3 not valid. Reason code &4.

MSS0118

Global name token &3 not valid. Reason code &4.

MSS0123

Internal processing error occurred.

MSS0124

Error while managing distribution catalog.

MSS0132

Catalog entry already exists for object &1.

MSS0133

Not authorized to add distribution catalog entry.

MSS0135

Distribution catalog entry not added.

MSS0136

Global name already exists.

MSS0137

Distribution catalog entry not added.

MSS0138

Object, file member, folder or document does not exist.

MSS0139
Library *LIBL not valid when local storage location is *STD.

MSS013A
Member *FIRST not valid when local storage location is *STD.

MSS013B
Catalog entry already exists for document &1.

MSS013C
Catalog entry already exists for the folder.

MSS013D
Object not loaded into distribution repository.

MSS014E
Value &2 for OBJ is more than 8 characters.

MSS014F
&1 is not an installable object.

MSS0185
Stream file not loaded into distribution repository.

MSS0188
Catalog entry already exists for the stream file.

MSS019B
Description for object &1 not found.

MSS019D
Folder description not found.

MSS01A1
Object type &2 not supported.

MSS01A6
Global name format is only valid for OS/400 products.

MSS01A7
Target release in global name token &4 not supported.

MSS01A9
Stream file not found.

MSS01AA
Stream file cannot be accessed. Reason code &1.

MSS01D2
File not cataloged.

MSS01D5
Match attribute &4 for token &3 not valid.

MSS01D6
Length of global name token &3 not valid.

MSS01D7
Value of global name token &3 not valid.

MSS01D8
Global name not valid.

MSS01DA
Version attribute &4 for token &3 not valid.

MSS01E1

Library must be QSYS or *LIBL.

MSS01FB

Object specified does not contain a product.

MSS01FC

Global name not valid for product.

MSS01FE

Target release &1 not valid.



ADDDSTLE (Add Distribution List Entry) Command Description

ADDDSTLE Command syntax diagram

Purpose

The Add Distribution List Entry (ADDDSTLE) command adds new entries to an existing distribution list. A distribution list can include local, remote, indirect, and independent work station users. It can also include remote distribution lists, but not local distribution lists.

The ADDDSTLE command allows up to 300 entries to be added to a distribution list at one time. In addition, up to 50 local distribution list IDs can be specified whose members are all to become part of this list.

The distribution list must exist before this command can be run. The Create Distribution List (CRTDSTL) command can be used to create a new distribution list.

Restriction: The user of this command must have security administrator authority to add entries to a distribution list owned by someone else. Users can add entries to a distribution list they have created without restrictions.

Required Parameter

LSTID Specifies the two-part list identifier of the distribution list that is to have entries added.

Element 1: List Identifier

list-ID: Specify the list identifier (ID) of the distribution list.

Element 2: List Qualifier

list-ID-qualifier: Specify the list ID qualifier of the distribution list.

Note:

The distribution list identifier has two parts, the ID and the qualifier, separated by at least one space. If lowercase characters are specified, the system changes them to uppercase.

The naming rules for the two-part list ID are identical to the rules for the user ID and address. A complete description of these rules is in the SNA Distribution

Services  book.

Optional Parameters

USRID

Specifies the user ID, address, and description of the users for whom additions to the distribution list are made. Both the user ID and address must be provided and must be separated by at least one space. If any lowercase characters are specified, the system changes and stores them as uppercase.

A list ID and address can be used in place of the user ID and address to identify a remote distribution list ID that is added to the distribution list. A remote distribution list must be defined as a remote user in the directory, or a default *ANY entry must exist.

***NONE:** No user ID or list ID is specified. Members are added to this list using the FROMLSTID parameter. If the FROMLSTID parameter also specifies *NONE, an error message is returned.

Element 1: User ID

user-ID: Specify the user ID (or the valid list ID for a remote list entry) of the user for whom the entry is made.

Element 2: User Address

user-address: Specify the user address (or the valid list address for a remote list entry) of the user for whom the entry is made. Separate the ID and address by at least one space.

Element 3: User Description

***FIRST:** The first description in the specified user ID and address (or list ID and address) is added. If only one entry exists, it is the one added to the list.

user-description: Specify the description for the user. If a list ID is specified, enter the list description. The description can be up to 50 characters in length. The description entered must be exactly the same as a directory entry or an error message is returned.

FROMLSTID

Specifies the name of an existing distribution list whose entries are added to this list. Up to 50 different list IDs can be specified. Duplicate entries are not removed. Duplicates are defined as entries that have the same user ID, address, and description.

***NONE:** No list ID is specified. Members are added to this list using the USRID parameter. If *NONE is specified for the USRID, an error message is returned.

Element 1: From List ID

from-list-ID: Specify the list ID of a distribution list whose entries are added to this list.

Element 2: From List Qualifier

from-list-qualifier: Specify the list qualifier of a distribution list whose entries are added to this list.

CMDCHRID

Specifies the character identifier (graphic character set and code page) for data being specified as parameter values on this command. This character identifier (CHRID) is related to the display device used to specify the command. More information about CHRID processing is in the

Application Display Programming  book.

***SYSVAL:** The system determines the graphic character set and code page values for the command parameters from the QCHRID system values.

***DEVVD:** The system determines the graphic character set and code page values for the command parameter from the display device description where the command is entered. This option is valid only when specified from an interactive job. If this value is specified in an interactive CL program or a batch job, an error message is sent.

Element 1: Character Set

graphic-character-set: Specify the character set used to create the command parameters. Valid values range from 1 through 9999 characters.

Element 2: Code Page

code-page: Specify the code page. Valid values range from 1 through 9999.

Example for ADDDSTLE

```
ADDDSTLE LSTID(CHICAGO DLIST)
  USRID((HURST NEWYORK 'Manager of Payroll')
  (LEE DEPT554 *FIRST)
  (BOCA DLIST 'Remote Distribution
  list for Boca') (ERIC WAREHSE))
  FROMLSTID((DEPT48K DLIST) (ALLMGRS DLIST))
```

This command specifies that four user IDs are added to the distribution list CHICAGO DLIST. The third user ID is in fact a remote distribution list. The fourth user ID (ERIC WAREHSE) defaults to the first description for that user ID. In addition, all of the entries in two distribution lists are added to this distribution list.

Additional Considerations

The list ID specified to have entries added to it must already exist in the directory. If the list does not exist, an error message is returned. The Create Distribution List (CRTDSTL) command is used to create a new distribution list.

Up to 300 sets of user IDs, addresses and user descriptions can be entered on this command. Each set of user ID and address is examined to determine whether it exists in the directory. If it is not in the directory, the directory is searched for an entry with user ID *ANY and an address that matches the address of the specified user. If this is found, the user ID and address is added to the distribution list. If it is not found, but a *ANY *ANY entry exists, the user ID is added to the list. Otherwise an error message is returned, stating that the user ID is not valid.

The command processes each set of user ID, address, and description. Each invalid set returns an error message, but the valid sets are added to the distribution list. At the end of processing of the command, a final message is returned indicating how many entries are in fact added to the distribution list, and how many entries are invalid.

The command allows input of up to 50 distribution list IDs whose members are added to the specified distribution list. If any list ID is not found in the directory, an error message is returned. Like the sets of user IDs, and addresses, any valid list IDs will have their members added to the distribution list, and a message is returned for each invalid list ID.

Error messages for ADDDSTLE

***ESCAPE Messages**

CPF9024

System cannot get correct record to finish operation.

CPF905C

Error occurred trying to find a translation table.

CPF9090

No entries added to distribution list &1 &2.

CPF9091

&1 entries added and &2 lists copied to list &3 &4. &5 entries not added and &6 lists not copied.

CPF9096

Cannot use CMDCHRID(*DEV D), DOCCHRID(*DEV D) in batch job.

CPF9838

User profile storage limit exceeded.

CPF9845

Error occurred while opening file &1.

CPF9846

Error while processing file &1 in library &2.

ADDDSTQ (Add Distribution Queue) Command Description

ADDDSTQ Command syntax diagram

Purpose

The Add Distribution Queue (ADDDSTQ) command adds an entry to the distribution services queue table. Distribution queues are used to store distributions before they are sent or forwarded to other systems.

Interactive display support is provided by the Configure Distribution Services (CFGDSTSRV) command.

More information about configuring a distribution network is in the SNA Distribution Services  book.

Distribution queue names are translated to the graphic character set and code page 930 500, using the job's coded character set identifier (CCSID).

Restrictions:

1. This command is shipped with public *EXCLUDE authority, and the QPGMR and QSYSOPR user profiles have private authorities to use the command.
2. The combination of remote location name, mode, remote network identifier, and local location name must be unique within the type of distribution queue. This combination does not need to be unique within the system, for SNA distribution services (SNADS) distribution queues in the distribution services queue table (SNADS-type distribution queues), and for SystemView* distribution services (SVDS) distribution queues (SVDS-type distribution queues). The default value *LOC, which can be specified on the RMTNETID parameter and the LCLLOCNAME parameter, and the default value *NETATR, which can be specified on the MODE parameter, represent any possible values that the system determines are not already configured for another SNADS or SVDS distribution queue of the same type.
3. A unique remote location name must be specified for each RPDS-type distribution queue in the queue table. RPDS queues do not use modes, remote network identifiers, or local location names.
4. Configuration in the routing table is not required for SVDS-type distribution queues. SVDS queues may be configured optionally in the SNADS routing table. However, normal SNADS mail can neither be routed to change management queues nor be received through change management connections, and change management connections can neither be routed to SNADS queues nor be received through SNADS connections.
5. SVDS-type distribution queues can support only a single queue view (the queue is not divided into normal and priority halves). For configurations and operations purposes, only the normal queue is specified.
6. Messages that report errors about distribution queues may display or print different characters than the user entered for the distribution queue name because of internal system transformations. Similarly (depending on the language used for the work station), the internal value for a distribution queue name may differ from the characters shown on the Work with Distribution Queue (WRKDSTQ) command. An error may be reported if the character-string value specified for the DSTQ parameter does not match

the rules for an internal distribution queue value or if it does not match the internal value for any defined distribution queue (ignoring case differences).

Required Parameters

DSTQ Specifies a maximum of 16 characters for the name of the distribution queue being added to the distribution services queue table.

RMTLOCNAME

Specifies the name of the remote location where distributions are sent from this distribution queue. The remote location name must be configured in the device description of the device used when sending distributions to another system from this distribution queue.

Optional Parameters

DSTQTYPE

Specifies the type of distribution queue being added.

***SNADS:** SNADS is the distribution queue type. SNADS queues are used to send distributions within a SNADS network.

***DLS:** Document library services (DLS) is the distribution queue type. DLS queues are used to communicate between the local system and document library services on a remote system.

***RPDS:** RPDS is the distribution queue type. RPDS queues are used to communicate between the local system and the iSeries 400 VM/MVS bridge, or JES (2,3) on System/370*-type systems, and for the SNADS extended bridge function of the Communications Utilities/400 licensed program.

***SVDS:** SystemView distribution services (SVDS) is the distribution queue type. SVDS queues support the communications bridge between a SNADS network and System Manager for iSeries 400 change management. An SVDS queue must be defined in order to receive from as well as send to a remote system using change management.

MODE Specifies the name of the mode that defines the sessions on the device used by the distribution queue. This parameter is ignored if *RPDS is specified on the DSTQTYPE parameter.

***NETATR:** The mode name specified in the network attributes is used.

mode-name: Specify a maximum of 8 characters for the name of the mode. The mode name cannot be CPSVCMG or SNASVCMG; these mode names are reserved for system use.

RMTNETID

Specifies the remote network identifier of the remote network to which this distribution queue sends distributions. This parameter is ignored if *RPDS is specified on the DSTQTYPE parameter.

***LOC:** The remote network identifier defined in the device description used by this distribution queue is used.

***NONE:** No remote network identifier is specified.

remote-network-ID: Specify the remote network identifier.

LCLLOCNAME

Specifies the name used to identify the local system to remote systems in the network. Whenever possible, the name should be the same as the local system name.

***LOC:** The local location name defined in the device description used by this distribution queue is used.

local-location-name: Specify a maximum of 8 characters for the local location name.

NRMPY

Specifies the queue sending conditions for distributions having a service level of data low.

Element 1: Send Time

The send time is the time period during which queued distributions of this priority are sent from this distribution queue. If no time period is specified, the transmissions are controlled by queue depth and are not related to time. The time must be specified in the 24-hour format hhmm where hh= hours and mm = minutes.

***ALWAYS:** Distributions of this priority are sent from this distribution queue regardless of the time of day.

from-time: Specify the time of day at which the system starts sending distributions of this priority from this distribution queue if the value specified for send depth is reached. If *from-time* is specified, *to-time* must also be specified.

to-time: Specify the time of day at which the system stops sending distributions of this priority from this distribution queue. If *to-time* is specified, *from-time* must also be specified.

Element 2: Force Time

Force time is a specific time during which distributions of this priority are sent regardless of queue depth. If *ALWAYS is specified for the send time, the force time can be set to any time of day. If a specific to-time and from-time is specified for the send time, the force time must occur within that time period. The time must be specified in the 24-hour format hhmm where hh = hours and mm = minutes.

***NONE:** No force time is specified.

force-time: Specify the force time.

Element 3: Depth

Queue depth specifies the number of distributions of this priority that are on the queue before sending begins.

1: Distributions are sent when they are put on the queue.

***MANUAL:** Distributions are sent only when an operator manually sends them using the Work with Distribution Queue (WRKDSTQ) command or the Send Distribution Queue (SNDDSTQ) command.

depth: Specify the number of distributions of this priority that are on this distribution queue before any are sent. Valid values range from 1 through 999.

HIGHPTY

Specifies the queue sending conditions for distributions having a service level of fast, status, or data high. On this parameter, time must be specified in the 24-hour format hhmm where hh = hours and mm = minutes.

Element 1: Send Time

***ALWAYS:** Distributions of this priority are sent from this distribution queue regardless of the time of day.

from-time: Specify the time of day at which the system starts sending distributions of this priority from this distribution queue if the value specified for send depth is reached. If *from-time* is specified, *to-time* must also be specified.

to-time: Specify the time of day at which the system stops sending distributions of this priority from this distribution queue. If *to-time* is specified, *from-time* must also be specified.

Element 2: Force Time

***NONE:** No force time is specified.

force-time: Specify the force time.

Element 3: Depth

1: Distributions are sent when they are put on the queue.

***MANUAL**: Distributions are sent only when an operator manually sends them using the WRKDSTQ command or the SNDDSTQ command.

depth: Specify the number of distributions of this priority that are on this distribution queue before any are sent. Valid values range from 1 through 999.

RTYNBR

Specifies the maximum number of times the system attempts to resend distributions from this distribution queue after a failure occurs. This parameter applies to communications line failures and recoverable distribution failures on a remote system. The SNADS job serving this distribution queue ends when the number of retries is exceeded.

3: The system attempts to resend distributions a maximum of 3 times after a failure.

number: Specify the maximum number of times the system can attempt to resend distributions after a failure. Valid values range from 0 through 9999.

RTYITV

Specifies the interval (in minutes) between each retry attempt.

5: The number of minutes between retries is 5.

minutes: Specify the interval between retries. Valid values range from 0 through 9999 minutes.

SNdq Specifies whether this distribution queue ignores the send time and depth values specified on the NRMPty and HIGHPTY parameters and begins sending when a distribution is received from the SNADS system to which the queue sends its distributions.

This parameter is valid only if *SNADS is specified on the DSTQTYPE parameter.

***NO**: Distributions are sent from this queue only when the queue's sending conditions are met.

***YES**: This distribution queue begins sending when distributions are received from the SNADS system to which the queue sends its distributions, regardless of the queue's other sending conditions. Distributions are automatically sent for manual queues (queues that have no specified depth variable).

Examples for ADDDSTQ

Example 1: Adding a SNADS Distribution Queue

```
ADDDSTQ DSTQ(CHICAGO) RMTLOCNAME(CHICAGOLU)
        MODE(NEWMODE)
```

This command adds a distribution queue named CHICAGO. The queue uses remote location name CHICAGOLU and mode NEWMODE when sending SNADS distributions.

Example 2: Adding a DLS Distribution Queue

```
ADDDSTQ DSTQ(DLSQUEUE) DSTQTYPE(*DLS)
        RMTLOCNAME(DLSLU) MODE(DLSMODE)
```

This command adds a DLS type of distribution queue named DLSQUEUE. The queue uses remote location name DLSSLU and mode DLSSMODE when sending DLS requests.

Example 3: Adding an SVDS Distribution Queue

```
ADDDSTQ DSTQ(CHICACM) RMTLOCNAME(CHICAGOLU)
        DSTQTYPE(*SVDS)
```

This command adds an SVDS type of distribution queue named CHICACM. The queue uses remote location name CHICAGOLU when sending and receiving SVDS change management distributions.

Error messages for ADDDSTQ

*ESCAPE Messages

CPF8802

Distribution queue &1 was not found.

CPF8807

Error occurred while using QSNADS journal.

CPF8809

Errors detected on SNADS internal queues.

CPF881D

High priority data not allowed for *SVDS distribution queues

CPF8826

Distribution queue entries exist for distribution queue &1.

CPF8827

Routing table entries exist for distribution queue &1.

CPF8828

Remote document library entries exist for *DLS distribution queue &1.

CPF8833

Distribution queue &1 already exists.

CPF8849

Queue &1 in use by another distribution services function.

CPF9845

Error occurred while opening file &1.

CPF9846

Error while processing file &1 in library &2.

CPF9847

Error occurred while closing file &1 in library &2.

CPF9899

Error occurred during processing of command.

ADDDSTRTE (Add Distribution Route) Command Description

ADDDSTRTE Command syntax diagram

Purpose

The Add Distribution Route (ADDDSTRTE) command adds an entry to the distribution services routing table. The routing table determines which distribution queue receives a distribution on its way to a particular destination.

Distributions are routed to distribution queues based on service levels. One or more service levels must be specified for each routing table entry. This system will not route distributions for service levels that have not been configured. Normally, all service levels routed to the same destination use the same distribution queue. However, the user can configure several distribution queues for one destination based on distribution service levels.

Interactive display support is provided by the Configure Distribution Services (CFGDSTSRV) command.

System names, system group names, and distribution queue names are translated to the graphic character set and code page 930 500, using the job's coded character set identifier (CCSID).

Restrictions:

1. This command is shipped with public *EXCLUDE authority, and the QPGMR and QSYSOPR user profiles have private authorities to use the command.
2. An error occurs if a distribution route specifying a SystemView distribution services (SVDS) type of distribution queue includes another type of distribution queue (such as SNA distribution services (SNADS) or VM/MVS bridge (RPDS)).

Required Parameter

SYSNAME

Specifies the system name and group name of the remote system that is the destination for this routing table entry. The local system name cannot be specified unless a group name is also specified.

Element 1: System Name

***ANY:** *ANY is used for the system name. When *ANY and a system group name are specified, any system in the group is the destination for the routing table entry. Only one *ANY value is allowed for each group in the routing table.

system-name: Specify a maximum of 8 characters for the name of the remote system that is the destination for this routing table entry.

Element 2: System Group Name

***ANY:** *ANY is used for the system group name. *ANY can be specified for the group name only if *ANY is also specified for the system name. Only one SYSNAME(*ANY *ANY) entry is allowed in the routing table and is used to resolve a distribution destination that does not match any other routing table entries.

system-group-name: Specify a maximum of 8 characters for the system group name. The system name and group name must be separated by at least one blank.

Optional Parameters

FAST Specifies the distribution queue and maximum hop count to the destination system for fast service level distributions. The fast service level is the highest priority level.

The maximum hop count is the maximum number of times in a SNADS network that a distribution can be routed back and forth (hop) between the systems participating in the SNADS level routing, including the hop to the final destination system. The maximum hop count does not include the hops made by advanced peer-to-peer networking (APPN) routing. If the maximum number of hops is exceeded, the distribution is ended and an error is sent to the user who originally sent the distribution.

***NONE:** No distribution queue is specified for distributions requiring a fast service level. Distributions requiring fast service cannot be routed using this routing table entry.

Element 1: Distribution Queue

distribution-queue-name: Specify the name of the distribution queue to which fast distributions using this routing entry are sent. The distribution queue must already exist and cannot be a DLS (document library services) type of queue.

Element 2: Maximum Hop Count

***NETATR:** The system network attribute value for the maximum hop count is used. The current system default value can be displayed using the Display Network Attributes (DSPNETA) command.

hop-count: Specify the maximum hop count. Valid values range from 1 through 255.

STATUS

Specifies the distribution queue and maximum hop count to the destination system for status service level distributions. The status service level is used for network status and other feedback information.

***NONE:** No distribution queue is specified for distributions requiring a status service level. Distributions requiring status service cannot be routed using this routing table entry.

Element 1: Distribution Queue

distribution-queue-name: Specify the name of the distribution queue to which status distributions using this routing entry are sent. The distribution queue must already exist and cannot be a DLS-type queue.

Element 2: Maximum Hop Count

***NETATR:** The system network attribute value for the maximum hop count is used. The current system value can be displayed using the Display Network Attributes (DSPNETA) command.

hop-count: Specify the maximum hop count. Valid values range from 1 through 255.

DATAHIGH

Specifies the distribution queue and maximum hop count to the destination system for data high service level distributions. The data high service level is used for high priority data traffic.

***NONE:** No distribution queue is specified for distributions requiring a data high service level. Distributions requiring data high service cannot be routed using this routing table entry.

Element 1: Distribution Queue

distribution-queue-name: Specify the name of the distribution queue to which data high distributions using this routing entry are sent. The distribution queue must already exist and cannot be a DLS-type queue.

Element 2: Maximum Hop Count

***NETATR:** The system network attribute value for the maximum hop count is used. The current system value can be displayed using the Display Network Attributes (DSPNETA) command.

hop-count: Specify the maximum hop count. Valid values range from 1 through 255.

DATALOW

Specifies the distribution queue and maximum hop count to the destination system for data low service level distributions. The data low service level is used for most data traffic.

***NONE:** No distribution queue is specified for distributions requiring a data low service level. Distributions requiring data low service cannot be routed using this routing table entry.

Element 1: Distribution Queue

distribution-queue-name: Specify the name of the distribution queue to which data low distributions using this routing entry are sent. The distribution queue must already exist and cannot be a DLS-type queue.

Element 2: Maximum Hop Count

***NETATR:** The system network attribute value for the maximum hop count is used. The current system value can be displayed using the Display Network Attributes (DSPNETA) command.

hop-count: Specify the maximum hop count. Valid values range from 1 through 255.

TEXT Specifies the text that briefly describes the distribution route. More information on this parameter is in Commonly used parameters.

Note:

Double-byte character set (DBCS) characters can be entered on this parameter.

***BLANK:** No text is associated with the new distribution route.

'description': Specify no more than 50 characters of text, enclosed in apostrophes.

Examples for ADDDSTRTE**Example 1: Adding an Entry for an Adjacent System**

```
ADDDSTRTE  SYSNAME(SYSTEMA GROUPA)
           FAST(SYSTEMA)  STATUS(SYSTEMA)
           DATAHIGH(SYSTEMA)  DATALOW(SYSTEMA)
```

This command adds a routing table entry for a system that is directly connected to this system (via a physical advanced program-to-program communications (APPC) connection or a logical APPN connection). The distribution queue is given the same name as the destination system. The hop count defaults to the system default value.

Example 2: Adding a Generic Routing Table Entry

```
ADDDSTRTE  SYSNAME(*ANY GROUPNM1)
           FAST(SYSTEMA)  STATUS(SYSTEMA)
           DATAHIGH(SYSTEMA)  DATALOW(SYSTEMA)
```

This command adds a routing table entry for all systems in system group GROUPNM1. The distribution queue SYSTEMA is used to route distributions to all systems in the group.

Error messages for ADDDSTRTE***ESCAPE Messages****CPF8801**

Document library services (*DLS) queue &1 not allowed in routing table.

CPF8802

Distribution queue &1 was not found.

CPF8807

Error occurred while using QSNADS journal.

CPF881E

Distribution route contains combination of distribution queues that is not allowed.

CPF881F

System group name cannot be blank for distribution route

CPF8815

Routing table entry &1 &2 not found.

CPF8831

Entry &1 &2 already exists in routing table.

CPF8837

System name/Group &1 &2 in use by another distribution services function.

CPF8849

Queue &1 in use by another distribution services function.

CPF9845

Error occurred while opening file &1.

CPF9846

Error while processing file &1 in library &2.

CPF9847

Error occurred while closing file &1 in library &2.

CPF9899

Error occurred during processing of command.

ADDDSTSYSN (Add Distribution Secondary System Name) Command Description

ADDDSTSYSN Command syntax diagram

Purpose

The Add Distribution Secondary System Name (ADDDSTSYSN) command adds an entry to the distribution services secondary system name table. The table contains names of all of the alternate (or alias) system names for which the local system receives and may redirect distributions. The SNA distribution services (SNADS) function automatically receives distributions with the local system as the destination system name, so the local system cannot be added to the secondary system name table.

This command can be used to:

- Give the local system a group name
- Rename the local system
- Combine two systems into one system or divide one system into two systems
- Allow a system to receive mail destined for a system from which many users have moved

Interactive display support is provided by the Configure Distribution Services (CFGDSTSRV) command.

System names and system group names are translated to the graphic character set and code page 930 500, using the job's coded character set identifier (CCSID).

Restrictions:

1. This command is shipped with public *EXCLUDE authority, and the QPGMR and QSYSOPR user profiles have private authorities to the command.
2. The secondary system name table does not operate with SystemView distribution services (SVDS) types of distributions.

Required Parameter

SYSNAME

Specifies the alternate system name and system group name being added to the distribution services secondary system name table. The current system name cannot be specified unless a group name is also specified.

Element 1: System Name

system-name: Specify a maximum of 8 characters for the name of the system for which the local system is to receive distributions.

Element 2: System Group Name

system-group-name: Specify a maximum of 8 characters for the system group name of the system for which the local system is to receive distributions. The system name and group name must be separated by at least one blank.

Optional Parameter

TEXT Specifies the text that briefly describes the secondary system name. More information is in Commonly used parameters.

Note: Double-byte character set (DBCS) characters can be entered on this parameter.

***BLANK:** No text is associated with the secondary system name.

'description': Specify a maximum of 50 characters of text, enclosed in apostrophes.

Example for ADDDSTSYSN

```
ADDDSTSYSN  SYSNAME(SYS2LAJ1 ROCHESTR)
```

This command adds the system named SYS2LAJ1 ROCHESTR to the distribution services secondary system name table. The local system will receive distributions that contain SYS2LAJ1 ROCHESTR as the destination system name.

If the local system is named SYS2LAJ1, this command allows the local system to participate in a network that requires a group name of ROCHESTR for each participating system.

Error messages for ADDDSTSYSN

*ESCAPE Messages

CPF8807

Error occurred while using QSNADS journal.

CPF8818

Secondary system name table entry &1 &2 not found.

CPF8835

System name/Group &1 &2 already specified.

CPF8837

System name/Group &1 &2 in use by another distribution services function.

CPF9845

Error occurred while opening file &1.

CPF9846

Error while processing file &1 in library &2.

CPF9847

Error occurred while closing file &1 in library &2.

CPF9899

Error occurred during processing of command.

ADDDLOAUT (Add Document Library Object Authority) Command Description

ADDDLOAUT Command syntax diagram

Purpose

The Add Document Library Object Authority (ADDDLOAUT) command gives a user access to a document or folder. This command allows the user to specify authority for others in the following ways:

- Give specific authority to a user
- Give a set of users authority by specifying a previously defined authorization list
- Give a group of users access by specifying an access code

Restriction: The user of this command must be the owner of the objects, or have *ALL authority to the objects or *ALLOBJ special authority.

Required Parameter

DLO Specifies the name of the document or folder to which authority is added.

***ALL:** All objects in the specified folder have authority added. If *ALL is specified, the FLR parameter is required.

***SYSOBJNAM:** The system object name specified in the SYSOBJNAM parameter has authority added.

document-library-object-name: Specify the user-assigned name of the document or folder. Up to 12 characters can be specified.

Optional Parameters

FLR Specifies the name of the folder that contains the document.

***NONE:** A folder name is not specified. If DLO(document-library-object-name) is specified and the object is located in a folder, FLR(*NONE) cannot be specified. If DLO(*ALL) is specified, FLR(*NONE) cannot be specified.

folder-name: Specify the user-assigned name of the folder. The folder name can consist of a series of folder names if the object specified in the DLO parameter is located in a folder that is contained in another folder. Up to of 63 characters can be specified.

SYSOBJNAM

Specifies the system object name. This parameter is valid only when DLO(*SYSOBJNAM) or DOCL(*SYSOBJNAM) is specified. A full ten characters must be specified.

USRAUT

Specifies the name of a specific user and the user's authority.

***NONE:** No additional user authority is added.

user-profile-name: Specify the name of the user profile for whom specific authority is added.

The possible authority values are:

***ALL:** The user can perform all operations except those limited to the owner or controlled by authorization list management authority. The user can control the object's existence, specify the security for the object, change the object, and perform basic functions on the object. The user also can change ownership of the document library object.

***CHANGE:** The user can perform all operations on the object except those limited to the owner or controlled by object existence authority and object management authority. The user can change and perform basic functions on the object. Change authority provides object operational authority and all data authority.

***USE:** The user can perform basic operations on the document library object, such as display or print document or folder content and description information. The user is prevented from changing the object. Use authority provides object operational authority and read authority.

***EXCLUDE:** The user cannot access the document library object.

AUTL Specifies that the authority for the object named in the DLO parameter comes from the authorization list containing the list of users and their authorities.

***NONE:** An authorization list is not specified.

authorization-list-name: Specify the name of the authorization list used.

ACC Specifies the access codes that are added. The access code must be defined to the system (using the Add Access Code (ADDACC) command) before it can be assigned to an object. An access code of zero allows public *USE authority for the object. An access code of zero cannot be added to an object if the document library object is marked personal.

***NONE:** No access code is added.

access-code: Specify the access code assigned to the object. An access code can be any number between 0 and 2047.

Example for ADDDLOAUT

```
ADDDLOAUT DLO(*ALL) USER(MIKE (*CHANGE))
AUTL(*NONE) FLR(MYFLR) ACC(1023)
```

This command adds *CHANGE authority for user MIKE to all objects in the folder MYFLR. An access code of 1023 was also added to the object.

Error messages for ADDDLOAUT

*ESCAPE Messages

CPF8A75

Not authorized to access folder &1.

CPF8A77

Folder &1 not found.

CPF8A78

Folder &1 in use.

CPF8A79

Folder &1 is logically damaged.

CPF8A80

Document &2 in use in folder &1.

CPF8A82

Document &2 not found in folder &1.

CPF8A83

Not authorized to access document &2 in folder &1.

CPF8A88

Operation not allowed on document &2 in folder &1.

CPF8A89

Document &2 in folder &1 is logically damaged.

CPF90BA

Authority request for document library object failed.

CPF901F

*AUTL was specified for a user other than *PUBLIC.

CPF9073

No authority to view or change the security of document library object &1.

CPF908A

Requester &1 not enrolled.

CPF908B

Document library object not found.

CPF908E

&1 objects changed; &2 objects not changed.

CPF909A

Document &2 in folder &1 is damaged.

CPF9095

Folder &1 is damaged.

ADDEMLCFGE (Add Emulation Configuration Entry) Command Description

ADDEMLCFGE Command syntax diagram

Purpose

The Add Emulation Configuration Entry (ADDEMLCFGE) command is used to add a configuration entry for a 3270 device emulation session to the configuration file. You can use this command to specify display and printer emulation options, such as setting the maximum image size for a display, or tracing the data stream for a printer.

The values you specify are used during an emulation session when the configuration entry is specified on the Start 3270 Display Emulation (STREML3270) or Start Printer Emulation (STRPRTEML) commands.

Required Parameter

EMLCFGE

Specifies the name of the emulation configuration entry that you are adding to the configuration file.

Note:

The configuration file is shipped with the default configuration entry for 3270 emulation sessions, QEMDFTCFGE.

Optional Parameters

EMLDBGJOB

Specifies whether to trace the printer data stream being passed to the printer function manager when a job that is using printer emulation is being debugged with the Trace Job (TRCJOB) command.

***NOTRACE:** The printer data stream is not traced in the TRCJOB output.

***TRACE:** The printer data stream is traced in the TRCJOB output.

EMLSIG

Specifies how an iSeries 400 emulation job that is not in send mode responds to a Systems Network Architecture (SNA) request for permission to send data (signal).

***SAVE:** The emulation job stores the SNA signal, and if appropriate grants the host permission to send when a piece of data with the change direction indicator (CD) on it is received.

***IGNORE:** The emulation job does not grant the host permission to send.

EMLATR

Specifies how the 3270 emulation job responds to an incorrect character attribute or attribute value that is received during a single-byte character set (SBCS) session. In 3270 data stream terminology, a **character attribute** is a set attribute instruction (SA order) and an **attribute value** is a value on which the instruction operates (value).

***IGNORE:** The emulation job ignores the incorrect value.

***REJECT:** The emulation job sends a negative response to the data stream containing the incorrect character attribute or attribute value.

EMLMAXSCR

Specifies the maximum size of the image to be shown on the display screen.

***DEV D:** The maximum size is whatever the device can support.

***MOD2:** The maximum size is 24 rows by 80 columns.

***MOD5:** The maximum size is 27 rows by 132 columns.

Note:

If the device does not support 27 rows by 132 columns, the maximum size defaults to 24 rows by 80 columns.

EMLTRC

Specifies whether trace points are issued by the data stream translation component of the Licensed Internal Code (LIC) when the following are true:

- The job is using the data stream translation API (application program interface).
- The Trace Job (TRCJOB) command is running on the job using the data stream translation API. The TRCJOB was started before the first call to the API for that session.
- The value for EMLTRC was specified on the QEMDFTCFGE emulation configuration entry.
- A VLIC source/sink trace is also running.

***NOTRACE:** Data stream translation trace points are not issued.

***TRACE:** Data stream translation trace points are issued.

EMLSTR

Specifies whether to call the Trace Job (TRCJOB) command when a printer emulation job is started.

***NOTRACE:** The TRCJOB command is not called.

***TRACE:** The TRCJOB command is called.

EMLINLSCN

Specifies whether to show the Emulation Initialization In Progress display when a 3270 device emulation session is starting.

***YES:** The display is shown.

***NO:** The display is not shown.

EMLGRDLIN

Specifies whether to suppress gridlines on the display.

Note: This parameter does not apply when strategic gridlines are used.

***NO:** Gridlines are not suppressed.

***YES:** Gridlines are suppressed.

EMLDBCS

Specifies whether to support the DBCS-graphic character string for input fields.

Note: This parameter is valid only for customer applications that specify field attributes or that specify character attributes for the entire input field. If only part of a field is defined with the DBCS-graphic attribute, you will get unpredictable results.

***NO:** The DBCS-graphic character string is not supported.

***YES:** The DBCS-graphic character string is supported.

EMLPRTFMT

Specifies whether to use the values supplied by the STRPRTEML command for the number of lines (NUMLIN) parameter and the number of columns (NUMCOL) parameter instead of using the values supplied by the printer file.

Note: This parameter is valid only for SNA character string (SCS) printer sessions.

***NO:** The STRPRTEML command values for lines and columns are not used.

***YES:** The STRPRTEML command values for lines and columns are used.

Note: The STRPRTEML command values remain in effect until a set horizontal format or a set vertical format command is received from the host.

EMLBUF

Specifies whether the foreground or background buffer is used when base gridlines are displayed. This parameter does not apply when strategic gridlines are used.

***FRONT:** The foreground buffer is used.

***BACK:** The background buffer is used.

EMLVLG

Specifies whether to generate a VLIC log when the datastream translation routines send a negative response to the host because a command or order is not valid.

***NO:** The VLIC log is not generated.

***YES:** The VLIC log is generated. The emulation session continues.

EMLSCS

For SCS printer sessions only, this parameter says whether to follow architecture and default to a page size or page width of 1 line when a bad set horizontal format (SHF) or a bad set vertical format (SVF) is received.

***NO:** The architecture is followed. Page size is one line.

***YES:** The architecture is not followed, page size and width default to values entered on the start printer emulation (STRPRTEML) command.

EMLSNACLR

Specifies how the 3270 emulation job recovers when an attempt to get data sent by the host system fails after an SNA CLEAR command is received.

Note:

This parameter is valid only for display emulation sessions.

***RETRY:** The emulation job tries again to get the data.

***IGNORE:** The emulation job does not try again to get the data.

Examples for ADDEMLCFGE

Example 1: Adding a Configuration Entry

```
ADDEMLCFGE EMLCFGE(ARTSDEPT) EMLMAXSCR(*MOD5)
EMLGRDLIN(*YES)
```

This command adds an emulation configuration entry named ARTSDEPT for a display with a maximum screen image of 27 rows by 132 columns. Field outlining does not show on the display.

Example 2: Adding a Configuration Entry for a Printer

```
ADDEMLCFGE EMLCFGE(FASBPRINT) EMLDBGJOB(*TRACE)
EMLATR(*REJECT)
```

This command adds an emulation configuration entry named FASBPRINT to the configuration file. The FASBPRINT configuration entry traces the printer data stream when a job is running with a trace on it and sends a negative response when incorrect data is received.

Error messages for ADDEMLCFGE

*ESCAPE Messages

CPF853A

Emulation entry &1 already exists.

CPF854B

Internal error in emulation configuration routines.

ADDENVVAR (Add Environment Variable) Command Description

ADDENVVAR Command syntax diagram

Purpose

The Add Environment Variable (ADDENVVAR) command adds an environment variable consisting of a character string in the form 'environment variable name=environment variable value'. Environment variables can be used, for example, to specify configuration values to application programs on systems that are compliant with the Single UNIX Specification.

If you are not ready to set the environment variable value, you can use this command to add an environment variable with a null value. You can then use the Add Environment Variable (ADDENVVAR) or the Change Environment Variable (CHGENVVAR) command to associate the environment value with the environment variable name.

Restriction: You must have *JOBCTL special authority to use this command to add system-level environment variables.

Required Parameter

ENVVAR

Specifies the name of the environment variable to be added. If an environment variable by this name currently exists at the specified level (LEVEL parameter), error message CPF980 is issued, unless the REPLACE(*YES) option is used, in which case the variable is set to the new value specified.

ADDENVVAR limits environment variable name to a maximum of 128 bytes in length. The case is preserved when lowercase characters are specified. Valid values include all EBCDIC characters except the equal sign (=), the null-terminator (X'00') and blank (X'40'). The name must be enclosed in apostrophes if it contains any non-alphanumeric character. If an apostrophe is intended, two apostrophes must be used (").

Optional Parameters

VALUE

Specifies the environment variable value.

***NULL:** The value of the environment variable is the null character (X'00').

'environment-variable-value': Specify the value of the environment variable. ADDENVVAR limits value to a maximum of 1024 bytes in length. The case is preserved when lowercase characters are specified. Valid values include all EBCDIC characters. The value must be enclosed in apostrophes if it contains any non-alphanumeric character or blanks. If an apostrophe is intended, two apostrophes must be used (").

CCSID

Specifies the coded character set identifier (CCSID) of the text supplied on the ENVVAR and the VALUE parameters. This value is stored with the environment variable.

***JOB:** The CCSID of the text is assumed to be the CCSID of the job running this command.

***HEX:** The CCSID of 65535 is stored with this environment variable.

coded-character-set-identifier: Specify the CCSID to be stored with the environment variable. Valid values range from 1 through 65535.

LEVEL

Specifies the level of the environment variable.

***JOB:** This is a job-level environment variable.

***SYS:** This is a system-level environment variable.

REPLACE

Specifies whether the value of an existing environment variable should be reset to the new value.

***NO:** Do not replace. If an environment variable with the specified name (ENVVAR parameter) exists at the specified level (LEVEL parameter), error message CPFA980 is issued.

***YES:** If an environment variable with the specified name (ENVVAR parameter) exists at the specified level (LEVEL parameter), its value will be replaced by the new value.

Examples for ADDENVVAR

Example 1: Add an Environment Variable with CCSID 37

```
ADDENVVAR ENVVAR(altdir)
          VALUE('/mydir/dir2') CCSID(37)
```

This command adds the environment variable named altdir with the value /mydir/dir2 to the environment variables for the job. The value 37 is stored with the environment variable to indicate its CCSID.

Example 2: Set an Environment Variable to Null

```
ADDENVVAR ENVVAR(LIBPATH) VALUE(*NULL)
```

This command adds the environment variable named LIBPATH with the null (x'00) character value to the environment variables for the job.

Example 3: Add a System-level Environment Variable

```
ADDENVVAR ENVVAR(homedir)
          VALUE(/home) LEVEL(*SYS)
```

This command adds a system-level environment variable named homedir with value /home.

Example 4: Reset a Job-level Environment Variable

```
ADDENVVAR ENVVAR(altdir)
          VALUE('/mydir/dir3') REPLACE(*YES)
```

This command replaces the existing value of the variable altdir with the new value of /mydir/dir3.

Error messages for ADDENVVAR

*ESCAPE Messages

CPFA980

Environment variable name exists.

CPFA982

ENVVAR character not valid.

CPFA983

Unexpected error occurred.

CPFA984

Maximum number of environment variables exist.

CPFA98E

*JOBCTL special authority required to update system-level environment variables.

CPF3BCA

CCSID &1 not supported.

ADDEXITPGM (Add Exit Program) Command Description

ADDEXITPGM Command syntax diagram

Purpose

The Add Exit Program (ADDEXITPGM) adds an exit program entry for a specific exit point. Each exit point can have a single entry or multiple entries. The exit program number indicates the sequence in which the exit programs are run.

Required Parameters

EXITPNT

Specifies the exit point name to which the exit program is added. If no exit point by this name exists, and CRTEXITPNT(*YES) is specified, an exit point is created.

FORMAT

Specifies the exit point format name of the exit program that is added.

PGMNBR

Specifies the sequence in which the exit programs are run when multiple exit point programs for a specific exit point are defined.

***LOW:** The lowest available number for that specific exit point is assigned.

***HIGH:** The highest available number for that specific exit point is assigned.

program-number: Specify the exit program sequence number. Valid values range from 1 through 2147483647. Processing sequence is from the lowest number to the highest number. Exit program numbers do not need to be consecutive.

PGM Specifies the name of the exit program to be called. The program does not have to exist on the system when this command is run.

The name of the exit program can be qualified by one of the following library values:

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

program-name: Specify the name of the exit program.

Optional Parameters

TEXT Specifies the text that briefly describes the exit program. More information is in Commonly used parameters.

***BLANK:** Text is not specified.

***MSGID:** The description is taken from the message specified by the MSGID and MSGF parameters. The description is retrieved when exit program information is displayed using the WRKREGINF (Work with Registration Information) command or retrieved using the QusRetrieveExitInformation API.

'description': Specify no more than 50 characters of text, enclosed in apostrophes.

MSGID

Specifies the message identifier that contains the text that describes the exit program. The message is retrieved from the message file specified by the MSGF parameter. This parameter can only be specified if TEXT(*MSGID) is specified.

message-identifier: Specify the seven-character message identifier of the message that describes the exit program.

MSGF Specifies the message file and library that contains the message specified by the MSGID parameter. This parameter can only be specified if TEXT(*MSGID) is specified.

The name of the message file can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

library-name: Specify the name of the library where the message file is located.

message-file: Specify the name of the message file.

REPLACE

Specifies whether the attributes and data for an exit program entry are replaced. New values for the program and the text or message identifier can be specified. The program name and library cannot be changed.

***NO:** The attributes and data for an exit program entry are not replaced.

***YES:** The attributes and data for an exit program entry are replaced.

CRTEXTIPNT

Specifies whether the exit point is automatically created if it does not already exist.

***NO:** The exit point is not created. If the specified exit point does not exist, the exit program is not added and an error message is returned.

***YES:** The specified exit point is created.

THDSAFE

Specifies whether the exit program entry is threadsafe. This is intended for documentation purposes only. It may be used in determining the MLTTHDACN value, but there is no direct relation between the THDSAFE and MLTTHDACN keywords.

***UNKNOWN:** The threadsafe status of the exit program entry is not known.

***NO:** The exit program entry is not threadsafe.

***YES:** The exit program entry is threadsafe.

MLTTHDACN

Specifies the multithreaded job action when the exit program is called. It should be used by exit point providers when calling exit programs in a multithreaded job. The THDSAFE attribute of the exit program can be used in determining the multiple thread action, however, there is no direct relationship between the THDSAFE and MLTTHDACN keywords.

***SYSVAL:** Use the QMLTTHDACN system value to determine the action to take.

***RUN:** Run the exit program in a multiple threaded job.

***MSG:** Run the exit program in a multiple threaded job, but send an informational message. CPI3C80 can be used as the informational message.

***NORUN:** Do not run the exit program in a multiple threaded job. Depending on the exit point, do one of the following:

1. Send an escape message and do not call the exit program. CPF3C80 can be used as the escape message.
2. Send an informational message and do not call the exit program. CPI3C80 can be used as the informational message.
3. Call the exit program in a non-multithreaded job.

If you do use the THDSAFE value to determine the value for MLTTHDACN, please read the following recommendations:

1. If the THDSAFE value is *NO, MLTTHDACN should be set to *NORUN.
2. If the THDSAFE value is *UNKNOWN, MLTTHDACN should be set to *SYSVAL.
3. If the THDSAFE value is *YES, MLTTHDACN should be set to *RUN.

PGMDTA

Specifies the data that is passed to the exit program. This data should correspond to the input data defined by the exit point provider.

***NONE:** No data is passed to the exit program.

Element 1: CCSID for Program Data

***JOB:** The CCSID (coded character set identifier) of the current job is used.

CCSID-for-data: Specify the CCSID associated with the data passed to the exit program.

Element 2: Length of Program Data

***CALC:** The length is determined by the number of bytes specified for the third element of this parameter.

length-of-data: Specify the number of bytes of data passed to the exit program.

Element 3: Program Data

program-data: Specify the character or hexadecimal program data to be passed to the exit program. If you specify more program data than the length specified, the program data passed to the exit program is truncated. If you specify less program data than the length specified, the program data passed to the exit program is padded on the right with blanks. You can specify up to 2048 bytes of program data.

Example for ADDEXITPGM

```
ADDEXITPGM  EXITPNT(USER_EXIT_ONE)  FORMAT(EXIT1)
            PGMNBR(1)  PGM(LIB2/MYPGM)  TEXT(*MSGID)
            MSGID(TXT2345)  MSGF(LIB1/MYMSGF)
```

This command adds exit program MYPGM in library LIB2 to exit point USER_EXIT_ONE. This is first exit program run for the exit point. The text description for the exit program is retrieved from message TXT2345 in message file MYMSGF in library LIB1. No program data is passed to the exit program.

Error messages for ADDEXITPGM

None

ADDEWCBCDE (Add Extended Wireless Controller Bar Code Entry) Command Description

ADDEWCBCDE Command syntax diagram

Purpose

The Add Extended Wireless Controller Bar Code Entry (ADDEWCBCDE) command adds a set of bar code group parameters to an extended wireless controller source file member. The bar code group defines the parameters for scanning a particular bar code label. The Portable Transaction Computer (PTC) group specifies the bar code groups that are used to configure the bar code scanner.

Restrictions:

1. If the values specified for the INZFILE and INZMBR parameters of this command do not match the values specified for the corresponding parameters of the wireless controller description, extended wireless controller configuration data will not be downloaded to the wireless adapter.

Note:

You can use the Change Controller Description (Local Work Station) (CHGCTLLWS) command to view or change values specified for the INZFILE and INZMBR parameters in the wireless controller description.

2. You must have *IOSYSCFG special authority to use this command.

Required Parameters

BCDGRP

Specifies the bar code group name to be added. This name is used to identify configuration data related to a bar code group. If the bar code group name has been previously added, this command will fail. The bar code group name is a unique alphanumeric character string of as many as 16 characters in length.

INZMBR

Specifies the extended wireless controller source file member to add the bar code entry to. The bar code configuration data is added to this member.

Optional Parameters

INZFILE

Specifies the name of the source physical file that contains the extended wireless controller source file member. If the source physical file does not exist, this command will fail.

The name of the source file can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

QEWCSRC: The source file name QEWCSRC is used.

source-file-name: Specify the name of the physical file that contains the source member.

BCDTYPE

Specifies the bar code type defined by this bar code group.

***UPC:** UPC bar codes are used.

***EAN:** EAN bar codes are used.

***PLESSEY:** Plessey bar codes are used.

***ALPHAPLESSEY:** Alpha Plessey bar codes are used.

- ***ISBNPLESSEY**: ISBN Plessey bar codes are used.
- ***PUREPLESSEY**: Pure Plessey bar codes are used.
- ***SAINPLESSEY**: Sainsbury Plessey bar codes are used.
- ***UPCA**: UPC-A bar codes are used.
- ***UPCE**: UPC-E bar codes are used.
- ***EAN8**: EAN-8 bar codes are used.
- ***EAN13**: EAN-13 bar codes are used.
- ***CODABAR**: CODABAR bar codes are used.
- ***CODE3OF9**: CODE 3 of 9 bar codes are used.
- ***CODE2OF5**: CODE 2 of 5 bar codes are used.
- ***DISCR2OF5**: Discrete 2 of 5 bar codes are used.
- ***INTERL2OF5**: Interleaved 2 of 5 bar codes are used.
- ***INDUST2OF5**: Industrial 2 of 5 bar codes are used.
- ***CODE11**: CODE 11 bar codes are used.
- ***CODE128**: CODE 128 bar codes are used.
- ***CODE93**: CODE 93 bar codes are used.

LBLLEN

Specifies the maximum label length of a bar code label for the specified bar code group.

00: The label length is variable from 1 to 64.

label-length: Specify the maximum character length of a bar code label for this bar code group.

CHK1DIGIT

Specifies whether the check digit or the first check digit are checked on the bar code label. This is valid only when *PLESSEY, *CODE3OF9, *CODE11, or *CODE2OF5, are specified by the BCDDTYPE parameter.

***NO**: The check digit or the first check digit are ignored.

***YES**: The check digit or the first check digit are checked for a valid read.

CHK2DIGIT

Specifies whether the second check digit is checked on the bar code label. This is valid only when *PLESSEY or *CODE11 are specified by the BCDDTYPE parameter.

***NO**: The second check digit is ignored.

***YES**: The second check digit is checked for a valid read.

ALLZERO

Specifies whether a bar code label of all 0's is a valid scan. This is valid only when BCDDTYPE(*PLESSEY) is specified.

***NO**: A bar code label of all 0's is not a valid scan.

***YES**: A bar code label of all 0's is a valid scan.

ALPHADSP

Specifies whether to display bar code label characters : ; < = > and ? as the alphabetic characters A, B, C, D, E and F, respectively. This is valid only when BCDDTYPE(*PLESSEY) is specified.

***NO**: Display the characters normally.

***YES**: Display the characters as alphabetic characters.

ADDON2

Specifies whether a 2-digit add on is valid or ignored. This is valid only when *UPC or *EAN are specified by the BCDTYPE parameter.

***NO:** A 2-digit add on is ignored.

***YES:** A 2-digit add on is valid.

ADDON5

Specifies whether a 5-digit add on is valid or ignored. This is valid only when *UPC or *EAN are specified by the BCDTYPE parameter.

***NO:** A 5-digit add on is ignored.

***YES:** A 5-digit add on is valid.

SYS1UPCE

Specifies whether a System 1 UPC-E is valid or ignored. This is valid only when *UPC or *EAN are specified by the BCDTYPE parameter.

***NO:** A System 1 UPC-E is ignored.

***YES:** A System 1 UPC-E is valid.

SYS0UPCE

Specifies whether a System 0 UPC-E is valid or ignored. This is valid only when *UPC or *EAN are specified by the BCDTYPE parameter.

***NO:** A System 0 UPC-E is ignored.

***YES:** A System 0 UPC-E is valid.

UPCE Specifies whether a UPC-E should be expanded to UPC-A. This is valid only when *UPC or *EAN are specified by the BCDTYPE parameter.

***NO:** UPC-E bar codes are unaffected.

***YES:** UPC-E bar codes are expanded to UPC-A.

EAN13

Specifies whether UPC and EAN bar codes are expanded to EAN-13. This is valid only when *UPC or *EAN are specified by the BCDTYPE parameter.

***NO:** UPC and EAN bar codes are unaffected.

***YES:** UPC and EAN bar codes are expanded to EAN-13.

ADDON

Specifies the direction of add on digits. This is valid only when *UPC or *EAN are specified by the BCDTYPE parameter.

***BIDIRECTIONAL:** Add on digits are valid in both directions.

***FORWARD:** Add on digits are valid only in the forward direction.

EXT3OF9

Specifies whether the extended character set is allowed. This parameter is valid only when BCDTYPE(*CODE3OF9) is specified.

***YES:** The extended character set is allowed.

***NO:** The extended character set is not allowed.

DROPBEGIN

Specifies the number of characters to drop from the beginning of the bar code label. The valid range of values is from 0 through 64.

0: No characters are dropped from the beginning of the bar code label.

drop-begin: Specify the number of characters to drop from the beginning of the bar code label. The valid range of values is from 0 through 64.

DROPEND

Specifies the number of characters to drop from the end of the bar code label. The valid range of values is from 0 through 64.

0: No characters are dropped from the end of the bar code label.

drop-end: Specify the number of characters to drop from the end of the bar code label. The valid range of values is from 0 through 64.

TEXT Specifies text that briefly describes the PTC entry. More information on this parameter is in Commonly used parameters.

***BLANK**: Text is not specified.

'description': Specify no more than 50 characters of text, enclosed in apostrophes.

Example for ADDEWCBCDE

```
ADDEWCBCDE BCDGRP(BCD01) INZMBR(EWC01)
INZFILE(QGPL/QEWCSRC) BCDTYPE(*UPC)
```

This command adds a bar code group named BCD01 with a bar code type of *UPC in a extended wireless controller configuration source file member named EWC01 in source physical file QEWCSRC in QGPL.

Error messages for ADDEWCBCDE

None

ADDEWCM (Add Extended Wireless Controller Member) Command Description

ADDEWCM Command syntax diagram

Purpose

The Add Extended Wireless Controller Member (ADDEWCM) command adds a source file member with extended wireless controller parameters to the specified source file. This data is downloaded to the wireless controller when the controller is varied on. Specific Portable Transaction Computer (PTC) and bar code configurations are added to this member using the Add Extended Wireless Controller PTC Entry (ADDEWCPTCE) and Add Extended Wireless Controller Bar Code Entry (ADDEWCBCDE) commands.

Restrictions:

1. If the values specified for the INZFILE and INZMBR parameters of this command do not match the values specified for the corresponding parameters of the wireless controller description, extended wireless controller configuration data will not be downloaded to the wireless adapter.

Note:

You can use the Change Controller Description (Local Work Station) (CHGCTLLWS) command to view or change values specified for the INZFILE and INZMBR parameters in the wireless controller description.

2. You must have *IOSYSCFG special authority to use this command.

Required Parameter

INZMBR

Specifies the name of the source file member containing the extended wireless controller configuration data that is added to the source file.

Optional Parameters

INZFILE

Specifies the name of a source physical file to contain extended configuration source file member. If the source physical file does not exist, this command will fail.

The name of the source file can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

QEWC SRC: The source file name QEWC SRC is used.

source-file-name: Specify the name of an existing source physical file to which the member is added.

TXPADR

Specifies the local destination identifier of the controller. This is a 4-byte hexadecimal number with valid values ranging from 4001 through 4FFE.

4001: The value 4001 is used.

destination-ID: Specify the local destination identifier.

TXPPORT

Specifies the local transport port connection number of the controller. The valid range is from 0 through 15.

0: The value 0 is used.

transport-port: Specify the local transport port connection number.

TEXT Specifies the text that briefly describes the program and its function. More information is in Commonly used parameters.

***BLANK:** Text is not specified.

'description': Specify no more than 50 characters of text, enclosed in apostrophes.

Example for ADDEWCM

```
ADDEWCM INZMBR(EWC01) INZFILE(*LIBL/QEWC SRC)
```

This command adds a member named EWC01 in the default source physical file QEWC SRC in the library list with the default TXPADR of 4001 and default TXPPORT of 0.

Error messages for ADDEWCM

None

ADDEWCPTCE (Add Extended Wireless Controller PTC Entry) Command Description

ADDEWCPTCE Command syntax diagram

Purpose

The Add Extended Wireless Controller PTC Entry (ADDEWCPTCE) command adds a set of Portable Transaction Computer (PTC) group parameters to an extended wireless controller source file member. The PTC group parameters are the configurable PTC 5250 emulation operating parameters. These parameters are sent to each configured PTC at emulation startup. The Add Extended Wireless Controller Member (ADDEWCM) command must be run before this command to create the source file member.

Restrictions:

1. If the values specified on the INZFILE and INZMBR parameters of this command do not match the values specified on the corresponding parameters of the wireless controller description, extended wireless controller configuration data will not be downloaded to the wireless adapter.

Note:

You can use the Change Controller Description (Local Work Station) (CHGCTLLWS) command to view or to change values specified on the INZFILE and INZMBR parameters in the wireless controller description.

2. You must have *IOSYSCFG special authority to use this command.

Required Parameters

PTCGRP

Specifies the PTC group name to be added. This name is used to identify configuration data related to a group of PTCs bound by the PTCRANGE parameter. The PTC group name is a unique alphanumeric character string with a maximum of 16 characters.

INZMBR

Specifies the extended wireless controller source file member in which the PTC entry is added. The PTC configuration data is added to this member.

Optional Parameters

INZFILE

Specifies the name of the source physical file that contains the extended wireless controller source file member. If the source physical file does not exist, this command will fail.

The name of the source file can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

QEWC SRC: The source file name QEWC SRC is used.

source-file-name: Specify the name of the source physical file.

PTCRANGE

Specifies the beginning and ending PTC ID value to use for this PTC group. The defined configuration data is downloaded to any PTC detected within this range. A PTC group of one can be specified by setting the beginning and ending PTC ID to the same value.

Element 1: Beginning ID

0001: The value 0001 is used as the beginning PTC Id of the PTC group.

begin-id: Specify the beginning PTC Id of the PTC group. The valid values range from 0001 through 1022 (decimal).

Element 2: Ending ID

1022: The value 1022 is used as the ending PTC Id of the PTC group.

end-id: Specify the ending PTC Id of the PTC group. The valid values range from 0001 through 1022 (decimal).

INTENSITY

Specifies how the emulation screen on the PTC handles a field with the intensity attribute set.

***NORMAL:** The field is displayed as normal.

***INVERSE:** The field is displayed in reverse image.

STSLINE

Specifies whether the status line is displayed on the PTC.

***YES:** The status line is shown.

***NO:** The status line is not shown.

CSRTYPE

Specifies the type of cursor to be shown on the PTC.

***UNDERLINE:** The cursor is an underline cursor.

***BLOCK:** The cursor is a block cursor.

INACTTMR

Specifies, in minutes, the inactivity timer value for the PTC. The PTC will power down if no activity occurs on the PTC during this time period.

***DEV:** The default device inactivity timer value is used.

inactivity-timer: Specify the inactivity timer value in minutes. The valid range of values is from 0 through 9999.

BCKLTTMR

Specifies, in seconds, the backlight inactivity timer value for the PTC. The PTC turns the backlight off if no activity occurs on the PTC during this time period.

***DEV:** The default device backlight inactivity timer value is used.

backlight-timer: Specify the backlight inactivity timer value in seconds. The valid range of values is from 0 through 9999.

BCKLTKEY

Specifies whether the backlight turns on when a key is pressed on the PTC.

***YES:** The backlight turns on when a key is pressed.

***NO:** The backlight does not turn on when a key is pressed.

BYPASSEXIT

Specifies whether to bypass exit processing when leaving emulation on the PTC.

***NO:** Exit processing runs.

***YES:** Exit processing does not run.

AUTORUN

Specifies whether emulation software automatically runs on the PTC at system IPL.

***NO:** Emulation software does not automatically run at system IPL.

***YES:** Emulation software automatically runs at system IPL.

PRINTER

Specifies whether the printer for the PTC is the system printer or a printer that is locally attached to the PTC.

***SYSTEM:** The system printer is used.

***PTC:** The printer that is locally attached to the PTC is used.

WANDTYPE

Specifies the type of wand scanner being used.

***NONE:** No wand scanner is used.

***PENCIL:** A pencil wand scanner is used.

***LASER:** A laser wand scanner is used.

***RS232:** The wand scanner is attached on the RS-232 connector on the PTC.

PECKRATE

Specifies the wand pecking rate, in milliseconds. This value sets the time interval between the wand power on and power off states that is used to detect whether a label is present.

***DEV:** The default device wand pecking rate is used.

peck-rate: Specify the wand pecking rate in milliseconds. Valid values are 2, 4, 8, 16, 32, and 48.

LASERTMR

Specifies the laser read timer value, in milliseconds. If a good scan has not been performed before the given timer value, the laser is turned off.

***DEV:** The default device laser read timer value is used.

laser-read-timer: Specify the laser read timer value in milliseconds. Valid values are 1440, 2880, 4320, and 5760.

BCDFKEY

Specifies whether function keys can be entered by bar code labels.

***NO:** Function key entry by bar code is disabled.

***YES:** Function key entry by bar code is enabled.

AUTOENTER

Specifies whether the PTC Auto Enter function is on or off.

***ON:** The Auto Enter function is on.

***OFF:** The Auto Enter function is off.

CSRLOC

Specifies the new position of the cursor when the cursor is moved from one window chunk to another window chunk. A chunk is defined as being a portion of the 5250 emulation screen equal to the size of the PTC display.

***HOLD:** The cursor holds its position when moving from one window chunk to another.

***FIRST:** The cursor moves to the first active field when moving from one window chunk to another.

SHORTSCAN

Specifies whether a bar code label that does not completely fill an input field is processed as if it has filled that field.

***YES:** Bar code labels that do not fill an input field are processed as if they had filled the field.

***NO:** Bar code labels must fill the input field before they are processed.

SCANEOF

Specifies whether an erase end of field is done when a bar code label is shorter than the input field.

***YES:** An erase to end of field is done after a bar code scan.

***NO:** An erase to end of field is not done after a bar code scan.

POLL Specifies, in milliseconds, the fast poll interval for the radio module on the PTC.

***DEV:** The default device fast poll interval value is used.

poll-interval: Specify the fast poll interval in milliseconds. The valid range of values is from 0 through 9999.

POLLDLY

Specifies, in milliseconds, the fast poll delay parameter for the PTC radio module.

***DEV:** The default device fast poll delay value is used.

poll-delay: Specify the fast poll delay value in milliseconds. The valid range of values is from 0 through 9999.

POLLDECAY

Specifies the fast poll decay for the PTC radio module.

***DEV:** The default device fast poll decay value is used.

poll-decay: Specify the fast poll decay value. The valid range of values is from 0 through 255.

SLOWPOLL

Specifies, in milliseconds, the slow poll interval for radio module on the PTC.

***DEV:** The default device slow poll interval value is used.

slow-poll: Specify the slow poll interval in milliseconds. The valid range of values is from 0 through 99999.

DESTHOP

Specifies alternate controller destination id's and radio parameters to which the PTC attempts to connect if the primary one becomes unavailable. This is a four-element field and a maximum of 8 destination hops can be specified.

***NONE:** No destination hops are identified.

Element 1: Controller Destination ID

4001: The value 4001 is used.

destination id: Specify the 4-byte hexadecimal destination id of a wireless controller. The valid range of values is from 4001 through 4FFE.

Element 2: Frequency Channel Number

Specifies which center frequency to use on the direct sequence radio. Frequencies are referenced by channel number since the exact frequency value may change from country to country.

1: Frequency channel number 1 is used.

frequency-channel-number: Specify the frequency channel number. Valid values are from 1 to 5 and 901 to 911. The 1 to 5 range applies only to PTCs operating in the 2.4 Ghz range. The 901 to

911 range applies only to PTCs operating in the 900 Mhz range. If a value in the 901 to 911 range is used, the data rate may not be specified (900 Mhz data rates are fixed by the frequency channel number).

Element 3: Data Rate

Specifies the wireless LAN data rate.

***NONE:** No data rate is specified.

2M: A wireless LAN data rate of 2 megabits per second is used.

1M: A wireless LAN data rate of 1 megabit per second is used.

Element 4: System-ID

Specifies the 6-character hexadecimal radio system identifier to be used.

000002: The value 000002 is used.

system-ID: Specify a system ID to be used. Valid values range from 000002 to FFFFFE in hexadecimal format and the last digit must be even (for example, 0, 2, 4, 6, 8, A, C, E).

BCDGRP

Specifies the bar code group names used to define the bar code scanning capability of the PTC group. Bar code group names are defined and modified by the Add Extended Wireless Controller Bar Code Entry (ADDEWCBCDE) command or the Change Wireless Controller Bar Code Entry (CHGEWCBCDE) command.

***NONE:** No bar code group names are defined for this PTC group.

bar-code-group: Specify the bar code group name that corresponds to the bar code scanning capabilities required by the PTC. A maximum of 6 can be specified.

TEXT Specifies text that briefly describes the PTC entry.

***BLANK:** Text is not specified.

'description': Specify no more than 50 characters of text, enclosed in apostrophes.

Example for ADDEWCPTCE

```
ADDEWCPTCE PTCGRP(PTC01) INZMBR(EWC01)
PTCRANGE(0001 0020)
```

This command adds the configuration parameters for the PTC for a PTC group named PTC01 for PTCs with addresses from 1 to 20 to the extended wireless controller source file member EWC01 in source physical file QEWCSRC in the library list.

Error messages for ADDEWCPTCE

None

ADDEWLM (Add Extended Wireless Line Member) Command

Description

ADDEWLM Command syntax diagram

Purpose

The Add Extended Wireless Line Member (ADDEWLM) command adds a source file member that contains extended wireless line parameters to the specified source file. This data is downloaded to the wireless local area network (LAN) adapter when the line is varied on.

Restrictions:

1. If the values specified on the INZFILE and INZMBR parameters of this command do not match the values specified on the corresponding parameters of the wireless line description, extended wireless line configuration data will not be downloaded to the wireless adapter.

Note:

You can use the Change Line Description (Wireless) (CHGLINWLS) command to view or change values specified on the INZFILE and INZMBR parameters in the wireless line description.

2. You must have *IOSYSCFG special authority to use this command.

Required Parameter

INZMBR

Specifies the name of the new source file member that contains the extended wireless controller configuration data.

Optional Parameters

INZFILE

Specifies the name of the existing source physical file to contain the extended configuration source file member. If the source physical file does not exist, this command will fail.

The name of the source file can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

QEWSRC: The source file name QEWSRC is used.

source-file-name: Specify the name of an existing source physical file to add to the member.

ADPTCFG

Specifies the wireless LAN adapter configuration. The wireless LAN adapter has two communication methods: radio and wire backbone. The radio is a direct-sequence spread spectrum radio that can be used for wireless communications. The wire backbone is used to connect access points to a wireless LAN adapter. These access points also have a direct-sequence spread spectrum radio and are used to extend the radio coverage of the wireless network.

***ALL:** The wireless LAN adapter uses both radio and wire backbone communications.

***RADIO:** The wireless LAN adapter uses only radio communications.

***WIRED:** The wireless LAN adapter uses only wire backbone communications.

HOPID

Specifies the 12-character hexadecimal radio identifier on the wireless LAN adapter. This is an internal identifier that is used to determine the destination of a data packet, during its next hop on the network.

Note:

The value specified for the HOPID parameter is one of two different 12-character hexadecimal identifiers used by a wireless LAN adapter. The other is an endpoint identifier that is equivalent to the identifier used in Ethernet or token-ring networks.

***ADPT:** The preset wireless input/output adapter (IOA) address is used.

hop-ID: Specify the wireless LAN adapter hop address that overrides the preset address. The hop address must be an individual address (it cannot be a group address). Valid values range from 020000000000 to FFFFFFFF in hexadecimal format. The second digit from the left of the address must be a 2, 6, A, or E.

ROOT Specifies whether the radio of the wireless LAN adapter is a root cell.

A wireless network consists of a group of wireless access points that are interconnected in the form of a logical spanning tree. One of these wireless access points must be designated as the root cell for the network.

***YES:** The radio on the wireless LAN adapter is a root cell.

***NO:** The radio on the wireless LAN adapter is not a root cell.

FREQUENCY

Specifies which center frequency to use on the direct sequence radio. There are five center frequencies that are divided into two groups: A (channels 1,3 and 5) and B (channels 2 and 4). A center frequency from one of these groups will not overlap radio coverage with any other center frequency of that group; but, it will overlap with center frequencies of the other group. Frequencies are referenced by channel number since the exact frequency value may change based on channel set and country.

1: A channel number of 1 is used.

channel-number: Specify the frequency channel number to be used. Valid values are 1 to 5.

DATARATE

Specifies the wireless LAN data rate.

2M: A wireless LAN data rate of 2 megabits per second is used.

1M: A wireless LAN data rate of 1 megabit per second is used.

SYSID Specifies the 6-character hexadecimal radio system identifier to be used.

000002: The value 000002 is used.

system-ID: Specify a radio system identifier. Valid values range from 000002 to FFFFFE in hexadecimal format, but the last digit must be even (such as: 0, 2, 4, 6, 8, A, C, E).

TEXT Specifies the text that briefly describes the program and its function. More information on this parameter is in Commonly used parameters.

***BLANK:** Text is not specified.

'description': Specify no more than 50 characters of text, enclosed in apostrophes.

Example for ADDEWLM

```
ADDEWLM INZMBR(EWL01)
```

This command adds a member named EWL01 in the default source physical file QEWLSRC in the library list.

Error messages for ADDEWLM

None

ADDFNTTBLE (Add Font Table Entry) Command Description

ADDFNTTBLE syntax diagram

Purpose

The Add Font Table Entry (ADDFNTTBLE) command adds an entry in the specified font table. This command adds an entry in the user font mapping tables used by PSF/400 that controls:

1. Host resident to printer resident font character set mapping
2. Printer resident to host resident font character set mapping
3. Host resident to printer resident code page mapping
4. Printer resident to host resident code page mapping
5. Printer resident to printer resident font substitution mapping

To override the mapping of an existing entry in the system font and code page tables, you need to add a corresponding entry in the user tables.

In performing the printer to host and host to printer font mapping (first four tables above), the user tables are searched first for a match. If no match is found, then the System font or code page tables are searched.

For the printer resident to printer resident font substitution table, the following processing is done by the system:

- If the printer resident font specified in the print job is supported by the printer, then it is used. The printer resident to printer resident font substitution table is not searched.
- If the printer resident font specified in the print job is not supported by the printer, then the printer-resident to printer-resident font substitution table is searched.
 - If a matching entry is found in the printer resident font substitution table and the entry is supported by the printer, then the specified substitute font in the printer resident font substitution table is used.
 - If a matching entry is not found in the printer resident font substitution table or if the specified substitute font is not supported by the printer, then the system will use its internal font substitution tables to perform the font substitution.

Refer to Appendix D in the Printer Device Programming  book for more information on font mapping tables.

Restriction: The PSF/400 feature is required to use this command.

Required Parameter

FNTTBL

Specifies the name of the font table to be changed.

***PHFCS:** The printer resident to host resident font character set table is to be changed (add an entry).

This table would be used when your application (like Office Vision/400, DDS, etc) references printer resident fonts and the printer (3827, 3825, 3820, 3900 Model 1, etc) does not support resident fonts. PSF/400 must map the references from printer resident fonts to host resident fonts and download them.

***PHCP:** The printer resident to host resident code page mapping table is to be changed (add an entry).

This table is like the QPHFCS table, in that it is used when the application references printer resident code pages and the printer being used does not support printer resident code pages. The printer resident code page must be mapped to a host resident code page and down loaded to the printer by PSF/400.

***HPFCS:** The host resident to printer resident font character set table is to be changed (add an entry).

This table is used when your application references host resident fonts (font character sets and code pages) and the printer (4224, 4234, 4230, 64XX,etc) does not support down loading of host resident fonts. PSF/400 must map the references from host resident fonts to printer resident fonts.

***HPCP:** The host resident to printer resident code page mapping table is to be changed (add an entry).

This table is like the QPHFCS table, in that it is used when the application references host resident code pages and the printer being used does not support host resident code pages. The host resident code page must be mapped to a printer resident code page and down loaded to the printer by PSF/400.

The name of the font table must be specified when a printer resident to printer resident font substitution table is changed. This printer resident font substitution table should be used when all three of the following conditions exit.

- You are printing to a PSF/400 attached printer
- Your application specifies a printer resident font which is not supported by the printer you are using.
- You want to specify a different substitute printer resident font than the one selected by the system.

To use a printer resident to printer resident font substitution table with a particular PSF/400 printer, you need to specify the name of the font table on the FNTTBL parameter of the Create PSF Configuration (CRTPSFCFG) or Change PSF Configuration (CHGPSFCFG) command.

The name of the printer resident to printer resident font substitution table can be qualified by one of the following library values:

***CURLIB:** The current library is used to store the font table. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library where you want to store the font table.

font-table-name: Specify the name of the printer resident to printer resident font substitution table to be created.

Optional Parameters

PHFCS

Specifies the printer resident to host resident font character set mapping. The printer resident font, along with it's specified attributes will be mapped to a host resident font character set.

Element 1: From Font Identifier


from-font-identifier: Specify the printer resident font identifier to be mapped to a host resident font.

Element 2: Font Width

***PTSIZE:** The width for this font identifier will be calculated from the point size specified. When *PTSIZE is specified for width, the point size parameter can not be *NONE or *WIDTH. When mapping a typographic raster font (2304 - 3839, 4096 - 53247, 61440 - 65534), a point size value should be specified. The width value can be *PTSIZE or a value can be given.

***NONE:** No width is specified for this font identifier. *NONE should be specified when mapping to an outline font.

font-width: Specify a width for the font identifier. When mapping a fixed pitch raster font (1 - 750, 3840 - 4095), a width should be specified. The point size value can be *WIDTH or a value can be

given. See Appendix D in the Printer Device Programming  book for more information on font widths for printer resident fonts.

Element 3: Font Attributes

***NONE:** No special font attributes are specified on this font.

***BOLD:** The printer resident font is a bold font.

***ITALIC:** The printer resident font is an italic font.

***BOLDITC:** The printer resident font is a bold italic font.

***DBLWIDE:** The printer resident font is a double wide font.

***ITCDBLWIDE:** The printer resident font is an italic double wide font.

Element 4: Graphic Character Set

Specifies the graphic character set for the font identifier. This parameter allows you to map a font identifier to more than one host font character set (based upon graphic character set). The value specified is mapped to a superset graphic character set. See the Printer Device Programming



book for information on superset graphic character sets.

***SYSVAL:** The graphic character set specified in the system value QCHRID is used. A change to this system value will only take effect for the font mapping tables when the print writer is started. If QCHRID is changed and a printer is currently active, you must end the print writer and start it again.

graphic-character-set: Specify the graphic character set for the font identifier. The graphic character set is the first part of the graphic character identifier which consists of the graphic character set and code page. For example, for the graphic character identifier 697 500, 697 is the graphic character set and 500 is the code page. In this example, specify 697 for the graphic character set.

Element 5: Point Size

***WIDTH:** The font point size is computed from the font width value specified. When mapping a fixed pitch raster font (1 - 750, 3840 - 4095), it is recommended that a width value should be specified and the point size value should be *WIDTH.

***NONE:** No point size is specified for this font identifier. *NONE should be specified when mapping to an outline font.

point-size: Specify a point size ranging from 0.1 through 999.9. When mapping a typographic raster font (2304 - 3839, 4096 - 53247, 61440 - 65534), a point size value should be specified.

Element 6: To Font Character Set

to-character-set: Specify the font character set.

Element 7: Font Type

***RASTER:** The host resident font is a raster font.

***OUTLINE:** The host resident font is an outline font.

HPFCS

Specifies the host resident to printer resident font character set mapping. The host resident font, along with its specified attributes will be mapped to a printer resident font.

Element 1: From Host Font Character Set


from-font-character-set: Specify the font character set.

Element 2: Font Type

***RASTER:** The host resident font is a raster font.

***OUTLINE:** The host resident font is an outline font.

Element 3: To Font Identifier


***NONE:** To disable the mapping of a host resident to printer resident font, specify ***NONE** for the font identifier. See the Printer Device Programming  book for more information on disabling the mapping of host resident to printer resident fonts.

to-font-identifier: Specify the printer resident font identifier to be mapped from a host resident font.

Element 4: Font Width

***PTSIZE:** The width for this font identifier will be calculated from the point size specified. When ***PTSIZE** is specified for width, the point size parameter can not be ***NONE** or ***WIDTH**. When mapping a typographic raster font (2304 - 3839, 4096 - 53247, 61440 - 65534), a point size value should be specified. The width value can be ***PTSIZE** or a value can be given.

***NONE:** No width is specified for this font identifier. ***NONE** should be specified when mapping to an outline font.

font-width: Specify a width for the font identifier. When mapping a fixed pitch raster font (1 - 750, 3840 - 4095), a width should be specified. The point size value can be ***WIDTH** or a value can be given. See Appendix D in the Printer Device Programming  book for more information on font widths for printer resident fonts.

Element 5: Font Attributes

***NONE:** No special font attributes are specified on this font.

***BOLD:** The printer resident font is a bold font.

***ITALIC:** The printer resident font is an italic font.

***BOLDITC:** The printer resident font is a bold italic font.

***DBLWIDE:** The printer resident font is a double wide font.

***ITCDBLWIDE:** The printer resident font is an italic double wide font.

Element 6: Graphic Character Set

Specifies the graphic character set for the font identifier. This parameter allows you to map a font identifier to more than one host font character set (based upon graphic character set). The value specified is mapped to a superset graphic character set. See the Printer Device Programming



book for information on superset graphic character sets.

***SYSVAL:** The graphic character set specified in the system value QCHRID is used. A change to this system value will only take effect for the font mapping tables when the print writer is started. If QCHRID is changed and a printer is currently active, you must end the print writer and start it again.

graphic-character-set: Specify the graphic character set for the font identifier. The graphic character set is the first part of the graphic character identifier which consists of the graphic character set and code page. For example, for the graphic character identifier 697 500, 697 is the graphic character set and 500 is the code page. In this example, specify 697 for the graphic character set.

Element 7: Point Size

***WIDTH:** The font point size is computed from the font width value specified. When mapping a fixed pitch raster font (1 - 750, 3840 - 4095), it is recommended that a width value should be specified and the point size value should be *WIDTH.


***NONE:** No point size is specified for this font identifier. *NONE should be specified when mapping to an outline font.

point-size: Specify a point size ranging from 0.1 through 999.9. When mapping a typographic raster font (2304 - 3839, 4096 - 53247, 61440 - 65534), a point size value should be specified.

PPFCS

Specifies the printer resident font substitution mapping. When a printer resident font is not supported by a printer, you can specify the substitute printer resident font to be used instead of the substitute printer resident font selected by the system.

Element 1: From Font Identifier

from-font-identifier: Specify the printer resident font identifier for which a new substitution printer resident font is to be added. See Appendix D in the Printer Device Programming  book for more information on printer resident fonts that are supported, and which ones are scalable (require point size) and which ones are not scalable (specify point size *NONE).

Element 2: From Point Size

***NONE:** No font point size is specified. This should be specified for all non-scalable fonts.

***ALL:** Specifies that all point sizes for a scalable font will be mapped. If the font is not scalable, then this will be treated the same as *NONE.

from-point-size: Specify a point size ranging from 0.1 through 999.9.

Element 3: To Font Identifier

to-font-identifier: Specify the substitute printer resident font.

Element 4: To Point Size

***NONE:** No font point size is specified. This should be specified for all non-scalable fonts.

***ALL:** Specifies that all point sizes for a scalable font will be mapped. If the font is not scalable, then this will be treated the same as *NONE.

to-point-size: Specify a point size ranging from 0.1 through 999.9.

PHCP Specifies the printer resident to host resident code page mapping. The printer resident code page will be mapped to a host resident code page

Element 1: From Graphic Character Set

***SYSVAL:** The graphic character set specified in the system value QCHRID is used. A change to this system value will only take effect for the font mapping tables when the print writer is started. If QCHRID is changed and a printer is currently active, you must end the print writer and start it again.

from-graphic-character-set: Specify the graphic character set for the printer resident code page. The graphic character set is the first part of the graphic character identifier which consists of the graphic character set and code page. For example, for the graphic character identifier 697 500, 697 is the graphic character set and 500 is the code page. In this example, specify 697 for the graphic character set.

Element 2: From Code Page

from-code-page: Specify the printer resident code page value

Element 3: To Host Code Page

to-host-code-page: Specify the host resident code page value

HPCP Maps a host resident code page to a printer resident code page. The host resident code page will be mapped to a printer resident code page

Element 1: From Host Code Page

from-code-page: Specify the name of the host resident code page

Element 2: To Graphic Character Set

***SYSVAL:** The graphic character set specified in the system value QCHRID is used. A change to this system value will only take effect for the font mapping tables when the print writer is started. If QCHRID is changed and a printer is currently active, you must end the print writer and start it again.

to-graphic-character-set: Specify the graphic character set for the printer resident code page. The graphic character set is the first part of the graphic character identifier which consists of the graphic character set and code page. For example, for the graphic character identifier 697 500, 697 is the graphic character set and 500 is the code page. In this example, specify 697 for the graphic character set.

Element 3: To Code Page

to-code-page: Specify the printer resident code page value

Examples for ADDFNTTBLE

Example 1: Override Existing Font Entry in System Table

```
ADDFNTTBLE  FNTTBL(*PHFCS)
             PHFCS((254 84 *NONE 2039 7.0) (C0D0GT18 *RASTER))
```

This command adds an entry to the QPHFCS table (printer resident to host resident font character set table). To override the mapping of an existing entry in the System printer resident to host resident font character set table, you need to add a corresponding entry in the QPHFCS table. The following is the recommended steps to change the mapping of an entry in the system tables.

- Use the DSPFNTTBL command (DSPFNTTBL FNTTBL(*SYSPHFCS) OUTPUT(*PRINT)) to print the entries in the system font mapping table.
- Find the entry you want to change, and add an entry to the corresponding user font mapping table. In the above example, font identifier 254, width of 84, and point size 7.0 is to be added. to the user font table (QPHFCS). The width of 84 and point size of 7.0 is gotten from the system table. The entry has no special attributes (*NONE) and graphic character set 2039 is used.

In performing the font mapping, the attributes of the resident font specified in the print application are compared to those in the font table QPHFCS. If a match is found, then the specified host resident font (C0D0GT18) is down loaded to the printer. If no match is found, then the System printer resident to host resident font character set table is searched.

Note that the print application may specify the normal graphic character set (for example, 697 in 697 500 specified in QCHRID system value). The 697 is mapped to 2039 and will result in a match for this entry.

Example 2: Override Existing Font Symbol Entry in System Table

```
ADDFNTTBL FNTTBL(*PHFCS)
PHFCS((254 84 *NONE 1275 7.0) (C0SYMBOL *RASTER))
```

This command adds an entry to the QPHFCS table (printer resident to host resident font character set table) for use when using the special symbols code page (code page 259). As specified in Example 1, to override an existing entry in the System printer resident to host resident font character set table, you need to add a corresponding entry in the QPHFCS table.

Use the DSPFNTTBL command to display the system font mapping table and find the entry you want to change. In this example, you want to add an entry that mappings a printer resident to host resident font character set for the special symbol code page (259). As in the previous example, font identifier 254, width of 84, and point size 7.0 is to be added to the user font table (QPHFCS). The width of 84 and point size of 7.0 is gotten from the system table. The entry has no special attributes (*NONE) and graphic character set 1275 is used.

We now have two entries in the printer resident to host resident font character set table. Both entries have the same font identifier, width, and point size. The first entry will be used when the standard code page and graphic set is used by the application (697 500 in this example). The second entry will be used when a print application specifies special symbols (340 259).

Example 3: Add Font Entry that does not exist in System Table

```
ADDFNTTBL FNTTBL(*PHFCS)
PHFCS((65500 *PTSIZE *NONE *SYSVAL 7.0)
(C0NEWFNT *RASTER))
```

This command adds an entry to the QPHFCS table (printer resident to host resident font character set table) that does not exist in the system printer resident to host resident font character set table.

When adding entries that do not exist in the system printer resident to host resident font character set table, it is recommended that you specify a specific value for font width or point size, but not both. For fixed pitch fonts, you should specify a font width and *WIDTH for point size. For typographic fonts, you should specify a point size and *PTSIZE for font width. In this example, a typographic font of 65500 with point size 7.0 is added to the printer resident to host resident font character set table (QPHFCS).

Example 4: Override Existing Code Page Entry in System Table

```
ADDFNTTBL FNTTBL(*PHCP)
PHCP((*SYSVAL 38) (T1V00038))
```

This command adds an entry to the QPHCP table (printer resident to host resident code page table). To override an existing entry in the System printer resident to host resident code page table, you need to add a corresponding entry in the QPHCP table. The following is the recommended steps to change the mapping of an entry in the system tables.

- Use the DSPFNTTBL command (For example, DSPFNTTBL FNTTBL(*SYSPHCP) OUTPUT(*PRINT)) to print the entries in the system code page table.
- Find the entry you want to change, and add an entry into the corresponding user code page table. In the above example, code page 38 is to be added to the user code page table (QPHCP).

In performing the font mapping, the attributes of the resident code page specified in the print application are compared to those in the code page table (QPHCP). If a match is found, then the specified host resident code page (T1V00038) is down loaded to the printer. If no match is found, then the System printer resident to host resident code page table is searched.

Error messages for ADDFNTTBLE

None

ADDHDBDLFM (Add Host Database to DataLink File Manager)

Command Description

ADDHDBDLFM Command syntax diagram

Purpose

The Add Host Database to Datalink File Manager (ADDHDBDLFM) command adds a database name to the list of valid systems that can call the DataLink File Manager with a link request.

Restriction: You must have *IOSYSCFG special authority to use this command.

Required Parameters

HOSTDBLIB

Specifies the name of one or more host database libraries. A value must be specified for the HOSTDBLIB, HOSTDBINST, and HOSTDB parameters, or for the SRCFILE parameter.

host-database-library-name: Specifies the libraries (collections) on the host relational database system, that will contain files with DataLinks.

HOSTDBINST

Specifies the name of the database instance.

host-database-instance: Specifies the path or library in which the DB2 product was installed on the host database. If the host database is an iSeries 400 server, then the value QSYS must be used, since that is the library in which the DB2 product is installed.

HOSTDB

Specifies the name of the host database. If the host database is an iSeries 400 server, then the value specified must match the *LOCAL entry in that system's Relational Database Directory.

host-database-name: Specifies the name of the host relational database that will contain files with DataLinks.

SRCFILE

Specifies the name of a source file containing names of host databases to register. A value must be specified for the SRCFILE parameter or for the HOSTDBLIB, HOSTDBINST, and HOSTDB parameters.

The name of the database file can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

file-name: Specifies the name of a source file that contains the database names and libraries to register.

SRCMBR

Specifies the name of the source member in the specified file that contains the host database information. The source file must have a host database name, host database instance, and host database library on every row of the source member.

member-name: Specify the name of the file member that contains the host database names.

Example for ADDHDBDLFM

Registering a Datalink File Manager Host Database

```
ADDHDBDLFM
  HOSTDBLIB(MYLIB TESTLIB PERSONNEL)
  HOSTDBINST(QSYS)
  HOSTDB(RCHASXYZ)
```

This command registers RCHASXYZ as a valid system for calling the DataLink File Manager with a link request. Libraries MYLIB, TESTLIB, and PERSONNEL are libraries (or collections) on the host database system that can have database files containing DataLinks. QSYS is used as the host database instance, since the system is an iSeries 400 server.

Error messages for ADDHDBDLFM

*ESCAPE Messages

CPF3168

DataLink File Manager (DLFM) command failed.



ADDIMGCLGE (Add Image Catalog Entry) Command Description

ADDIMGCLGE Command syntax diagram

Purpose

The Add Image Catalog Entry (ADDIMGCLGE) command is used to create a virtual optical image in the target directory (as specified by the directory (DIR) parameter on the CRTIMGCLG command). If the optical image is added successfully, the image will be loaded and the image catalog (*IMGCLG) in library QUSRSYS will be updated. Optical images can be added from the following sources:

1. Physical CD/DVD media (by specifying an optical device containing the CD/DVD image to be added)
2. An optical image located in a directory on the system

If a file of that name already exists in the target directory, one of the following will occur:

1. If REPLACE(*NO) is specified, an error message will be issued and the catalog entry in the image catalog will not be replaced.
2. If REPLACE(*YES) is specified, the catalog entry in the image catalog will be replaced.
3. If REPLACE(*INSERT) is specified, the catalog entry in the image catalog will be inserted. If a catalog entry already exists at the index specified, the remaining entries will be incremented by one up to the next available index.

If the image does not exist in the target directory, the image is copied and the image catalog is updated with information about this image.

Restrictions:

1. You must have *SECADM and *ALLOBJ special authorities to use this command.

Required Parameters

IMGCLG

Specifies the name of the image catalog to which the image is to be added.

image-catalog-name: Specify the name of the image catalog to which the image is added.

FROMDEV

Specifies the name of the CD/DVD device from which the optical image is to be copied. A value must be specified for either the FROMDEV or FROMFILE parameter to identify the optical image to be copied.

device-name: Specify the name of the device from which the optical image is to be copied.

FROMFILE

Specifies the path name of the image file to be copied. A value must be specified for either the FROMFILE or FROMDEV parameter to identify the optical image to be copied to the target directory.

'file-name': Specify the path name of the optical image file to be added to the image catalog.

Optional Parameters

TOFILE

Specifies the name given to the file that will be copied to the target directory.

***GEN**: The file name will be generated from the source image. If the source image is from optical media, the name given to the file will be generated from the volume ID. If the source image is from another directory, the file name will be generated from the file name in the source directory.

***FROMFILE**: The file name will be the same name as the source image. If the source image is from optical media, the name given to the file will be the volume ID. If the source image is from another directory, the file will be given the same name as that in the source directory.

'file-name': Specify the file name of the optical image file to be added to the image catalog.

IMGCLGIDX

Specifies the image catalog index to be assigned to the image being added.

***AVAIL**: The image catalog index number assigned to the image will be the first sequential number available.

image-catalog-index-number: Specify an index number from 1 to 64.

REPLACE

Specifies the action to take if an catalog entry with the same index number as specified on the IMGCLGIDX parameter already exists in the image catalog.

The possible values are:

***NO**: Specify that the existing catalog entry will not be replaced and an error message will be issued.

***YES**: Specify that the existing image catalog entry will be replaced.

***INSERT**: Specify that the catalog entry specified will be added (or inserted if an existing index number already exists). If the image catalog entry is inserted, the remaining catalog entries will be incremented by one up to the next available index number.

TEXT Specifies the text that briefly describes the image being loaded.

***GEN**: If there is a description for the optical image file being added, the text field will be set to the description, otherwise it will be set to the volume ID of the CD.

'text-description': Specify up to 50 characters of text for this image file.

Examples for ADDIMGCLGE

Example 1: Adding an Image Catalog Entry from CD/DVD Media

```
ADDIMGCLGE IMGCLG(MYCLG) FROMDEV(OPT01)
```

This command adds the optical image in device OPT01 to the target directory and updates catalog MYCLG.

Example 2: Adding an Image Catalog Entry from a Directory

```
ADDIMGCLGE IMGCLG(MYCLG) FROMFILE('/MyDir/MyFile.img')
```

This command adds the optical image file /MyDir/Myfile.img to the target directory /MyNewDir and updates catalog MYCLG. The target directory is the directory specified on the DIR parameter of the CRTIMGCLG command.

Example 3: Adding an Image Catalog Entry with REPLACE(*INSERT)

```
ADDIMGCLGE IMGCLG(MYCLG) FROMDEV(OPT01) REPLACE(*INSERT)
```

This command adds the optical image in device OPT01 to the target directory and, if a catalog entry exists at the specified index, will insert the entry in the catalog and increment the remaining entries up to the next available index.

Error messages for ADDIMGCLGE

*ESCAPE Messages

CPFBC28

Catalog entry not added to image catalog &1.

CPFBC40

Not authorized to command &1.

CPFBC41

&1 command failed.



ADDICFDEVE (Add Intersystem Communications Function Program Device Entry) Command Description

ADDICFDEVE Command syntax diagram

Purpose

The Add Intersystem Communications Function Program Device Entry (ADDICFDEVE) command adds a program device entry with the specified name and attributes to an Intersystem Communications Function (ICF) file.

Required Parameters

FILE Specifies the qualified name of the ICF file to which the ICF program device entry is added.

The name of the ICF file can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

file-name: Specify the name of the file that contains the program device entry to be added.

PGMDEV

Specifies the name by which the ICF program device entry being added is known. The total number of program device entries that can be added (specified on the ADDICFDEVE or CHGICFDEVE command) to an ICF file is determined by the MAXPGMDEV parameter on the Create Intersystem Communications Function File (CRTICFF) command or the Change Intersystem Communications Function File (CHGICFF) command.

The name specified for this parameter is the ICF program device entry with which the user's program communicates. This name is used on device-specific input/output operations to identify the program device and its attributes. Although the user may specify the same remote location name on more than one device entry, each program device name must be unique among the entries for the ICF file. This allows the user to have more than one session to the same remote location, and/or to have different attribute values for each session to the same remote location.

Note:

Refer to the APPC, APPN, and HPR topic in the Information Center for information on how the system uses the RMTLOCNAME, DEV, LCLLOCNAME, and RMTNETID parameters to select an APPC device description.

RMTLOCNAME

Specifies the remote location name of the system with which this object communicates.

***REQUESTER:** The name used to refer to the communications device through which the program is started is used. The session that is assigned when the program device is acquired is the same session in which the program start request is received. If the program is not started as a result of a program start request, the acquire operation of the program device fails. The target program uses *REQUESTER as the remote location name in the intersystem communications function (ICF) file to connect to the session that the source program used to send the program start request.

The *REQUESTER value can be specified on only one program device entry and is valid only for a target communication job. If *REQUESTER is specified in any other type of job, a message is sent.

remote-location-name: Specify the full name of a remote location. The remote location does not need to exist at the time this command is run, but it must exist (be configured on the system as a device description or in the advanced peer-to-peer networking (APPN) function) for this remote location at the time the program acquires the program device. The same remote location name may be specified for many different program device entries. However, only one program device name associated with each asynchronous (ASYNCR), SNA upline facility (SNUF), or binary synchronous communication equivalence link (BSCCL) remote location may be added to the file at any one time. This value cannot be specified with CNVTYPE(*SRCPGM).

Optional Parameters

CMNTYPE

Specifies which type of communications parameters are shown on the prompt display. This parameter is used only for the purpose of prompting. The value specified for this parameter determines the subset of other parameters that are shown (prompted) for the user.

***ALL:** The parameters for all of the communications types appear in the prompt.

***APPC:** The advanced program-to-program communications (APPC) parameters appear in the prompt.

***ASYNC:** The asynchronous (ASYNC) parameters appear in the prompt.

***BSCSEL:** The binary synchronous communications equivalence link (BSCSEL) parameters appear in the prompt.

***FINANCE:** The finance parameters appear in the prompt.

***INTRA:** The intrasystem (INTRA) parameters appear in the prompt.


***RETAIL:** The retail parameters appear in the prompt.

***SNUF:** The SNA upline facility (SNUF) parameters appear in the prompt.

DEV Specifies the communications device used in the remote location. This parameter should only be specified for APPC, FINANCE, RETAIL, INTRA, and SNUF communications types.

***LOC:** The device associated with the remote location is used. If several devices are associated with the remote location, the system determines which device is used.

device-name: Specify the name of a communications device associated with the remote location. If the device name is not valid for the remote location, a message is sent when the program device

entry is acquired. More information on device names is in the SNA Distribution Services  book.

LCLLOCNAME

Specifies the local location name.

This parameter applies only to the APPC communications type and is ignored for all other communications types.

***LOC:** The device associated with the remote location is used. If several devices are associated with the remote location, the system determines which device is used.

***NETATR:** The LCLLOCNAME value specified in the system network attributes is used.

local-location-name: Specify the local location name to be associated with the program device entry. If the local location name is not valid for the remote location or remote location and device, an escape message is sent when the program device entry is acquired.

MODE Specifies the mode name used. This parameter applies only to the APPC communications type and is ignored for all other communications types.

***NETATR:** The mode name specified in the network attributes is used.

***BLANK:** The mode name consisting of 8 blank characters is used.

mode-name: Specify a mode name for the APPC communications device. If the specified mode is not valid for any combination of remote location, device, local location, and remote network ID, an escape message is sent when the program device entry is acquired.

RMTNETID

Specifies the remote network ID used with the remote location. This parameter applies to the APPC communications type only and is ignored for all other communications types.

***LOC:** The remote network identifier (ID) associated with the remote location is used. If several remote network IDs are associated with the remote location, the system determines which remote network ID is used.

***NETATR:** The RMTNETID value specified in the system network attributes is used.

***NONE:** No remote network identifier (ID) is used.

remote-network-ID: Specify a remote network ID for the APPC communications device.

FMTSLT

Specifies the record format selection used for input operations.

***PGM:** The program determines which record formats are selected. If an input (read) operation with a record format name is specified, that format is selected. If an input operation without a record format is specified, the default format (the first record format in the file) is selected. This also means that if there are any record identification (RECID) parameters specified in the data description specifications (DDS) for the file, or if any remote formats are received, they are not taken into consideration when the record is selected.

***RECID:** The record identification (RECID) keywords specified in the DDS for the file are used to do record selection. If there are no RECID keywords in the file, an error message is sent, the acquire operation of the program device ends, and the device is not acquired.

***RMTFMT:** The remote format names received from the sending system are used for record selection. If the device is not an APPC or INTRA device and *RMTFMT is specified when the program device entry is acquired, a run time error message is sent.

APPID

Specifies (in characters) the virtual telecommunications access method (VTAM*) identifier of the Customer Information Control System for Virtual Storage (CICS/VS) or Information Management System/Virtual Storage (IMS/VS) host subsystem sent with the sign-on message. This parameter applies to the SNUF and Finance communications type only and is ignored for all other communications types.

***DEVD:** The application identifier specified in the device description is sent with the sign-on message.

***USER:** The application program can send messages or a logon to the host. This is valid only when using the 3270 program interface.

application-ID: Specify the application identifier that is sent with the sign-on message.

BATCH

Specifies, for both CICS/VS and IMS/VS, whether this session is to be used for batch jobs. This parameter applies to the SNUF, RETAIL and INTRA communications types only and is ignored for all other communications types.

***NO:** Batch jobs do not occur.

***YES:** Batch jobs occur.

HOST Specifies the host or remote subsystem with which this session communicates. This parameter applies to the SNUF communications types only and is ignored for all other communications types.

***DEVD:** The host system specified in the device description is used.

***CICS:** The session communicates with CICS/VS.

***IMS:** The session communicates with IMS/VS.

***IMSRTR:** The session communicates with IMS/VS using the ready-to-receive option.

ENDSSNHOST

Specifies how SNUF ends the session with the host. This parameter is only valid for SNUF communications.

***RSHUTD:** SNUF sends a request-shut-down command to the host.

***TERMSELF:** SNUF sends a terminate-self command to the host. This value may be required if the value *RSHUTD fails to end a session with a non-IBM host.

SPCHOSTAPP

Specifies whether SNUF customizes support for special host applications outside the CICS* or IMS application layer.

***DEV**: The special host application specified in the device description is used.

***NONE**: SNUF does not customize support for special host applications.

***FLASH**: SNUF customizes support for the Federal Link Access for Secondary Half-sessions (*FLASH) protocol application.

INZSELF

Specifies whether a formatted INIT-SELF is built in place of the unformatted sign-on normally sent by SNUF to the host.

***NO**: The unformatted default sign-on provided by SNUF is used.

***YES**: The formatted INIT-SELF provided by SNUF is used.

HDRPROC

Specifies, for both CICS/VS and IMS/VS, whether the received function management headers are passed to the application program. This parameter applies to the SNUF communication type only and is ignored for all other communications types.

***SYS**: SNA upline facility (SNUF) removes function management headers before passing data to the program.

***USER**: Function management headers are passed with the data to the program.

MSGPTC

Specifies, for both CICS/VS and IMS/VS, whether message protection is used for this session. This parameter applies to the SNUF communications types only and is ignored for all other communications types.

***YES**: Message protection is used. SNUF saves messages until they are responded to and tries synchronization again if errors occur. *YES is only valid when BATCH(*NO) is also specified.

***NO**: Message protection is not used.

EMLDEV

Specifies that the program device entry is used to send and receive data streams to and from specific types of 3270 display or printer devices being emulated. This parameter consists of an emulation device type and an emulation device data format. The emulation device data format specifies the format of the type 3270 data stream being sent or received. A 20- or 32-byte common header that contains type 3270 command and data flow information is located at the start of the I/O buffer that is sending or receiving the type 3270 data stream. This parameter applies to the SNUF communications type only. This parameter can be specified as a list of two values (elements) or as a single value (*NONE).

***NONE**: This program device entry is not used for sending and receiving 3270 data streams.

Element 1: Type of Device

3278: The data stream is for a 3279, 3278 or 3277 display device.

3284: The data stream is for a 3284 Printer.

3286: The data stream is for a 3286 Printer.

3287: The data stream is for a 3287 Printer.

3288: The data stream is for a 3288 Printer.

3289: The data stream is for a 3289 Printer.

Element 2: Format of Data Stream

***UNFORMAT**: An unformatted 3270 data stream is sent or received. The user application program must translate the data stream into a display or printer image.

***FIELD:** A formatted 3270 data stream is sent or received. The formatted 3270 data stream contains a display or printer image followed by field definitions. The field definitions indicate the location and characteristics of each field.

***NOFIELD:** A formatted 3270 data stream that has no field definitions but contains a display or printer image is sent or received.

***EXTFIELD:** A formatted 3270 data stream contains extended field attribute information. The extended field attribute information is in the field definitions which follow the display image. The field definitions indicate the location and characteristics of each field. The value *EXTFIELD is valid only if the value 3278 is specified for the type of device on the EMLDEV parameter.

Note: If *FIELD, *NOFIELD, or *EXTFIELD is specified, BATCH(*NO) must also be specified.

CNVTYPE

Specifies the conversation type for which the application program is designed. This parameter applies to the APPC communication type only and is ignored for all other communications types.

More information on the APPC communications type can be found in the APPC, APPN, and HPR topic in the Information Center.

***SYS:** The advanced program-to-program communications (APPC) mapped conversation support is used.

***USER:** The advanced program-to-program communications (APPC) basic conversation support is used.

***SRCPGM:** The target program accepts the conversation type specified by the source program. If this value is specified, RMTLOCNAME(*REQUESTER) must also be specified.

BLOCK

Specifies that either the system or the user controls whether records are combined into blocks when they are sent. This parameter applies to the BSCEL communications type only and is ignored for all other communications types.

With this parameter, the user may specify one of the following conditions of record formatting:

- No blocking or deblocking: The record format described in the DDS is the format for both the record and the block.
- User blocking and/or deblocking: Gives the BSC controls needed to describe the record format of the system.
- System blocking with record separator characters: Specify the record separator character used by the system to determine record boundaries within the block.
- System blocking of fixed-length records: The system uses fixed-length records, and blocks and/or deblocks records accordingly.

If a parameter value other than *NONE or *USER is specified, records are blocked for output and are deblocked upon input as required by the system.

Element 1: Blocking Options

***DEV D:** The block option specified in the device description is used.

***NONE:** Blocking or deblocking is not done by the system.


***ITB:** The records are blocked or deblocked, based on the location of an intermediate text block (ITB) control character. For input files, a record is delimited by locating the next intermediate text block character. An end-of-text or end-of-transmission block character is used as an intermediate

text block character to delimit a block. For output files, an ITB character is added after the record. If it is the last character of the block, the ITB is replaced by an end-of-text or end-of-transmission block character.

***IRS:** The records are blocked or deblocked based on the location of an interrecord separator (IRS) character. For input files, a record is delimited by locating the next IRS character. For output files, an IRS character is added after the record.

***NOSEP:** No record separator character is contained in the block that is sent to or received from the device. The system blocks and deblocks the records by using a fixed-length record, as specified in the DDS format specifications.

***USER:** The program provides all the control characters (including record separator characters, binary synchronous communications (BSC) framing characters, and transparency characters) necessary to send records. More information about the device and binary synchronous communications equivalence link (BSC) support characteristics is in the BSC Equivalence Link

Programming  book.

***SEP:** The records are blocked or deblocked based on the location of a record separator character specified by the user. For input files, a record is delimited by locating the next record separator character. For output files, a record separator character is added after the record.

Element 2: Record Separator

X'1E': The record separator character X'1E' is used.

record-separator-character: Specify a record separator character that is unique and 1 byte in length. The record separator character may be specified as 2 hexadecimal characters, as in BLOCK(*SEP X'FD'), or the character may be specified as a single character by specifying a value ranging from 0 to 9 or A to F, as in BLOCK(*SEP A).

The following are dedicated BSC control characters that should not be used as record separator characters:

Table 1. Characters Unavailable for Record Separators (Used for BSC Control)

EBCDIC	ASCII	BSC Control
X'01'	X'01'	SOH (start-of-header)
X'02'	X'02'	STX (start-of-text)
X'03'	X'03'	ETX (end-of-text)
X'10'	X'10'	DLE (data-link escape)
X'1D'	X'1D'	IGS (interchange group separator)
X'1F'	X'1F'	ITB (intermediate text block)
X'26'	X'17'	ETB (end-of-transmission block)
X'2D'	X'05'	ENQ (enquiry)
X'32'	X'16'	SYN (synchronization)
X'37'	X'04'	EOT (end-of-transmission)
X'3D'	X'15'	NAK (negative acknowledgment)

RCDLEN

Specifies the maximum record length (in bytes) for data sent and received. This parameter applies to the BSC and the SNUF communications types only and is ignored for all other communications types.

***DEVD:** The record length specified in the device description is used. If a record is longer than the specified record length, a run time error occurs at the time the record is sent or received.

record-length: Specify the maximum record length (in bytes) to use with this device file. The value must be at least the size of the largest record sent. If a record is longer than the specified record length, a run time error occurs when the record is sent or received. Valid values range from 1 through 32767 bytes for SNUF communications. For BSCCEL communications, the maximum record length is 8192 bytes.

For BSCCEL communications, the maximum record length is 8192.

BLKLEN

Specifies the maximum block length (in bytes) for data sent. This parameter applies to the BSCCEL and the SNUF communications types and is ignored for all other communications types.

***DEV**: The block length specified in the device description is used.

block-length: Specify the maximum block length (in bytes) of records sent. The value must be at least the size of the largest record sent. Valid values range from 1 through 32767 for SNA upline facility (SNUF). For binary synchronous communications equivalence link (BSCCEL) communications, the maximum block length is 8192.

TRNSPY

Specifies whether data is sent in transparent text mode. Text transparency allows all 256 extended binary-coded decimal interchange code (EBCDIC) character codes to be sent; use this function when sending packed or binary data fields. This parameter applies to the BSCCEL communications type only and is ignored for all other communications types.

***DEV**: The text transparency option specified in the device description is used.

***NO**: Text transparency is not used.

***YES**: Text transparency is used, which allows all 256 EBCDIC character codes to be sent. ***YES** is valid only if **BLOCK(*NONE)**, **BLOCK(*NOSEP)**, or **BLOCK(*USER)** is specified.

Note:

Transparency of received data is determined by the data stream; therefore, this parameter is not relevant for received data. If **TRNSPY(*YES)** is specified with **BLOCK(*USER)**, BSCCEL ignores the transparency indicator during write operations. Correct controls must be given with the data to get transparent sending of data. For example, the data-link escape (DLE) and start-of-text (STX) control characters must first be specified; the system provides the remaining control characters required for transparent sending of data.

DTACPR

Specifies whether data compression is performed.

Note:

This parameter applies to the BSCCEL communications type only and is ignored for all other communications types.

***DEV**: The data compression option specified in the device description is used.

***NO**: No data compression or decompression occurs. **DTACPR(*YES)** cannot be specified if **TRNSPY(*YES)** is specified.

***YES**: Data is compressed for output and decompressed for input.

TRUNC

Specifies whether trailing blanks are removed from output records. This parameter is for the BSCEL communications type only and is ignored for all other communications types.

***DEV**: The truncation option specified in the device description is used.

***NO**: Trailing blanks are not removed from output records.

***YES**: Trailing blanks are removed from output records. TRUNC(*YES) cannot be specified if BLOCK(*NOSEP) or BLOCK(*ITB) is specified. If TRUNC(*YES) is specified and DTACPR(*YES) or BLOCK(*USER) is specified, truncation is ignored.

OVRFLWDTA

Specifies whether overflow data is discarded or retained.

***DISCARD**: Overflow data is not kept.

***RETAIN**: Overflow data is kept.

GRPSEP

Specifies a separator for groups of data (for example, data sets and documents). This parameter applies to the BSCEL communications types only and is ignored for all other communications types.

***DEV**: The group separator option specified in the device description is used.

***DEV3740**: A null record (STXETX) is used as a data group separator.

***EOT**: A BSC control character EOT (end-of-transmission) is used as a data group separator.

***OFCSYS**: A sent block that ends with the BSC control character ETX (end-of-text) is used as a data group separator.

RMTBSCEL

Specifies the type of BSCEL session with the remote system. This parameter applies to the BSCEL communications types only and is ignored for all other communications types.

***DEV**: The RMTBSCEL option specified in the device description is used.

***NO**: The remote system cannot recognize BSCEL commands or messages. In most cases, *NO is used when communicating with remote systems such as a 3741 Data Entry Station, an Office System 6, a 5230 Data Collection System, or a System/38.

***YES**: The remote system can recognize the BSCEL transaction starting commands, transaction ending commands, and online messages. In most cases, *YES indicates that the remote system is another iSeries 400 server, a System/38, a System/36, or a System/34 with BSCEL support.

INLCNN

Specifies the method used to make a connection on the line for the session being acquired. This parameter applies to the binary synchronous communications equivalence link (BSCEL) communications types only.

***CTLD**: The initial connection option specified in the controller description is used.

***ANS**: The remote system starts the call and the local system answers the call.

***DIAL**: The local system starts the call.

Examples for ADDICFDEVE

Example 1: Using RECID Keywords for Record Selection

```
ADDICFDEVE FILE(ICFFILE1)
  PGMDEV(BSCEL2) RMTLOCNAME(BSCNYC)
  FMSTLT(*RECID)
```


This command adds the program device entry named BSCSEL2 with a corresponding remote location named BSCNYC for the ICF file ICFFILE1. The program device is added with the attributes of FMSTLT(*RECID).

Example 2: Using Remote Format Names for Record Selection

```
ADDICFDEVE FILE(QGPL/ICFTEST)
  PGMDEV(APPC1)
  RMTLOCNAME(*REQUESTER)
  FMSTLT(*RMTFMT)  CNVTYPE(*SYS)
```

This command adds the program device entry named APPC1 with a remote location name of *REQUESTER for the ICF file ICFTEST in the QGPL library. This program device entry has the FMSTLT(*RMTFMT) and CNVTYPE(*SYS) attributes.

Example 3: Adding a Program Device Entry

```
ADDICFDEVE FILE(ICFLIB/TESTFILE)
  PGMDEV(JOE)  RMTLOCNAME(LU0MPLS)
```

This command adds the program device entry named JOE with remote location named LU0MPLS for the ICF file TESTFILE in library ICFLIB.

Example 4: Adding a Program Device Entry

```
ADDICFDEVE FILE(TESTFILE)
  PGMDEV(APPC)  RMTLOCNAME(APPCMPLS)
  DEV(MPLSLINE2)
```

This command adds the program device entry named APPC with a remote location name of APPCMPLS using device MPLSLINE2 to the ICF file TESTFILE.

Error messages for ADDICFDEVE

*ESCAPE Messages

CPF7365

Device not added to file &1 in &2.

ADDIPSIFC (Add IP over SNA Interface) Command Description

ADDIPSIFC Command syntax diagram

Purpose

The Add IP over SNA Interface (ADDIPSIFC) command is used to define AF_INET Sockets over SNA interfaces. An interface is an IP address by which this local host is known on the SNA transport. Interfaces defined by the ADDIPSIFC command are logical interfaces. They are not physical interfaces and they are not associated with any line description or network interface. There may be multiple AF_INET Sockets over SNA logical interfaces defined on a host.

Note:

When an interface is added it is activated by default if AF_INET Sockets over SNA communications is active and the user issuing the ADDIPSIFC command is authorized to start AF_INET Sockets over SNA interfaces. A user must have authority to the Start IP over SNA Interface (STRIPSIFC) CL command to have the authority to start AF_INET Sockets over SNA interfaces. If the interface should not be active, use the End IP over SNA Interface (ENDIPSIFC) CL command to deactivate it.

Only eight (8) AF-INET sockets over SNA interfaces can be active on a single host. If the maximum number of interfaces are already active, the interface being added will not be started. If you want the interface started, you must first end one or more active interfaces using the End IP over SNA interfaces (ENDIPSIFC) CL command and then use the Start IP over SNA interfaces (STRIPSIFC) CL command to start the interface.

Restriction: The user must have *IOSYSCFG authority to use this command.

Required Parameters**INTNETADR**

Specifies an internet address that the local system responds to on this interface. The internet address is specified in the form *nnn.nnn.nnn.nnn*, where *nnn* is a decimal number ranging from 0 through 255. An internet address is not valid if it has a value of all binary ones or all binary zeros for the network identifier (ID) portion or the host ID portion of the address. If the internet address is entered from a command line, the address must be enclosed in apostrophes.

Restrictions:

1. The internet address cannot begin with 0 (for example, 0.nnn.nnn.nnn).
2. The internet address cannot begin with 127 (for example, 127.nnn.nnn.nnn). This address range is reserved for TCP/IP loopback addresses.
3. The internet address cannot be a class D or class E address. Class D addresses range from 224.nnn.nnn.nnn to 239.nnn.nnn.nnn. Class E addresses range from 240.nnn.nnn.nnn to 255.nnn.nnn.nnn.
4. Each interface must have a unique internet address and cannot be the same as any defined TCP/IP interface internet address.

SUBNETMASK

Specifies the subnet mask, which is a bit mask that defines the part of the network where this interface attaches. The mask is a 32-bit combination that is logically ANDed with the internet address to determine a particular subnetwork. The bits of the mask set to the value one (1) determine the network and subnetwork portions of the address. The bits set to the value zero (0) determine the host portion of the address.

Note:

The network portion must be equal to one bits in the subnetmask. The host portion of an address must be at least two bits wide.

subnet-mask: Specify the mask for the network subnet field and host address field of the internet address that defines a subnetwork. The subnetwork mask is in the form, *nnn.nnn.nnn.nnn*, where *nnn* is a decimal number ranging from 0 through 255. The subnet mask must mask off all bits of the network class's network ID portion of the internet address. This means the subnet mask for a

class A address must be 255.nnn.nnn.nnn, the subnet mask for a class B address must be 255.255.nnn.nnn, and the subnet mask for a class C address must be 255.255.255.nnn. For example, 255.255.255.0 could define a subnet mask for an interface with a class B internet address. In this example, the first two octets must be 1 bits because these octets define the network ID portion of the class B internet address. The third octet of this subnet mask defines the actual subnet mask ID portion of the interface's internet address. It is also all 1 bits. This leaves the fourth octet to define the host ID portion of the interface's internet address.

Note: The bits that identify the subnetwork are not required to be adjacent in the address. However, it is strongly recommended that the subnet bits be contiguous and located in the most significant bits of the host address.

Note: If the subnet mask is entered from a command line, the address must be enclosed in apostrophes.

Restriction: The subnet mask cannot be 255.255.255.255 for a class A, class B, or class C interface internet address.

Example for ADDIPSIFC

```
ADDIPSIFC INTNETADR('9.5.1.248')
SUBNETMASK('255.255.255.0')
```

This command designates an IP address for this local host on the SNA transport which is 9.5.1.248. The SUBNETMASK indicates that network 9 is subnetted using bytes 2 and 3 of the internet address as the subnetwork.

Error messages for ADDIPSIFC

*ESCAPE Messages

CPFA108

IP over SNA interface added but not started.

TCP8050

*IOSYSCFG authority required to use &1.

TCP9999

Internal system error in program &1.

ADDIPSLOC (Add IP over SNA Location Entry) Command Description

ADDIPSLOC Command syntax diagram

Purpose

The Add IP over SNA Location Entry (ADDIPSLOC) command is used to define AF_INET sockets over SNA location mapping entries. AF_INET sockets over SNA requires that an SNA location (network identifier/location name) be defined for each IP address that can be reached on an SNA transport. The location mapping entries define the SNA location for each IP address.

The SNA locations can be identified in one of two ways:

- A single host (or IP address) is specified with a single SNA location (network identifier/location name).
- A group of hosts designated by the network ID portion of the IP address is specified with a given SNA network identifier and a location name template.

Restriction: The user must have *IOSYSCFG authority to use this command.

Required Parameters

RMTDEST

Specifies the remote network, subnetwork or host associated with this location entry. You must specify all four bytes that make up an internet address though some of the bytes may be equal to 0. For example, a remote route destination to all the hosts on the 9.5.11 subnetwork is identified by entering 9.5.11.0 for the remote route destination. Used in combination with a subnet mask value, the remote route destination will identify a remote network or system.

The remote route destination can be specified in the form *nnn.0.0.0* for class A, *nnn.nnn.0.0* for class B, and *nnn.nnn.nnn.0* for class C, or *nnn.nnn.nnn.nnn* for any combination thereof, where *nnn* is a decimal number ranging from 0 through 255. Any combination thereof means that you may specify remote route destination such as 9.5.0.0 to the hosts on the 9.5 subnet, even though all 9.5.x.x addresses are class A network addresses.

Restrictions:

1. The remote route destination cannot start with a zero (0); for example, 0.nnn.nnn.nnn.
2. The remote route destination cannot start with 127; for example, 127.nnn.nnn.nnn. This address range is reserved for TCP/IP loopback addresses.
3. The remote route destination cannot be a class D or class E address. Class D addresses range from 224.nnn.nnn.nnn to 239.nnn.nnn.nnn. Class E addresses range from 240.nnn.nnn.nnn to 255.nnn.nnn.nnn.
4. You cannot specify a remote route destination of 255.255.255.255. This is the limited broadcast address.
5. You cannot specify a directed broadcast address for the remote route destination; for example, nnn.255.255.255 for class A, nnn.nnn.255.255 for class B, and nnn.nnn.nnn.255 for class C.
6. For a single host remote route destination, all bits in the host portion of the IP address cannot be zero (0).

SUBNETMASK

Specifies a bit mask that identifies to AF_INET sockets over SNA which bits of the value specified for the route destination (RMTDEST) compose the network and subnet portions of the internet address. By defining the network portion and subnetwork portion of the RMTDEST address, the subnet mask also defines which bits of the RMTDEST address make up the host portion. The mask is a 32-bit combination that is logically ANDed with the internet address to determine a particular subnetwork. The bits of the mask set to the value one (1) determine the network and subnetwork portions of the address. The bits set to the value zero (0) determine the host portion of the address.

***HOST:** Specify this value when the internet address value specified in the remote route destination field is a host address. The subnet mask value is calculated to be 255.255.255.255.

subnet-mask: Specify the mask for the network subnet field and host address field of the internet address that defines a subnetwork. The subnetwork mask is in the form, *nnn.nnn.nnn.nnn*, where *nnn* is a decimal number ranging from 0 through 255. The subnet mask must mask off all bits of the network class's network ID portion of the internet address. This means the subnet mask for a class A address must be 255.nnn.nnn.nnn, the subnet mask for a class B address must be 255.255.nnn.nnn, and the subnet mask for a class C address must be 255.255.255.nnn. For example, 255.255.255.0 could define a subnet mask for an interface with a class B internet address. In this example, the first two octets must be 1 bits because these octets define the network ID portion of the class B internet address. The third octet of this subnet mask defines the actual subnet mask ID portion of the interface's internet address. It is also all 1 bits. This leaves the fourth octet to define the host ID portion of the interface's internet address.

For example, a remote route destination's internet address value of 129.35.192.0 identifies a Class B subnetwork. The network ID part of its address is 129.35. The portion of the subnetmask that is

associated with the network portion of a particular class of address must equal 255. Therefore, the upper 2 bytes must be equal to 255.255 in the subnetmask. The subnetmask in this example may be 255.255.192.0 if the third octet is used as the subnetwork ID portion of the internet address.

LOCTPL

Specifies the SNA location names associated with the IP network or subnetwork specified by the remote route destination or a single location name if the remote route destination address is for a single host.

location-name-template: Specify an 8 character template to be used by the system for generating remote location names based on the remote IP address specified on socket system calls. The first character must be A (or a) through Z (or z), or special characters \$, #, or @ followed by 0 through 9, A (or a) through Z (or z), \$, #, @, or ?. The template must specify some of the characters for the location name. The system generates the remaining characters based on the class of the IP address.

System-generated location name characters are identified by a question mark (?) character. Each question mark represents a single character that is generated by the system. A question mark may be used anywhere within the location name template except in the first character position. For example, the following location name templates are valid:

- ABCD????
- AB??CD??
- A?B?C?D?

Notes:

1. The number of question mark (?) characters in the template is dependent on how the remote destination (RMTDEST) parameter is used:
 - If the internet address specified for the RMTDEST is for a single host, the SUBNETMASK value must be *HOST or 255.255.255.255 and the LOCTPL value must not contain any question mark (?) characters.
 - If the internet address specified for the RMTDEST is for a network or subnetwork, the LOCTPL value must be an 8 character template containing a minimum number of question mark (?) characters based on the number of host mask bits contained in the SUBNETMASK parameter value. The minimum number of question mark characters is determined by dividing the number of host mask bits in the SUBNETMASK value by 5 and rounding up to the next whole number. For example: If the SUBNETMASK is 255.255.255.128, there are 7 host mask bits. In this case there must be 2 question mark characters in the template.
2. If the RMTDEST is for a group of hosts, a location name template must be specified. A single location name will not work.

location-name: Specify the remote location name. This name can be one to eight characters in length. The first character must be A (or a) through Z (or z), or special characters \$, #, or @ followed by 0 through 9, A (or a) through Z (or z), \$, #, or @.

If the RMTDEST is for a single host, a single location name must be specified. A location template will not work.

Optional Parameter

RMTNETID

Specifies the name of the remote SNA network associated with the IP network or IP address specified by the remote route destination.

***NETATR**: The remote network identifier specified in the network attributes is used.

remote-sna-network-identifier: Specify the remote network identifier. This identifier can be one to eight characters in length. The first character must be A (or a) through Z (or z), or special characters \$, #, or @ followed by 0 through 9, A (or a) through Z (or z), \$, #, or @.

Examples for ADDIPSLOC

Example 1: Adding an AF_INET Sockets over SNA Location Entry

```
ADDIPSLOC RMTDEST('128.2.0.0')
          SUBNETMASK('255.255.255.128') LOCTPL('ABCD????')
```

This command adds an AF_INET sockets over SNA location entry for a subnetwork with network 128.2 and subnet mask of 255.255.255.128. Remote IP addresses for subnetwork 128.2 specified on socket system calls are algorithmically mapped into SNA names that use the SNA network identifier specified in the network attributes and location names which start with ABCD. The system creates the remaining four characters of the location name based on the IP address. See the Convert IP over SNA Interface (CVTIPSIFC) command for more information.

Example 2: Adding an AF_INET Sockets over SNA Location Entry for a Host

```
ADDIPSLOC RMTDEST('128.2.3.4')
          SUBNETMASK(*HOST) LOCTPL(XYZ00001)
```

This command adds an AF_INET sockets over SNA location entry for a host at IP address 128.2.3.4. The address 128.2.3.4 is mapped to the SNA location name of XYZ00001 and uses the default SNA network identifier specified in the network attributes.

Error messages for ADDIPSLOC

None

ADDIPSRTE (Add IP over SNA Route) Command Description

ADDIPSRTE Command syntax diagram

Purpose

The Add IP over SNA Route (ADDIPSRTE) command is used to identify a route to a remote network or a route to a remote destination system in the AF_INET sockets over SNA configuration.

Restrictions:

1. The user must have *IOSYSCFG authority to use this command.
2. A route cannot be added unless the internet address of the gateway system specified by the NEXTHOP parameter can be reached directly through a network associated with a previously defined AF_INET sockets over SNA interface. An interface can be added using the Add IP over SNA Interface (ADDIPSIFC) CL command.

Required Parameters

RTEDEST

Specifies the route destination being added. You must specify all 4 bytes that make up an internet address though some of the bytes may be equal to 0. For example, a route to all the hosts on the 9.5.11 subnetwork is identified by entering 9.5.11.0 for the route destination. Used in combination with a subnet mask and next hop, the route destination uniquely identifies a route to a network or system.

route-destination: Specify the route destination being added. The route destination can be specified in the form *nnn.0.0.0*, for Class A, *nnn.nnn.0.0* for Class B, and *nnn.nnn.nnn.0* for Class C, or *nnn.nnn.nnn.nnn* for any combination thereof, where *nnn* is a decimal number ranging from 0 through 255.

Any combination thereof means that you may specify a route, such as 9.5.0.0 to the hosts on the 9.5 subnet, even though all 9.5.x.x addresses are class A network addresses.

Restrictions:

1. The route destination cannot start with a zero (0); for example, 0.nnn.nnn.nnn.
2. The route destination cannot start with 127; for example, 127.nnn.nnn.nnn. This address range is reserved for TCP/IP loopback addresses.
3. The route destination cannot be a class D or class E address. Class D addresses range from 224.nnn.nnn.nnn to 239.nnn.nnn.nnn. Class E addresses range from 240.nnn.nnn.nnn to 255.nnn.nnn.nnn.
4. You cannot specify a route destination of 255.255.255.255.
5. You cannot specify a directed broadcast address for the remote route destination; for example, nnn.255.255.255 for class A, nnn.nnn.255.255 for class B, and nnn.nnn.nnn.255 for class C.
6. For a single host route destination, all bits in the host portion of the IP address cannot be zero (0).
7. For a single host route destination, an interface cannot exist with the same internet address as the RTEDEST internet address.

SUBNETMASK

Specifies a bit mask that identifies to AF_INET sockets over SNA which bits of the value specified for the route destination (RTEDEST) compose the network and subnet portions of the internet address. By defining the network portion and subnetwork portion of the RTEDEST address, the subnet mask also defines which bits of the RTEDEST address make up the host portion. The mask is a 32-bit combination that is logically ANDed with the internet address to determine a particular subnetwork. The bits of the mask set to the value one (1) determine the network and subnetwork portions of the address. The bits set to the value zero (0) determine the host portion of the address.

***HOST:** Specify this value when the internet address value specified in the route destination field is a host address. The subnet mask value is calculated to be 255.255.255.255.

subnet-mask: Specify the mask for the network subnet field and host address field of the internet address that defines a subnetwork. The subnetwork mask is in the form, *nnn.nnn.nnn.nnn*, where *nnn* is a decimal number ranging from 0 through 255. The subnet mask must mask off all bits of the network class's network ID portion of the internet address. This means the subnet mask for a class A address must be 255.nnn.nnn.nnn, the subnet mask for a class B address must be 255.255.nnn.nnn, and the subnet mask for a class C address must be 255.255.255.nnn. For example, a destination route's internet address value of 129.35.192.0 identifies a Class B subnetwork. The network ID part of its address is 129.35. The portion of the subnet mask that is associated with the network portion of a particular class of address must equal 255. Therefore, the upper 2 bytes must be equal to 255.255 in the subnet mask. The subnet mask in this example may be 255.255.192.0 if the third octet is used as the subnetwork ID portion of the internet address.

NEXTHOP

Specifies the internet address of the next system (gateway) on the route. A route cannot be added unless the internet address specified by the NEXTHOP parameter can be reached directly through a network associated with a previously defined AF_INET sockets over SNA interface. An interface can be added by using the Add IP over SNA Interface (ADDIPSIFC) CL command.

internet-address: Specify the internet address. The internet address is specified in the form *nnn.nnn.nnn.nnn*, where *nnn* is a decimal number ranging from 0 through 255. An internet address is not valid if it has a value of all binary ones or all binary zeros for the network identifier (ID)

portion or the host ID portion of the address. If the internet address is entered from a command line, the address must be enclosed in apostrophes.

Restrictions:

1. The next hop internet address cannot begin with 0 or 127 (for example, 0.nnn.nnn.nnn).
2. The next hop internet address cannot be a class D or class E address. Class D addresses range from 224.nnn.nnn.nnn to 239.nnn.nnn.nnn. Class E addresses range from 240.nnn.nnn.nnn to 255.nnn.nnn.nnn.

Examples for ADDIPSRTE

Example 1: Adding a Network Route on the Network

```
ADDIPSRTE RTEDEST('128.2.0.0')
SUBNETMASK('255.255.0.0') NEXTHOP('9.2.3.4')
```

This command defines a network route for all remote hosts on the network 128.2. Network 128.2 is not subnetted since the first two octets of a class B internet address are the network ID portion of the address and the subnet mask is only masking off the first two octets. The gateway specified by NEXTHOP must be in the same network or subnetwork as one of the AF_INET sockets of SNA interfaces that has already been defined on the local host.

Example 2: Adding a Network Route on the Subnetwork

```
ADDIPSRTE RTEDEST('129.1.1.0')
SUBNETMASK('255.255.255.0')
NEXTHOP('128.3.4.5')
```

This command defines a network route for all remote hosts on the subnetwork 129.1.1. Network 129.1 is subnetted, with the subnet portion of the IP address contained in byte 3. For this example assume that subnet mask 129.1.1 is directly accessible only through the AF_INET Sockets over SNA interface 128.3.4.5 that has already been defined on the local host. Since the network id portion of interface 128.3.4.5 is not the same as the route destination's network id, we need to specify that the NEXTHOP is the local interface 128.3.4.5. This tells AF_INET Sockets over SNA to use local interface 128.3.4.5 to get to subnetwork 129.1.1.

Error messages for ADDIPSRTE

***ESCAPE Messages**

TCP2665

&2 &1 not added successfully.

TCP2666

&2 &1 not added.

TCP8050

*IOSYSCFG authority required to use &1.

TCP9999

Internal system error in program &1.

ADDJOBQE (Add Job Queue Entry) Command Description

ADDJOBQE Command syntax diagram

Purpose

The Add Job Queue Entry (ADDJOBQE) command adds a job queue entry to the specified subsystem description. A job queue entry identifies the job queue from which jobs are selected for running in the subsystem. Jobs can be placed on a job queue by spooling readers or by using the following commands:

- Submit Database Jobs (SBMDBJOB)
- Submit Diskette Jobs (SBMDKTJOB)
- Submit Job (SBMJOB)
- Transfer Job (TFRJOB)
- Transfer Batch Job (TFRBCHJOB)

In a subsystem, job queues with lower sequence numbers are processed first. For more information, refer to the description of the SEQNBR parameter.

Restriction: To use this command, the user must have object operational and object management authorities for the specified job queue. The specified job queue must already exist in the system if the library qualifier is not given. A job queue is created by the Create Job Queue (CRTJOBQ) command.

Required Parameters

SBSD Specifies the qualified name of the subsystem description to which the job queue entry is added.

The name of the subsystem description can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

subsystem-description-name: Specify the name of the subsystem description.

JOBQ Specifies the qualified name of the job queue that is a source of batch jobs that are started by the subsystem. If the job queue does not exist when the entry is added, a library qualifier must be specified because the job queue name is retained in the subsystem description.

The name of the job queue can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

job-queue-name: Specify the name of the job queue.

Optional Parameters

MAXACT

Specifies the maximum number of jobs that can be active at the same time from this job queue. More information on this parameter is in Commonly used parameters.

1: Only one job from the job queue can be active at any one time. However, the maximum activity level of the routing entries might prevent routing steps from being started. If *NOMAX is specified, all the jobs on the job queue are started (within the limit specified by the MAXJOBS parameter in the subsystem description), even though the activity level of the storage pool being used might prohibit them from running at the same time.

***NOMAX:** There is no maximum number of jobs that can be active at the same time.

maximum-active-jobs: Specify the maximum number of jobs from this job queue that can be active at the same time.

MAXPTYn

Specifies the number of jobs that can be started for a specified job priority level.

***NOMAX:** There is no disconnect limit.

0: No jobs are started from a specified priority level.

number-of-jobs: Specify the number of jobs started in a specified priority level. Valid values range from 1 through 99.

SEQNBR

Specifies a job queue sequence number, which is used by the subsystem to determine the order in which the job queues are processed.

10: A sequence number of 10 is assigned to this job queue.

sequence-number: Specify the sequence number assigned to the job queue. The sequence number must be unique in the subsystem description. Valid values range from 1 through 9999.

The subsystem first selects jobs from the job queue with the lowest sequence number. When all jobs on that queue have been processed or the number of jobs specified on the MAXACT parameter has been reached, the subsystem processes jobs on the queue with the next higher sequence number. This process continues until all job queue entries have been processed or until the subsystem has reached its limit for overall maximum jobs (as specified on the MAXJOBS parameter in the subsystem description). In some cases, the sequence is interrupted and the subsystem processes a queue with a lower sequence number. This occurs for this subsystem when one of the following conditions occurs:

- A held job or job queue is released.
- A job is placed on or transferred to a queue.
- A new queue is allocated.
- A job ends.

Examples for ADDJOBQE

Example 1: Adding a Job Queue

```
ADDJOBQE SBSD(QGPL/NIGHTSBS) JOBQ(QGPL/NIGHT)
MAXACT(3)
```

This command adds a job queue entry for the NIGHT job queue (in the QGPL library) to the NIGHTSBS subsystem description contained in the QGPL library. The entry specifies that up to three batch jobs from the NIGHT job queue can be active at the same time in the subsystem. The default sequence number of 10 is assumed.

Example 2: Running Jobs in Specific Priority Levels

```
ADDJOBQE SBSB(QBASE) JOBQ(JOBQ1) MAXPTY1(2)
MAXPTY7(0) MAXPTY8(0) MAXPTY9(0)
```

This command controls the selection of jobs for the job queue named JOBQ1 by setting priority levels to the values specified. This prevents any jobs with priority levels 7 through 9 from running.

Error messages for ADDJOBQE

*ESCAPE Messages

CPF1619

Subsystem description &1 in library &2 damaged.

CPF1691

Active subsystem description may or may not have changed.

CPF1697

Subsystem description &1 not changed.

ADDJOBSCDE (Add Job Schedule Entry) Command Description

ADDJOBSCDE Command syntax diagram

Purpose

The Add Job Schedule Entry (ADDJOBSCDE) command schedules batch jobs by adding an entry to the job schedule. You can use this command to schedule a batch job to be submitted once, or to schedule a batch job to be submitted at regular intervals.

The job schedule entry contains all the information needed to submit the job, including the command that the job runs, the job description and user profile under which the job is run, the job queue to which the job is submitted, and the message queue to which messages are sent.

The job is submitted to the specified job queue at the date and time specified on this command. This does not guarantee, however, that the job begins running at the scheduled time. The job does not begin running if the job queue is held or attached to an inactive subsystem, or if the maximum number of active jobs allowed to run in the subsystem or on the system at one time has been reached.

Each job schedule entry is identified by the job name, which is specified on the JOB parameter of this command, and an entry number, which is assigned by the system when the entry is added. The message replacement text for the message sent when an entry is added contains the entry number. If there is more than one entry with the same job name, you may need to specify the number when changing the entry using the Change Job Schedule Entry (CHGJOBSCDE) command, removing the entry using the Remove Job Schedule Entry (RMVJOBSCDE) command, or holding or releasing the entry using the Hold Job Schedule Entry (HLDJOBSCDE) or Release Job Schedule Entry (RLSJOBSCDE) command. You can use the Work with Job Schedule Entries (WRKJOBSCDE) command to show or print entries.

More information is in the Work Management  book.

Restrictions:

1. The user must have *USE authority to the job description and the user profile.
2. The user must have *USE and *ADD authorities to the message queue.
3. The user must have *READ authority to the job queue and *EXECUTE authority to all libraries associated with the specified objects.

Required Parameters

JOB Specifies the name of the job schedule entry.

***JOBID:** The job description specified on the JOBID parameter is used for the name of the job schedule entry.

job-name: Specify the name of the job schedule entry.

Note: To avoid deleting, holding, or releasing entries created by IBM products when you are using generic names to delete, hold, or release your entries, do not add entries with job names beginning with the letter Q.

CMD Specifies the command that runs in the submitted job. The IBM-supplied default routing program QCMD must be used when the job is started or the job will not run. Because the command you specify is used for the request data, the value specified on the RQSDTA parameter in the job description is ignored. The command you specify is syntax-checked when the entry is added.

You can specify a maximum of 512 characters.

FRQ Specifies how often the job is submitted to run.

***ONCE:** The job is submitted once.

***WEEKLY:** The job is submitted on the same day or days of each week at the scheduled time.

***MONTHLY:** The job is submitted on the same day or days of each month at the scheduled time.

If you specify *MONTHLY and a month does not contain the date specified on the SCDDATE parameter, the job is not run that month. For example, if SCDDATE(01/31/93) and FRQ(*MONTHLY) are specified, the job is submitted on 01/31, 03/31, 5/31, 7/31, 8/31, 10/31, and 12/31, but will not run in February, April, June, September, or November. To submit a job on the last day of every month, specify SCDDATE(*MONTHEND).

If you specify *MONTHLY and your system or your job is configured to use Julian date format, the job is submitted to run on the day of the month that it would run if the system or job did not use Julian date format.

Optional Parameters

SCDDATE

Specifies the date on which the job is submitted to run.

If your system or your job is configured to use the Julian date format, the *MONTHSTR and *MONTHEND values are calculated as if the system or job did not use the Julian date format.

***CURRENT:** The current date is used.

***MONTHSTR:** The job is submitted on the first day of the month. If you specify *MONTHSTR, and if today is the first day of the month, and if the time you specify on the SCDTIME parameter has not passed, the job is submitted today. Otherwise, the job is submitted on the first day of the next month.

***MONTHEND:** The job is submitted on the last day of the month. If you specify *MONTHEND, and if today is the last day of the month, and if the time you specify on the SCDTIME parameter has not passed, the job is submitted today. Otherwise, it is submitted on the last day of the next month.

***NONE:** No date is specified for the job to be submitted.

date: Specify the date in the job date format.

SCDDAY

Specifies the day of the week on which the job is submitted.

If today is the day of the week specified on this parameter and the time specified on the SCDTIME parameter has not passed, the job is submitted today. Otherwise, the job is submitted on the next occurrence of the specified day. For example, if SCDDAY(*FRI) and SCDTIME(12:00:00) are specified, and you are adding this job schedule entry at 11:00 a.m. on a Friday, the job is submitted today. If you are adding the entry at 4:00 p.m. on a Friday, or at 11 a.m. on a Monday, the job is submitted the following Friday.

***NONE:** No day is specified for the job to be submitted.

***ALL:** The job is submitted every day.

***MON:** The job is submitted on Monday.

***TUE:** The job is submitted on Tuesday.

***WED:** The job is submitted on Wednesday.

***THU:** The job is submitted on Thursday.

***FRI:** The job is submitted on Friday.

***SAT:** The job is submitted on Saturday.

***SUN:** The job is submitted on Sunday.

SCDTIME

Specifies the time on the scheduled date at which the job is submitted to run.

Note:

Although the time can be specified to the second, the activity involved in submitting a job and the load on the system may affect the exact time at which the job is submitted.

***CURRENT:** The job is submitted at the current time. If you specify SCDTIME(*CURRENT) and SCDDATE(*CURRENT), the job is immediately submitted to the specified job queue.

time: Specify the time. The time is specified in 24-hour format with or without a time separator as follows:

- With a time separator, specify a string of 5 or 8 digits, where the time separator for the job separates the hours, minutes, and seconds. If you issue this command from the command line, the string must be enclosed in apostrophes. If a time separator other than the separator specified for your job is used, this command fails.
- Without a time separator, specify a string of 4 or 6 digits (hhmm or hhmmss) where **hh** = hours, **mm** = minutes, and **ss** = seconds. Valid values for **hh** range from 00 through 23. Valid values for **mm** and **ss** range from 00 through 59.

RELDAYMON

Specifies the relative day of the month on which the job is submitted to run.

You can specify a value on this parameter only if the SCDDAY parameter and FRQ(*MONTHLY) are specified.

1: The job is submitted on the specified day of the week the first time it occurs in the month. For example, if you specify SCDDAY(*TUE), FRQ(*MONTHLY), and RELDAYMON(1), the job is submitted on the first Tuesday of every month.

2: The job is submitted on the specified day the second time it occurs in the month.

3: The job is submitted on the specified day the third time it occurs in the month.

4: The job is submitted on the specified day the fourth time it occurs in the month.

5: The format of this tape is FMT3570E. The data format is written on the tape volume with a 3570E device.

***LAST:** The job is submitted on the specified day the last time it occurs in the month.

OMITDATE

Specifies a list of dates on which the job is not submitted. You can, for example, use this parameter to prevent recurring jobs from running on holidays. The date must be specified in the job date format.

***NONE:** No dates are specified when a job is not submitted.

date: Specify a date when a job is not submitted.

RCYACN

Specifies the recovery action to be taken if the scheduled job cannot be submitted at the designated time because the system is powered down or in restricted state. The action specified on this parameter occurs at the next initial program load (IPL) or when the system comes out of restricted state.

Jobs submitted during IPL or when the system comes out of restricted state are submitted in the order that they would have been had they been submitted at the times specified in the job schedule entries. If multiple occurrences of a recurring job are missed, the job is submitted only once. The first missed occurrence of a recurring job is used to order the jobs. The next occurrence of the job is calculated from the current date.

Since the scheduler portion of IPL need not be complete for the IPL of the system to be complete, other jobs may start on the system before all of the scheduled jobs have been submitted.

This parameter does not apply in the following instances:

- When a job is released after being held at the date and time it was to be submitted
- When the date and time at which a job is to be submitted passes because of changes to date and time system values

***SBMRLS:** The job is submitted in the released (RLS) state.

***SBMHLD:** The job is submitted in the held (HLD) state.

***NOSBM:** The job is not submitted.

Specifying *NOSBM affects only missed occurrences of the job. If the job schedule entry is a recurring job, future occurrences are not affected.

SAVE Specifies whether the entry for a job that is submitted only once is kept after the job is submitted. This parameter is valid only if FRQ(*ONCE) is specified.

***NO:** The entry is not kept after the job is submitted.

***YES:** The entry is kept after the job is submitted. If you specify *YES, the job is submitted once. The job is not submitted again until the Change Job Schedule Entry (CHGJOBSCDE) command is used to specify a new date and time.

JOB Specifies the qualified name of the job description used when submitting the job.

***USRPRF:** The job description specified in the user profile under which the submitted job runs is used. The user profile is specified on the USER parameter.

The name of the job description can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

job-description-name: Specify the name of the job description.

JOBQ Specifies the qualified name of the job queue on which this job is placed.

You must have authority to the queue to specify a name on this parameter. Authority to the queue cannot be received through program adoption.

***JOBQ:** The scheduled job is placed on the job queue specified in the job description. The job description is specified on the JOBQ parameter.

The name of the job queue can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

job-queue-name: Specify the name of the job queue.

USER Specifies the name of the user profile under which the scheduled job is submitted.

***CURRENT:** The user profile that is currently running is used.

***JOBQ:** The user profile specified in the job description is used for the job schedule entry.

user-name: Specify the name of the user profile that is used. You must be authorized to the user profile. The user profile must be authorized to the job queue, job description, and message queue specified on this command.

MSGQ

Specifies the qualified name of the message queue to which messages are sent.

Messages are sent when the job is submitted and when a submitted job has completed running. Messages indicating a serious error are sent to the QSYSOPR message queue regardless of the value specified on this parameter when:

- The message queue specified on this parameter is damaged.
- MSGQ(*NONE) is specified.
- MSGQ(*USRPRF) and USER(*JOBQ) are specified, and the job description specified on the JOBQ parameter is changed to USER(*RQD) after the entry is added.

Note:

When MSGQ(*USRPRF) is specified and the user profile contains a message queue name with *LIBL specified for the library, the results can be unpredictable. When the job is submitted, the library list from the system value object is used.

***USRPRF:** The message queue specified in the user profile under which the submitted job runs is used. The user profile is specified on the USER parameter.

***NONE:** Completion messages are not sent. Error messages are sent to the QSYSOPR message queue.

The name of the message queue can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

message-queue-name: Specify the name of the message queue to which messages are sent.

TEXT Specifies the text that briefly describes the job schedule entry. More information is in Commonly used parameters.

***BLANK:** Text is not specified.

'description': Specify no more than 50 characters of text, enclosed in apostrophes.

Examples for ADDJOBSCDE

Example 1: Scheduling a Weekly Job

```
ADDJOBSCDE JOB(CLEANUP) SCDDATE(*NONE)
           CMD(CALL PGM(CLNUPLIB/CLNUPPPGM))
           SCDDAY(*FRI) SCDTIME('23:00:00')
           FRQ(*WEEKLY) RCYACN(*NOSBM)
           JOBD(CLNUPLIB/CLNUPJOB)
```

This command submits a job named CLEANUP every Friday at 11 p.m. The job uses job description CLNUPJOB in library CLNUPLIB. If the system is powered down or is in the restricted state at 11 p.m. on Friday, the job is not submitted at IPL or when the system comes out of restricted state.

Example 2: Scheduling a Monthly Job

```
ADDJOBSCDE JOB(PAYROLLJOB) CMD(CALL PAYROLL)
           SCDDATE(*NONE) SCDDAY(*MON)
           SCDTIME('09:00:00') FRQ(*MONTHLY)
           RELDAYMON(1)
```

This command submits a job to run program PAYROLL at 9 a.m. on the first Monday of every month.

Example 3: Omitting Dates

```
ADDJOBSCDE JOB(MONTHEND)
           CMD(CALL INVENTORY)
           SCDDATE(*MONTHEND) SCDTIME('23:30:00')
           FRQ(*MONTHLY) OMITDATE('12/31/93')
```

This command submits a job to run program INVENTORY at 11:30 p.m. on the last day of every month except December 31, 1993.

Example 4: Scheduling a Daily Job

```
ADDJOBSCDE JOB(*JOB) CMD(CALL DAILYCLEAN)
           SCDDATE(*NONE) SCDDAY(*ALL)
           SCDTIME('18:00:00') FRQ(*WEEKLY)
           RCYACN(*NOSBM) USER(SOMEPMER)
```


This command submits a job to run program DAILYCLEAN every day at 6 p.m. The job runs under user profile SOMEPMER. If the system is powered down or is in the restricted state at 6 p.m., the job is not submitted at IPL or when the system comes out of restricted state.

Example 5: Scheduling a Weekly Job

```
ADDJOBSCDE JOB(*JOB) CMD(CALL PGM1) SCDDATE('06/01/93')
FRQ(*WEEKLY) USER(PGMR1)
```

This command submits a job to run program PGM1 every week starting on June 1, 1993 at the current time. Because June 1 is a Saturday, the job is submitted every Saturday.

Example 6: Scheduling a Job to Run Twice a Month

```
ADDJOBSCDE JOB(*JOB) CMD(CALL PGM2)
SCDDATE(*NONE) SCDDAY(*MON *WED)
FRQ(*MONTHLY) RELDAYMON(3)
SCDTIME('23:30:00')
```

This command submits a job to run program PGM2 every third Monday and every third Wednesday at 11:30 p.m. The job is submitted this month if the third Monday and Wednesday have not passed when this entry is added. If, for example, yesterday was the third Monday, today is the third Tuesday, and tomorrow is the third Wednesday, the job is submitted tomorrow, and then not again until next month.

Example 7: Scheduling a Job to Run Twice a Month

```
ADDJOBSCDE JOB(*JOB) CMD(CALL PGM3)
SCDDATE(*NONE) SCDDAY(*MON)
FRQ(*MONTHLY) RELDAYMON(1 3)
SCDTIME('09:00:00') USER(PGMR3)
```

This command submits a job to run program PGM3 on the 1st and 3rd Monday of every month at 9:00 a.m. The job runs under user profile PGMR3.

Example 9: Scheduling a Job to Run Every Weekday

```
ADDJOBSCDE JOB(*JOB) CMD('CALL PGM4')
SCDDATE(*NONE)
SCDDAY(*MON *TUE *WED *THU *FRI)
SCDTIME('19:00:00') FRQ(*WEEKLY)
```

This command submits a job to run program PGM4 every weekday at 7 p.m.

Error messages for ADDJOBSCDE

*ESCAPE Messages

CPF1633

Job schedule entry &3 number &4 not added.

ADDJOBJS (Add Job using Job Scheduler) Command Description

Note: To use this command, you must have the 5722-JS1 (Job Scheduler for iSeries) licensed program installed.

ADDJOBJS Command syntax diagram

Purpose

The Add Job using Job Scheduler (ADDJOBJS) command allows you to schedule batch jobs by adding an entry to the job schedule.

Note: When referring to a job in this command, we are referring to an entry in Job Scheduler. An **entry** in Job Scheduler is a user-defined name for commands or programs that you want to process at scheduled times and dates. Job Scheduler jobs (entries) are not OS/400 objects.

Jobs can be a single job or a member of a group of jobs or an application. You can use this command to schedule a batch job to be submitted once, at a regular interval and so on, based on the schedule code you specify. You can schedule jobs with user-defined calendars, holiday exception calendars and fiscal year calendars.

The job schedule entry contains all of the information needed to submit the job, including the commands to process, the job description and user profile under which the job is run, the job queue to which the job is submitted, the message queue to which messages are sent and so on.

At the date and time you specify or Job Scheduler calculates, the job is submitted to the specified job queue. This command does not guarantee that the job will begin running at the scheduled time, however. The job will not begin running if the job queue is held or attached to an inactive subsystem, or if the maximum number of active jobs allowed to run in the subsystem or on the system at one time has been reached.

Each job schedule entry is identified by a user-defined job, which is specified on the JOB parameter of this command.

Restrictions:

1. The user must have use authority to the job description and the user profile.
2. The user must have use and add authorities to the message queue and the output queue.
3. The user must have read authority to the job queue and to all libraries associated with the specified objects.
4. The user must have use authority to the *ADDJOB function.

Required Parameters

JOB Specifies the name of the job schedule entry. You must specify a job and alternately can specify a group to which the job belongs as well as a sequence number for the job within the group.

The first job of a group must be sequence number 1. Subsequent sequence numbers should leave gaps (10, 20, 30 and so on) to allow job insertions if necessary. An entire group can be held or released by holding or releasing the first member of the group. Individual jobs within a group may also be held or released.

Element 1: Job

job-name: Specify the name of the job that you want to add.

Element 2: Group

***NONE:** There is not a group associated with this job.

group-name: Specify the name of the group to which this job is a member.

Element 3: Group sequence

***NONE:** There is not a sequence number assigned to the job.

group-sequence-number: Specify the sequence number of the job. Sequence numbers can range from 1 to 99.

Optional Parameters

APP Specifies the name of the application to which the job specified in the JOB parameter belongs.

***JOBCTL:** The application specified in the job controls is used as the application for this job.

***NONE:** This job does not belong to an application.

application-name: Specify the name of the application to which the job that you are adding belongs.

SCDCDE

Specifies the schedule code that you want to assign to the job.

***DAILY:** The job is scheduled to run on a daily basis or on selected days every week. For example, it may be scheduled to run on Wednesday only, or every day of the week.

***CALENDAR:** You are using a calendar to schedule the job. Calendars are user defined in the Work with Calendars display.

***DATE:** The job is scheduled to run on specific dates throughout the year. They are retained in the system in month day (mmdd) or day month (ddmm) format depending on the system value and will be submitted on the specified dates. Up to 13 dates can be scheduled.

***DAY:** The job is to be run on particular calendar days of the month, every month. These are entered in day (dd) format.

***NUMDAY:** The job runs every specified number of days. The number of days specified must be 99 or less.

Note: If the scheduled time to run is greater than the current time, Job Scheduler will count that time as a whole day (if you enter a Job Scheduler job on Monday at 3:00 p.m. to run every 2 days at 5:00 p.m., the job will run the first time on Tuesday at 5:00 p.m.).

***MINUTES:** The job runs every specified number of minutes.

***ONCE:** The job is to be run once. The SAVE parameter indicates whether the job is to be saved (*YES) or deleted (*NO) after it is run. If it is a job group, the SAVE parameter will not be shown and will assume saved (*YES). The date that the job is scheduled to be run is specified in the SNGDATE parameter. Leaving the date blank will run the job when the scheduled time is reached.

***MONTHEND:** The job runs on the last day of the month.

***FIRST:** The job is to run on the first designated day of every month or specified fiscal periods. This option is used in conjunction with the SNGDAY parameter. For instance, if *TUE is specified in the SNGDAY parameter, the job will be scheduled to run on the first Tuesday of each month.

***SECOND:** The job is to run on the second designated day of every month or specified fiscal periods. This option is used in conjunction with the SNGDAY parameter. For instance, if *TUE is specified in the SNGDAY parameter, the job will be scheduled to run on the second Tuesday of each month.

***THIRD:** The job is to run on the third designated day of every month or specified fiscal periods. This option is used in conjunction with the SNGDAY parameter. For instance, if *TUE is specified in the SNGDAY parameter, the job will be scheduled to run on the third Tuesday of each month.

***FOURTH:** The job is to run on the fourth designated day of every month or specified fiscal periods. This option is used in conjunction with the SNGDAY parameter. For instance, if *TUE is specified in the SNGDAY parameter, the job will be scheduled to run on the fourth Tuesday of each month.

***FIFTH:** The job is to run on the fifth designated day of every month or specified fiscal periods. This option is used in conjunction with the SNGDAY parameter. For instance, if *TUE is specified in the SNGDAY parameter, the job will be scheduled to run on the fifth Tuesday of each month. If there is not a fifth occurrence in a month, the job will not run and will be scheduled for the next time there is a fifth occurrence of a selected day in a month.

***LAST:** The job is to run on the last designated day of every month or specified fiscal periods. This option is used in conjunction with the SNGDAY parameter. For instance, if *TUE is specified in the SNGDAY parameter, the job will be scheduled to run on the last Tuesday of each month.

***FIRSTWRK:** The job is to run on the first working day of every month or specified fiscal periods. This option is used in conjunction with the **Working days** prompt in the system controls.

***LASTWRK:** The job is to run on the last working day of every month or specified fiscal periods. This option is used in conjunction with the **Working days** prompt in the system controls.

***DEPJOB:** The job is to run depending on the outcome or processing of another job.

***ALTERNATE:** The job is to run when a regular job ends abnormally and has this job defined as its alternate job.

***NONE:** The job is a subordinate job group and assumes the schedule code of the sequence number 1 job in the group.

***JOBCTL:** The job is scheduled to run based on the schedule code specified in the job controls.

TIME Specifies the time or times that you want the job to process on specified days. Times are entered in hour, minute (HHMM) format and can range from 0001 to 0024 (midnight).

You can enter multiple values for this parameter. If you are on an entry display and you need additional entry fields to enter these multiple values, type a plus sign (+) in the entry field opposite the phrase "+ for more", and press the Enter key.

scheduled-time: Specify the time or times that you want the job to process.

ITVMIN

Specifies the number of minutes that you want to use as an interval with the *MINUTES schedule code. Interval times can range from 1 to 720 minutes.

Note: The ITVMIN parameter is required when you use the *MINUTES schedule code.

number-of-minutes: Specify the number of minutes between submissions of the job.

CAL Specifies the name of the calendar that you want to use to schedule the job. A calendar is a user-defined set of days or dates used with the *CALENDAR schedule code. It can be used to specify Job Scheduler processing.

***JOBCTL:** The calendar that is used in this job is the calendar specified in the job controls.

***NONE:** This job does not use a calendar.

calendar-name: Specify the name of the calendar that you want to use for this job.

HDYCAL

Specifies the name of the holiday calendar that you want to use with a job. A holiday calendar is a user-defined set of exception days or dates used with the *CALENDAR schedule code. It can be used to determine whether a job will process.

***JOBCTL:** The holiday calendar that is used in this job is the holiday calendar specified in the job controls.

***NONE:** This job does not use a holiday calendar.

holiday-calendar-name: Specify the name of the holiday calendar that you want to use for this job.

FSCCAL

Specifies the name of the fiscal year calendar that you want to use with the job specified in the JOB parameter. Fiscal year calendars are made up of 12 or 13 periods with starting and ending dates for each period.

***JOBCTL:** The fiscal calendar that is used in this job is the fiscal calendar specified in the job controls.

***NONE:** There is not a fiscal calendar for this job.

fiscal-calendar-name: Specify the name of the user-defined fiscal year calendar that you want to use for this job.

DAY Specifies the days that you want this job to process. Days of the week are used with the *DAILY and *MINUTES schedule codes or job groups with a sequence greater than 1.

You can enter multiple values for this parameter. If you are on an entry display and you need additional entry fields to enter these multiple values, type a plus sign (+) in the entry field opposite the phrase "+ for more", and press the Enter key.

***ALL:** The job runs on all days of the week.

***MON:** The job is scheduled to run on Monday.

***TUE:** The job is scheduled to run on Tuesday.

***WED:** The job is scheduled to run on Wednesday.

***THU:** The job is scheduled to run on Thursday.

***FRI:** The job is scheduled to run on Friday.

***SAT:** The job is scheduled to run on Saturday.

***SUN:** The job is scheduled to run on Sunday.

SNGDAY

Specifies the day of the week that you want this job to process. Single day is used with the *FIRST, *SECOND, *THIRD, *FOURTH, *FIFTH AND *LAST schedule codes.

Note: *SUN is the default day of the week when *FIRST, *SECOND, *THIRD, *FOURTH, *FIFTH or *LAST is specified.

***SUN:** The job is scheduled to run on Sunday.

***MON:** The job is scheduled to run on Monday.

***TUE:** The job is scheduled to run on Tuesday.

***WED:** The job is scheduled to run on Wednesday.

***THU:** The job is scheduled to run on Thursday.

***FRI:** The job is scheduled to run on Friday.

***SAT:** The job is scheduled to run on Saturday.

ITVDAY

Specifies the number of days that you want to use as an interval with the *NUMDAY schedule code. Intervals can range from 1 to 99 days.

Note: The ITVDAY parameter is required when you use the *NUMDAY schedule code.

number-of-days: Specify the number of days between submissions of the job.

DATE Specifies the dates that a job is scheduled to be processed. Dates are entered in month, day (MMDD) format or (DDMM) format depending on the system value. The DATE parameter is required when the SCDCDE parameter contains *DATE.

You can enter multiple values for this parameter. If you are on an entry display and you need additional entry fields to enter these multiple values, type a plus sign (+) in the entry field opposite the phrase “+ for more”, and press the Enter key.

dates-of-the-year: Specify the date or dates that you want the job to process.

SNGDATE

Specifies the date that a job is scheduled to be processed. Dates are entered in month, day (MMDD) format or (DDMM) format depending on the system value. The SNGDATE parameter is used when the SCDCDE parameter prompt contains *ONCE.

***NEXT:** The job is to be processed when the scheduled time is reached.

single-date: Specify the date of the year that the job is to be submitted.

SAVE Specifies whether this job is to be saved after it has run. The SAVE parameter is used with the *ONCE schedule code.

***YES:** The job is to be saved after it has run.

***NO:** The job is to be deleted after it has run.

DAYMONTH

Specifies the days of the month that you want this job to process. Days of the month are used with the *DAY schedule code. Days of the month can range from 1 to 31.

You can enter multiple values for this parameter. If you are on an entry display and you need additional entry fields to enter these multiple values, type a plus sign (+) in the entry field opposite the phrase “+ for more”, and press the Enter key.

day-of-month: Specify the day or days of the month that you want this job to process.

FSCPERIOD

Specifies the periods within a fiscal year for the job. Periods are used when a fiscal calendar name is specified in the FSCCAL parameter. Periods can range from 1 to 13.

You can enter multiple values for this parameter. If you are on an entry display and you need additional entry fields to enter these multiple values, type a plus sign (+) in the entry field opposite the phrase “+ for more”, and press the Enter key.

***ALL:** All periods are to be included for the job.

fiscal-period-number: Specify the fiscal periods that you want to include for the job.

TEXT Specifies the text related to the job.

***NONE:** The job does not have any text description associated with it.

print-text: Specify the text description associated with the job.

CMD Specifies the command that you want to process in the job. You can specify a command, a call to

a program or pass parameters to a called program. Commands are checked for validity and parameters are validated against the Job Scheduler parameter file. Object existence is not checked on added or changed jobs.

CL-command: Specify the command that you want to process in the job.

RMTLOCNAME

Specifies the location and network identification of the remote location name on which to run the job.

Note: A value specified in the RMTLOCNAME parameter will be ignored when used with schedule code *ALTERNATE.

***JOBCTL:** Use the remote location name specified in the job controls.

***LCL:** Run the job on the local iSeries 400 server.

remote-location-name: Specify the name of the location associated with the system on which to run the job.

network-ID.location-name: Specify the network identifier and the name of the location associated with the system. Specify these values using the format nnnnnnnn.cccccc where nnnnnnnn is the network identifier and cccccc is the location name.

RANGE

Specifies the starting and ending time and date range for the job.

Note: If you are using the *MINUTES schedule code, the **Beginning time** value is used in conjunction with the **Ending time**. The *MINUTES schedule code is the only schedule code that uses these two values to indicate when a job starts and stops. The **Beginning date** and **Ending date** can be used with most schedule codes.

Note: A value specified in the RANGE parameter will be ignored when used with schedule codes *ALTERNATE, *DEPJOB and *GROUP (subordinate job group).

This parameter contains two lists of two elements each.

Element 1: Beginning time

***NONE:** No beginning time is specified for this job.

beginning-time: Specify the beginning time for the job in hour, minute (HHMM) format.

Element 2: Beginning date

***NONE:** No beginning date is specified for this job.

beginning-date: Specify the beginning date for this job in job date format.

Element 3: Ending time

***NONE:** No ending time is specified for this job.

ending-time: Specify the ending time for this job in hour, minute (HHMM) format.

Element 4: Ending date

***NONE:** No ending date is specified for this job.

ending-date: Specify the ending date for this job in job date format.

MAXRUN

Specifies the maximum run duration in minutes for the job.

Note: A value specified in the MAXRUN prompt will be ignored when used with schedule code *ALTERNATE.

***NOMAX:** There is no maximum duration for the job.

maximum-run-time: Specify the number of minutes that is the maximum duration for this job. After this number of minutes has passed, Job Scheduler will end the job whether it has completed or not. The maximum minutes can range from 1 to 9999 minutes.

PGRRCPNORM

Specifies the pager recipient who is to receive normal completion messages for the job that you are adding. This field is used in conjunction with command specified in the CHGPGRJS command.

Note: A paging product must be installed before this feature may be used.

You can specify the pager message that you want to send to the specified recipient when the job completes normally. The values that you specify for Pager recipient normal and Pager message are the substitution values used for the &RCP and &MSGTXT variables respectively in the CHGPGRJS command.

Element 1: Pager recipient normal

***JOBCTL:** You are using the pager recipient specified in the Job Scheduler job controls.

***NONE:** No pager recipient is assigned to receive messages when this job completes normally.

recipient-name: Specify the name of a recipient who is to receive messages from the job when it completes normally.

Element 2: Pager message

***JOBCTL:** The pager recipient is sent the pager message from the Job Scheduler job controls.

***COMP:** The job completion of the job is sent.

pager-message: Specify the pager message that you want to send to the pager recipient when this job completes normally.

PGRRCPABN

Specifies the pager recipient who is to receive abnormal completion messages for the job that you are submitting. This field is used in conjunction with the CHGPGRJS command.

Note: A paging product must be installed before this feature may be used.

You can specify the pager message that you want to send to the specified recipient when the job completes abnormally. The values that you specify for Pager recipient abnormal and Pager message are the substitution values used for the &RCP and &MSGTXT variables respectively in the CHGPGRJS command.

Element 1: Pager recipient abnormal

***JOBCTL:** You are using the pager recipient specified in the Job Scheduler job controls.

***NONE:** No pager recipient is assigned to receive messages when this job completes abnormally.

recipient-name: Specify the name of a recipient who is to receive messages from the job when it completes abnormally.

Element 2: Pager message

***JOBCTL:** The pager recipient is sent the pager message from the Job Scheduler job controls.

***COMP:** The job completion of the job is sent.

pager-message: Specify the pager message that you want to send to the pager recipient when this job completes abnormally.

ALTJOB

Specifies the name of the alternate job for the job. Alternate jobs only run when a regular job ends abnormally. Jobs are not required to have alternate jobs. You can also specify the group and sequence number for the alternate job that you specify.

***NONE**: The job does not have an alternate job.

Element 1: Alternate job

alternate-job-name: Specify the name of the alternate job that you want to use with the job that you are adding.

Element 2: Group

***NONE**: The alternate job does not have an alternate group.

group-name: Specify the name of the group associated with the alternate job for this job.

Element 3: Group sequence

***NONE**: The alternate job does not have a sequence number assigned.

group-sequence-number: Specify the sequence number assigned to the alternate job. Sequence numbers can range from 1 to 99.

RPTDSTID

Specifies the report distribution ID that is used to distribute the reports generated as a result of processing the job.

***NONE**: The job does not have a report distribution ID.

report-distribution-ID: Specify the report distribution ID that you want to associate with this job.

RCYACN

Specifies the recovery action to be taken if the job cannot be submitted at the designated time because the system is powered down or in restricted state. The action specified on the parameter occurs at the next IPL or when the Job Scheduler system becomes active.

Jobs submitted during IPL or when the system comes out of restricted state are submitted in the same order that they would have been had the jobs been submitted at the times specified in the job schedule entries. If multiple occurrences of a recurring job are missed, the job is submitted only once. The first missed occurrence of the job is calculated from the current date.

Since the scheduler portion of IPL need not be complete for the IPL of the system to be complete, other jobs may start on the system before all of the jobs have been submitted.

This parameter does not apply:

- When a job is released after being held at the date and time it was to be submitted.

Note: A value specified in the RCYACN prompt will be ignored when used with schedule code *ALTERNATE, *DEPJOB and *GROUP (subordinate job group).

***JOBCTL**: The job uses the recovery action specified in the Job Scheduler job controls.

***SBMRLS**: The job is submitted in release state (RLS).

***SBMHLD**: The job is submitted in the held state (HLD).

***NOSBM:** The job is not submitted.

Specifying *NOSBM affects only missed occurrences of the job. If the job schedule entry is a recurring job, future occurrences are not affected.

JOB Specifies the name of the job description used with this job.

***JOBCTL:** The job control job description is used for this job.

***USRPRF:** The job description in the user profile under which the submitted job runs is used as the job description of the submitted job.

job-description-name: Specify the name (library-name/job-description-name) of the job description used for the job.

***LIBL:** The library list is used to locate the job description name.

***CURLIB:** The current library for the job is used to locate the job description name. If no library is specified as the current library for the job, QGPL is used.

library-name: Specify the name of the library where the job description name is located.

JOBQ Specifies the name of the job queue in which this job is placed.

***JOBCTL:** The job control job queue is used for this job.

***JOBQ:** The submitted job is placed on the job queue named in the specified job description.

job-queue-name: Specify the name (library-name/job-queue-name) of the job queue on which the submitted job is placed.

***LIBL:** The library list is used to locate the job queue name.

***CURLIB:** The current library for the job is used to locate the job queue name. If no library is specified as the current library for the job, QGPL is used.

library-name: Specify the name of the library where the job queue name is located.

JOBPTY

Specifies the job queue scheduling priority. Valid values range from 1 through 9, where 1 is the highest priority and 9 is the lowest priority.

***JOBCTL:** The scheduling default specified in the Job Scheduler job controls is used for the job.

***JOBQ:** The scheduling priority specified in the job description is used for the job.

scheduling-priority: Specify a value, ranging from 1 through 9, for the scheduling priority for the job.

OUTPTY

Specifies the output queue priority for spooled output files that are produced by this job. Valid values range from 1 through 9, where 1 is the highest priority and 9 is the lowest priority.

***JOBCTL:** The output priority default specified in the Job Scheduler job controls is used for the job.

***JOBQ:** The output priority specified in the job description is used for the job.

output-priority: Specify a value, ranging from 1 through 9, for the output priority for the job.

PRTDEV

Specifies the qualified name of the default printer device for this job.

***JOBCTL:** The printer specified in the Job Scheduler job controls is used by the job as the printer device.

***USRPRF:** The printer device specified in the user profile where the submitted job runs is used as the printer device for this job. The printer device name is obtained from the profile when the job is submitted.

***SYSVAL:** The printer device specified in the system value, QPRTDEV, when this job is submitted is used.

***JOB:** The printer device specified in the job description is used for the submitted job.

printer-device-name: Specify the name of the printer device used for the submitted job.

OUTQ Specifies the qualified name of the output queue that is used for spooled output produced by the job. This parameter only applies to spooled printer files that specify *JOB for the output queue.

***JOBCTL:** The output queue specified in the Job Scheduler job controls is used as the job's output queue.

***JOB:** The output queue named in the job description used with the submitted job is the job's default output queue.

output-queue-name: Specify the name (library-name/output-queue-name) of the output queue that is used as the default output queue by the submitted job.

***LIBL:** The library list is used to locate the output queue name.

***CURLIB:** The current library for the job is used to locate the output queue name. If no library is specified as the current library, QGPL is used.

library-name: Specify the name of the library where the output queue name is located.

USER Specifies the name of the user profile for the job being submitted. If *RQD is specified in the job description, *JOB cannot be specified; a user name must be specified instead.

Note: The following IBM-supplied objects are not valid on this parameter:

- QDBSHR
- QDFTOWN
- QDOC
- QLPAUTO
- QLPINSTALL
- QRJE
- QSECOFR
- QSPL
- QSYS
- QTSTRQS

***JOBCTL:** The user profile specified in the Job Scheduler job controls is used for the job being submitted.

***CURRENT:** The same user profile used by the job that is currently running is used for the submitted job.

***JOBDB:** The user profile named in the specified job description is used for the job being submitted.

user-name: Specify the name of the user profile that is used for the job being submitted. You must be authorized to the user profile; the user profile must be authorized to the job description.

PRTTXT

Specifies up to 30 characters of text that is printed at the bottom of each page of printed output and on separator pages.

***JOBCTL:** The value in the Job Scheduler job control is used for this job.

***BLANK:** No text is printed.

***JOBDB:** The value in the job description is used for this job.

***SYSVAL:** The system value, QPRTTXT, is used for this job.

print-text: Specify the character string that is printed at the bottom of each page. A maximum of 30 characters can be entered, enclosed in apostrophes. The text on the listing will be centered in the same way it is entered.

RTGDTA

Specifies the routing data used to start the first routing step in the job. The routing data is used to determine the routing entry that identifies the program that the job runs.

***JOBCTL:** The value in the Job Scheduler job controls for routing data is used for this job.

***JOBDB:** The routing data specified in the job description is used to start the routing steps.

routing-data: Specify the character string that is used as routing data for the job. A maximum of 80 characters can be entered, enclosed in apostrophes if necessary.

CURLIB

Specifies the name of the current library associated with the job being run.

***JOBCTL:** The Job Scheduler job control is used for the submitted job.

***USRPRF:** The current library in the user profile where the submitted job runs is used as the current library for the submitted job.

***CRTDFT:** There is no current library for the submitted job. If objects are created in the current library, QGPL is used as the default current library.

current-library-name: Specify the name of a library used as the current library of the submitted job.

LIBL

Specifies the name of the library list that is used to search for any operating system object names that were specified without a library qualifier. If you want to select a library list from a list, place the cursor within the LIBL parameter and press F4.

***JOBCTL:** The Job Scheduler job control is used for the library list.

***JOBDB:** The library list in the job description used with this job is used as the initial user part of the library list for the job.

***SYSVAL:** The system default user library list is used by this job. It contains the library names that were specified in the system value, QUSRLIBL, at the time that the job is started.

***NONE:** The user portion of the initial library list for this job will be empty.

library-list-name: Specify the name of the library list that you want to use for this job.

LOG

Specifies the message logging values used to determine the amount and type of information sent

to the job log by this job. This parameter has three elements: the message (or logging) level, the message severity, and the level of message text. If no values are specified on this parameter, the values specified in the job description associated with this job are used.

Element 1: Level

***JOBCTL:** The value specified in the Job Scheduler job controls for logging is used for this job.

***JOBBD:** The value specified for message logging in the job description is used for this job.

message-level: Specify a value, ranging from 0 to 4, that specifies the message logging level used for this job's messages.

0: No data is logged.

1: The following information is logged: All messages sent to the job's external message queue with a severity level greater than or equal to the message severity specified (this includes the indications of job start, job end and job completion status).

2: The following information is logged:

- Logging level 1 information
- Requests or commands being logged from a CL program for which messages are issued with a severity code greater than or equal to the severity level specified.
- All messages associated with a request, or commands being logged from a CL program, that results in a high-level message with a severity level greater than or equal to the severity specified.

3: The following information is logged:

- Logging level 1 information
- All requests or commands being logged from a CL program.
- All messages associated with a request, or commands being logged from a CL program, that results in a high-level message with a severity level greater than or equal to the severity specified.

4: The following information is logged:

- All requests or commands being logged from a CL program and all messages with a severity code greater than or equal to the severity specified, including trace messages.

Note: A high-level message is one that is sent to the program message queue of the program that received the request or commands being logged from a CL program.

- Message severity

***JOBCTL:** The value specified in the Job Scheduler job controls for message severity is used for this job.

***JOBBD:** The value specified for message logging in the job description is used for this job.

message-severity: Specify a value, ranging from 00 to 99, that specifies the lowest severity level that causes an error message to be logged in the job's log.

- Message text

***JOBCTL:** The value specified in the Job Scheduler job controls for message text is used for this job.

***JOBBD:** The value specified for message logging in the job description is used for this job.

***MSG:** Only message text is written to the job's log or shown to the user.

***SECLVL:** Both the message text and message help of the error message is written to the job's log or shown to the user.

***NOLIST:** No job log is produced if the job competes normally. If the job ends abnormally (if the end of job code is 20 or higher), a job log is produced. The messages appearing in the job's log contain both message text and online help information.

LOGCLPGM

Specifies whether the commands that are run in a control language program are logged to the job log by way of the CL program's message queue. This parameter sets the status of the job's logging flag. If *JOB has been specified for the Message logging prompt (LOG parameter) in the Create CL Program (CRTCLPGM) command, the flag set in the LOGCLPGM parameter Log CL program commands prompt (LOGCLPGM parameter) is used. Other values for the Message logging prompt (LOG parameter) override the Log CL program commands prompt (LOGCLPGM parameter). The commands are logged in the same manner as the requests.

***JOBCTL:** The value in the Job Scheduler job controls is used for this job.

***JOBID:** The value specified in the job description is used.

***YES:** The commands in a CL program are logged to the job log.

***NO:** The commands in a CL program are not logged to the job log.

INQMSGRPY

Specifies the way that predefined messages that are sent as a result of running this job are answered. You can specify that no change is made in the way that predefined messages are answered, or that all inquiry messages require a reply, or that a default reply is issued, or that the system reply list is checked for a matching reply as each predefined inquiry message is sent.

***JOBCTL:** The value in the Job Scheduler job controls for inquiry message reply is used for this job.

***JOBID:** The inquiry message reply control specified in the job description used with this job is used.

***RQD:** A reply is required by the receiver of the inquiry message for all inquiry messages that occur when this command is run.

***DFT:** The default message reply is used to answer any inquiry messages that occur when this command is run.

***SYSRPLY:** The system reply list is checked to see if there is an entry for any inquiry message that is issued as a result of running this job that has a message identifier and any comparison data that match the inquiry message identifier and message data. If a match occurs, the reply value in that entry is used. If no entry exists for that message, a reply is required.

HOLD Specifies whether this job is held at the time that it is put on the job queue. A job placed on the job queue in the hold state is held until it is released by the Release Job (RLSJOB) command or ended, either by the End Job (ENDJOB) command or by the Clear Job Queue (CLRJOBQ) command. The current value for the parameter does not change.

***JOBCTL:** The value in the Job Scheduler job controls for hold on job queue is used for this job.

***JOBID:** The value specified in the job description determines whether the job is held when it is put on the job queue.

***YES:** The job is held when it is put on the job queue until it is released by a Release Job (RLSJOB) command or ended by an End Job (ENDJOB) command.

***NO:** The job is not held when it is put on the job queue.

SWS Specifies the first settings for a group of eight job switches used with this job. These switches can

be set or tested in a CL program and used to control the flow of the program. Only 0's (off) and 1's (on) can be specified in the 8-digit character string. The current value for the parameter does not change.

***JOBCTL:** The value in the Job Scheduler job controls for job switches is used for this job.

***JOB:** The value specified in the job description is the first setting for the job's switches.

switch-settings: Specify any combination of eight zeros and ones that is used as the first switch setting for the submitted job.

MSGQ

Specifies the name of the message queue to which a completion message is sent when the submitted job has completed running, either normally or abnormally. If an abnormal ending occurs, the help information for the completion message specifies the possible causes.

***JOBCTL:** The value in the Job Scheduler job controls for message queue is used for this job.

***USRPRF:** A completion message is sent to the message queue specified in the user profile of the submitter.

***NONE:** No completion message is sent.

message-queue-name: Specify the name (library-name/message-queue-name) of the message queue on which the completion message is to be sent.

***LIBL:** The library list is used to locate the message queue name.

***CURLIB:** The current library list is used to locate the message queue name.

library-name: Specify the name of the library where the message queue name is located.

ACGCDE

Specifies the accounting code that is used when logging system resource use for jobs that use this description.

***JOBCTL:** The accounting code for jobs using this description is obtained from the job controls.

***JOB:** The accounting code for jobs using this description is obtained from the job description.

***USRPRF:** The accounting code for jobs using this description is obtained from the user profile associated with the job.

***BLANK:** An accounting code of 15 blanks is assigned to jobs that use this description.

accounting-code: Specify the accounting code that you want to use for jobs using this description.

RUNPTY

Specifies the run priority for the job. Run priority is a value ranging from 1 (highest priority) through 99 (lowest priority), that represents the importance of the job when it competes with other jobs for machine resources. This value represents the relative (not absolute) importance of the job. If the job is rerouted, this value is reset according to the job's importance within the job class.

***JOBCTL:** The run priority is obtained from the job controls.

***NOCHG:** The run priority is not changed when job processing starts.

machine-running-priority: Specify the run priority, ranging from 1 through 99, that the job uses.

ADLCMD

Specifies additional commands that you want to process in this job. You can specify a command, a

call to a program or pass parameters to a called program. Commands are checked for validity and parameters are validated against the Job Scheduler parameter file. Object existence is not checked on added or changed jobs.

You can enter multiple values for this parameter. If you are on an entry display and you need additional entry fields to enter these multiple values, type a plus sign (+) in the entry field opposite the phrase “+ for more”, and press the Enter key.

additional-command: Specify additional commands for the job that you are adding.

Examples for ADDJOBJS

Example 1: Adding a Job

```
ADDJOBJS JOB(JOB01) TIME(1000)
```

This command adds a job to the job schedule. In this example, job JOB01 is being added to the job schedule and is scheduled to run at 10:00 a.m.. Note that the default schedule code for this job is *DAILY and the DAY parameter is *ALL.

Example 2: Adding a Job with a *CALENDAR schedule code

```
ADDJOBJS JOB(JOB02) SCDCDE(*CALENDAR)  
CAL(CAL) TIME(1100) CMD(WRKACTJOB)
```

This command adds the job JOB02 to the job scheduler. The job is scheduled to run at 11:00 a.m. using an *CALENDAR schedule and a calendar called CAL. When the job runs it processes the WRKACTJOB command.

Error messages for ADDJOBJS

None

ADDLIBLE (Add Library List Entry) Command Description

ADDLIBLE Command syntax diagram

Purpose

The Add Library List Entry (ADDLIBLE) command adds a library name to the user portion of the library list (after the current library list entry if it exists) for the process in which the command was entered. The user can specify whether the library is added to the beginning or the end of the library list. In addition to this, the user can specify whether the library is added before, after, or replaces an existing library in the library list.

Required Parameter

LIB Specifies the name of the library added to the user portion of the library list. Up to 250 libraries may exist in the user portion of the library list. Only one library name is added at a time with this command.

Optional Parameter

POSITION

Specifies the position in the user portion of the library list where the library is added.

***FIRST:** The library is inserted before the existing libraries in the user portion of the library list, after the current library, if it exists.

***LAST:** The library is added to the end of the user portion of the library list.

Element 1: Library Position

***AFTER:** The library specified on the LIB parameter is added to the user portion of the library list after the reference library specified on the POSITION parameter.

***BEFORE:** The library specified on the LIB parameter is added to the user portion of the library list before the reference library specified on the POSITION parameter.

***REPLACE:** The library is inserted into the library list in the position currently held by the reference library, and the reference library is removed from the list.

Element 2: Reference Library Name

reference-library-name: Specify the name of a library that already exists in the user portion of the library list. The library specified in the LIB parameter is added after this library, before this library, or replaces this library in the user portion of the library list.

Example for ADDLIBLE

```
ADDLIBLE LIB(TESTLIB) POSITION(*LAST)
```

This command adds the library TESTLIB to the end of the user portion of the library list.

Error messages for ADDLIBLE

***ESCAPE Messages**

CPF2103

Library &1 already exists in library list.

CPF2106

Library list not changed.

CPF2110

Library &1 not found.

CPF2113

Cannot allocate library &1.

CPF2118

Library &1 not added.

CPF2149

Library &1 was not found in the user library list.

CPF2176

Library &1 damaged.

CPF2182

Not authorized to library &1.



ADDLICCRQA (Add License CRQ Activity) Command Description

Note: To use this command, you must have the 5722-SM1 (System Manager for iSeries) licensed program installed.

ADDLICCRQA Command syntax diagram

Purpose

The Add License CRQ Activity command adds a license key distribution activity to a change request description that performs a license key distribution function.

The activity can be conditioned so that it will only run after one or more other activities have completed (successfully or unsuccessfully). The activity can also be scheduled to run at a date and time in the future.

Restrictions:

1. This command is shipped with public *EXCLUDE authority.
2. The user must have *CHANGE authority to the change request description object and *EXECUTE authority to the library.
3. The user adding the activity does not need to be authorized to any objects that are to be manipulated.
4. If a node list (NODL) value is specified, the node list can only contain entries that have a value of *SNA for the address type.
5. The license key information does not need to exist in the license repository when a license activity is being added to a change request, but it must exist by the time the activity runs.

Notes:

1. All conditions must be satisfied before the activity can be performed.
2. The start times indicate when the activity can be started. Actual start times may be later due to network and system delays.
3. All the existing records in the central site license repository containing the license key information for the specified product are sent to the specified managed system or systems. Only those ones that match the system serial number are added to the managed system license repository.

Required Parameters

CRQD Change request description object name.

The possible library values are:

***LIBL:** All of the libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is used to locate the object.

library-name: Only the library named in this parameter is searched.

change-request-description: The name of the change request description object.

ACTIVITY

The identifier of the activity to add to the change request description.

***GEN:** An activity identifier will be generated. The activity ID is in the form QACTxxxxxx where xxxxxx is a number that is incremented for each activity added.

***LAST:** The activity will be the last to run in the change request.

activity-name: A 10-character activity identifier.

ACTION

The license distribution function that is to be performed.

***SND:** Send the license keys of the specified product and license term to the specified managed systems.

PRDID

Specifies the 7-character identifier of the product for which the license key information will be distributed.

***ALL:** The license keys for all the products existing in the central site license repository will be distributed.

Product-Identifier: Specifies the 7-character identifier of the product for which the license key information will be distributed.

LICTRM

Specifies the license term for which the license keys information will be sent.

***ALL:** The license keys for all license term of the specified product existing in the focal point license repository will be distributed.

license-term: Specify the license term in one of the following formats:

Vx The authorized usage limit is valid for the entire version of the product or feature.

VxRy The authorized usage limit is valid for the entire release of the product or feature.

VxRyMz

The authorized usage limit is valid only for a modification of the product.

where “x” and “y” can be a number from 0 through 9, and “z” can be a number 0 through 9 or a letter A through Z.

NODL Specifies that the node list parameter is the object name that contains a list of systems that are the destinations for the activity. This parameter cannot be specified if the control point name (CPNAME) parameter is specified.

***NONE:** The systems on which this activity is to be performed are not identified by a node list. Individual control point names must be specified.

The possible library values are:

***LIBL:** All of the libraries in the user and system portions of the job’s library list are searched for the node list object.

***CURLIB:** The current library for the job is used to locate the node list object.

library-name: Only the library named in this keyword is searched.

node-list-name: Specify the node list object name containing the list of systems on which the activity is to be performed.

CPNAME

Specifies the APPN control point names of the managed systems on which this activity is to be performed. Control point names cannot be specified if the node list (NODL) parameter is specified.

***NONE:** The systems on which this activity is to be performed are not identified individually. A node list must be specified.

***NETATR:** The network ID of the local system will be used. This is useful when the node being specified is in the same network as the local system.

network-identifier: Specify the APPN network identifier of the managed system on which the activity is to be performed.

control-point-name: Specify the APPN control point name of the managed system on which the activity is to be performed.

COND Specifies which conditions must be met before this activity can be performed. Each condition identifies an activity that must run before this activity and the value the end code from that activity must have to allow this activity to run. The default condition is that the previous activity (in alphabetical order) must complete successfully before this activity can be run.

Element 1: Conditioning Activity

The activity that must run before this activity.

***PRV:** This activity is conditioned on the previous activity. Activities are ordered alphabetically by activity name. If the activity being added is the first activity, a previous activity does not exist and any condition with *PRV is marked as having been met.

conditioning-activity-name: Specify the name of the activity that must run before this activity. The activity name specified in the activity (ACTIVITY) parameter cannot be specified in the conditioning activity name. An activity cannot be conditioned on itself.

generic-conditioning-activity-name:* Specify the generic name of the activities that must run before this activity.

Element 2: Relational Operator

This element is the relational operator to use when comparing the end code from the conditioning activity.

***EQ:** Equal

***GT:** Greater than

***LT:** Less than

***NE:** Not equal

***GE:** Greater than or equal

***LE:** Less than or equal

Element 3: Condition Code

This element is the value compared to the actual end code of the conditioning activity.

***SUCCESS:** The activity ended successfully (0 <= end code <= 9). This end code can only be specified with relational operator *EQ or *NE.

***FAIL:** The activity failed (10 <= end code <= 89). This end code can only be specified with relational operator *EQ or *NE.

***NOTRUN:** The activity never started (90 <= end code <= 99). This end code is only specified with relational operator *EQ or *NE.

***ANY:** The activity ended with any end code. This end code is only specified with relational operator *EQ.

end-code: Specify an integer value (0-99) that indicates the result of an activity (success or failure). The end code ranges and descriptions are:

- 00** Activity completed successfully.
- 01-09** Activity completed with warning messages.
- 10-29** Activity did not complete successfully.
- 30-39** Activity was canceled by a user before it completed.
 - 30 = Activity ended with *CNTRLD option
 - 35 = Activity ended with *IMMED option
 - 39 = Activity ended with *FRCFAIL option
- 40-49** Activity was not run due to errors detected by the application.
 - 40 = Activity not run for security reasons
- 90-99** Activity was not run because conditions or schedules were not met.
 - 95 = Scheduled start time expired
 - 99 = Conditions cannot be met

Element 4: Condition Mode

This element indicates which systems the conditioning activity must have completed on before this activity can be performed.

***ALLNODES:** The conditioning activity specified must complete on all nodes before this activity can run.

***SAMENODE:** When the conditioning activity specified completes for a given node, the activity specified on the ACTIVITY parameter may run for that same node even though the conditioning activity specified may not have completed for all other nodes. In the case where this activity lists a node not in the conditioning activity, this activity may run for that node; the condition is ignored.

***NONE:** There are no conditions for this activity.

STRTIME

Specifies the date and time when this activity can be started on the central site system. This is the date and time at which the specified license information will be sent. The current date and time values and next date values are determined when the change request is submitted.

Element 1: Start After Time

***CURRENT:** This activity can start any time on or after the time when the change request is submitted.

start-after-time: Specify the time when this activity can start. The time can be entered as 4 or 6 digits (hhmm or hhmmss), where hh = hours, mm = minutes, and ss = seconds. Seconds are optional.

The time can be specified with or without a time separator such as a colon (:). Without a time separator, specify a string of 4 to 6 digits (hhmm or hhmmss) where hh = hours, mm = minutes, and ss = seconds. With a time separator, specify a string of 5 or 8 digits (hh:mm or hh:mm:ss) where the time separator specified for your job is used to separate the hours, minutes, and seconds.

Element 2: Start After Date

***CURRENT**: This activity can start on or after the date on which the change request is submitted.

***NEXT**: The activity can start on any date after the date the change request is submitted.

start-after-date: Specify the date after this activity can start. The date must be specified in the job date format.

Element 3: Start Before Time

This element is ignored if the start before date is *ANY.

***ANY**: The activity can start at any time on or before the start before date.

***CURRENT**: The activity must start before the time at which the change request was submitted on the date specified on the start before date element.

start-before-time: Specify the time before which the activity must start. If the activity cannot be started before this time, it never starts. The time can be entered as 4 or 6 digits (hhmm or hhmmss) where hh = hours, mm = minutes, and ss = seconds. Seconds are optional. The time can be specified with or without a time separator such as a colon (:). With a time separator, specify a string of 5 or 8 digits (hh:mm or hh:mm:ss).

Element 4: Start Before Date

***ANY**: The activity can start at any time after the start after time and the start after date.

***CURRENT**: The activity must start on the date the change request is submitted.

***NEXT**: The activity must start by the day after the date the change request is submitted.

start-before-date: Specify the date before the activity must start. If the activity cannot be started by this date, it never starts. The date must be specified in the job date format.

HOLD Specifies that the activity be held when the change request is submitted.

***NO**: The activity is not held. It runs when all conditions are met at the start time.

***YES**: The activity is held for all nodes when the change request is submitted. The change request must be released by you before it will run.

TEXT Specifies the activity description.

***GEN**: A text description is generated based upon the action chosen.

text-description: Specify a 50-character description of the activity.

Examples for ADDLICCRQA

Example 1: Adding an Activity to Send Licenses

```
ADDLICCRQA CRQD(MYLIB/CR1)
ACTIVITY(ACT01)
ACTION(*SND)
PRDID(1ACCOUN)
LICTRM(V5R2M0)
CPNAME((*NETATR SYS1))
```

Add an activity to send the licenses for the 1ACCOUN product, with a license term of V5R2M0, to the license repository of the iSeries server SYS1. If 1ACCOUN product is installed on the managed system, the license is also activated.

Example 2: Adding an Activity to Send the License for Product 1CHECKS with License Term V5

```
ADDLICRQA CRQD(MYLIB/CR3)
  ACTIVITY(ACT02)
  ACTION(*SND)
  PRDID(1CHECKS)
  LICTRM(V5)
  STRTIME(('23:00:00' '9/30/02'))
  NODL(NETLIB/ACCTSYS)
```

This example shows how to add an activity to send the license for product 1CHECKS with license term V5 to the systems identified in the ACCTSYS node list. Only the license keys matching the managed system serial number are added to the license repository. The license activity will be performed at 11 PM on September 30, 2002.

Error messages for ADDLICRQA

*ESCAPE Messages

None <<

ADDLICENSE (Add License Key Information) Command Description

ADDLICENSE Command syntax diagram

Purpose

The Add License Key Information (ADDLICENSE) command can be used to add the software license key information to the license repository for products with keyed compliance. Products with **keyed compliance** require that you have a software license key from the software provider in order to change the usage limit or the expiration date of the license information.

The license repository stores product license information for each unique product, license term, feature, and system. The repository can contain licenses for any system, and the product need not be installed.

If the product is installed on the system and the license is for this system, this command installs the license, which changes the usage limit from the product's default usage limit to the licensed usage limit. The expiration date is also set.

If a software license key already exists in the repository for the unique product, license term, feature, and system, the software license key information is replaced.

If a license does not exist on the system, the added software license key uses the default values for its threshold (90 percent of the usage limit), message queue (*OPSYS), and log (usage limit violations are not logged) attributes. If a license already exists, the values on these attributes do not change. To change any of these values, you can use the Change License Information (CHGLICINF) command.

Restrictions: This command is shipped with public *EXCLUDE authority.

Optional Parameters

LICENSEINP

Specifies how the software license key information to be added is supplied by the user.

***PROMPT:** The software license key information is entered through prompting.

***LICKEYFILE:** The software license key information is taken from the file specified on the LICKEYFILE parameter.

***TAPE:** The software license key information is taken from a data file with the label QFILEPGMKEY on the tape device specified on the DEV parameter. The QFILEPGMKEY data file must be created on the tape using the Copy To Tape (CPYTOTAP) command, specifying a file in the format of QSYS/QALZAKEY on the FROMFILE parameter.

PRDID

Specifies the seven-character identifier of the product for which software license key information is added.

LICTRM

Specifies the license term for which software license key information is added. This information is supplied by the software provider. Specify the license term in Vx, VxRy, or VxRyMz format, where x and y can be a number from 0 through 9, and z can be a number from 0 through 9 or a letter from A through Z.

FEATURE

Specifies the feature of the product specified on the PRDID parameter for which the software license key information is added.

5001: The software license key information for feature 5001 is added.

feature: Specify the number of the feature for which software license key information is added. Valid values range from 5001 through 9999.

SERIAL

Specifies the serial number of the system for which software license key information is added.

Note:

The *REMOTE and *ALL values are valid only when *LICKEYFILE or *TAPE is specified on the LICKEYINP parameter.

***LOCAL:** The software license key information for the local system is added.

***REMOTE:** The software license key information for all remote systems named in the file specified on the LICKEYFILE parameter or in the tape file with the label QFILEPGMKEY is added.

***ALL:** The software license key information for all systems named in the file specified on the LICKEYFILE parameter or in the tape file with the label QFILEPGMKEY is added.

system-serial-number: Specify the serial number of the system for which software license key information is added.

PRCGRP

Specifies the group of the system processor for which software license key information is added. The software provider supplies this information with the software license key.

***ANY:** The software license key is for any processor group.

processor-group: Specify the processor group of the system for which software license key information is added.

LICKEY

Specifies the software license key that is supplied by the software provider. The license key has three elements, each of which are six characters long. Valid values for the characters are A-F and 0-9, and can be specified in the format: ccccc ccccc ccccc, where c is a valid character.

USGLMT

Specifies the usage limit for this product or feature. The software provider authorizes the value of the usage limit. For a concurrent usage limit, this is the maximum number of jobs allowed to access the product or feature at any given time. For a registered usage limit, this is the maximum number of license users that can be registered to use this product or feature.

1: The number of users is limited to one.

***NOMAX**: The number of users is not limited.

usage-limit: Specify the maximum number of users for this product or feature. Valid values range from 0 through 999999.

EXPDATE

Specifies the expiration date of the product license. After this date, no users over the default usage limit can use the product or feature in compliance with this software license key.

The software provider supplies the expiration date with the software license key. To use the product after the expiration date, you must obtain a new software license key from the software provider.

***NONE**: The license has no expiration date.

expiration-date: Specify the expiration date of the product license.

VNDDTA

Specifies the vendor data. The software provider supplies this information with the software license key.

***NONE**: No vendor data is specified.

vendor-data: Specify a maximum of eight characters of vendor data.

LICKEYFILE

Specifies the qualified name of the file from which the software license key information is taken. This input file must be in the format of QSYS/QALZAKEY, and can be created by using the LICKEYFILE parameter on the Display License Key Information (DSPLICKEY) command.

The name of the license key file can be qualified by one of the following library values:

***LIBL**: All libraries in the job's library list are searched until the first match is found.

***CURLIB**: The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

license-key-file: Specify the name of the file that contains the software license key information.

LICKEYMBR

Specifies the name of the member in the file specified on the LICKEYFILE parameter from which the software license key information is taken.

***FIRST**: The oldest member in the file is used.

***LAST**: The newest member in the file is used.

license-key-member: Specify the name of the member from which to get information.

DEV

Specifies the name of the tape device holding the tape from which the software license key information is copied.

VOL Specifies the volume identifier of the tape from which the software license key information is copied.

***MOUNTED:** The volume currently placed in the device is used.

volume-identifier: Specify the identifier of the volume from which the software license key information is copied.

SEQNBR

Specifies the sequence number of the data file from which the software license key information is copied.

***SEARCH:** A search is made for a data file with an identifier that matches the label QFILEPGMKEY.

***NEXT:** The next sequence is used if that sequence is for a file with the label QFILEPGMKEY.

sequence-number: Specify the sequence number of the data file with the label QFILEPGMKEY to use for copying the software license key information.

ENDOPT

Specifies what positioning operation is done automatically on the tape volume after the software license key information is copied.

***REWIND:** The tape is automatically rewound, but not unloaded, after the operation has ended.

***LEAVE:** The tape does not rewind or unload after the operation ends. It remains at the current position on the tape drive.

***UNLOAD:** The tape is automatically rewound and unloaded after the operation ends.

Examples for ADDLICKEY

Example 1: Adding License Key Information from Prompt Input

```
ADDLICKEY LICKEYINP(*PROMPT) PRDID(1MYPROD)
LICTRM(V3) SERIAL(1234567)
PRCGRP(P20) LICKEY(123456 7890AB CDEF12)
USGLMT(30) EXPDATE(*NONE) VNDDTA(12345678)
```

This command uses prompting to add the software license key information for feature 5001 of the product 1MYPROD to the license repository. The license term is Version 3. The license allows 30 users to use the system with serial number 1234567. That system is in the processor group of 20 or less. There is no expiration date on the license. Because the product is installed on a system with the specified serial number, the license also is installed.

Example 2: Adding All License Key Information from File Input

```
ADDLICKEY LICKEYINP(*LICKEYFILE) SERIAL(*ALL)
LICKEYFILE(*LIBL/MYKEYFILE) LICKEYMBR(*LAST)
```

This command adds the software license key information for all of the systems in the newest member of the file MYKEYFILE to the license repository. If the product is installed on the system, and the license is for this system, the license also is installed.

Example 3: Adding Local License Key Information from a License Key File

```
ADDLICKEY LICKEYINP(*LICKEYFILE) SERIAL(*LOCAL)
LICKEYFILE(*LIBL/MYKEYFILE) LICKEYMBR(*FIRST)
```

This command adds the software license key information found in the oldest member of the file MYKEYFILE to the license repository for this system only. If the product is installed on this system, the license is also installed.

Example 4: Adding Local License Key Information from Tape

```
ADDLICKEY LICKEYINP(*TAPE) DEV(TAP01)
```

This command searches the mounted volume on device TAP01 for the label QFILEPGMKEY. This data file is used and all software license keys for the local system are added to the repository. The tape is rewound after the operation.

Error messages for ADDLICKEY

*ESCAPE Messages

CPF9E2D

Usage limit cannot be less than current usage.

CPF9E56

&1 license key information records added, &2 not added.

CPF9E6C

The license key cannot be used for processor group &2.

CPF9E6E

Product identifier &1 not valid.

CPF9E69

License key information not found in license key file.

CPF9E80

Error occurred during restoring license keys from tape.

CPF9E83

Expiration date &2 is not valid.

ADDLNK (Add Link) Command Description

ADDLNK Command syntax diagram



Purpose

The Add Link (ADDLNK) command adds a link to an object. The NEWLNK parameter specifies the name for the new link. The OBJ parameter specifies the current name of the object that is to receive the new link. After the link is established the object may be referred to by either the old name or the new name.

For more information about integrated file system commands, see the Integrated file system topic in the File systems and management category of the Information Center.

Restrictions:

1. This command works on only one object. If a pattern is specified on the OBJ parameter and more than one object matches the pattern, you can select the object from a list in an interactive job. If this is a batch job, the command fails with the error message CPFA08E, "More than one name matches pattern." A pattern is not allowed if the LNKTYPE is symbolic, because the object is not required to exist.
2. The user must have write and execute authority to the directory that contains the new link (NEWLNK). If a hard link is being added, the user must also have *OBJEXIST authority to the existing object (OBJ) and execute authority to each of the path name prefixes of the OBJ name.
3. A hard link cannot be created to a symbolic link. When LNKTYPE(*HARD) is specified and the OBJ parameter names a symbolic link, the link is created to the resolved object (which must exist).
4. A hard link cannot be created to a directory.
5. A hard link cannot be created to an object in another file system.

6. No links can be created in the QSYS.LIB,  independent ASP QSYS.LIB,  or QDLS portion of the name space.

Required Parameters

OBJ Specifies the path name of the object you want to add a link to. The object must exist when a hard link is being added. See path names for more information on specify path names.

NEWLNK

Specifies the new path name that can be used to refer to the object. The new name must not exist. See path names for more information on specify path names.

Optional Parameter

LNKTYPE

Specifies whether the link is hard or symbolic.

***SYMBOLIC:** The link to the object is a representation of a path name. This representation is in the form of a path contained in a file. The actual path is determined by doing a path search based on the contents of the file. A symbolic link is also called a soft link.

Symbolic links can cross file systems. An object need not exist. An existing object can be deleted without removing the symbolic link.

***HARD:** The link to the object is an actual path to an existing object. A hard link is established by creating a directory entry.

Hard links cannot cross file systems. When all hard links to an object are removed, the space occupied by the object is freed and the object can no longer be accessed. An object cannot be removed while a hard link to it exists.

Examples for ADDLNK



Example 1: Adding a Symbolic Link

```
ADDLNK OBJ('DECEMBER-1994-MONTHLY-PAYROLL-FILE')
      NEWLNK('PAY')
```

This command adds a symbolic link named "PAY" to the DECEMBER-1994-MONTHLY-PAYROLL-FILE.

Example 2: Adding a Symbolic Link to a Source File

```
ADDLNK OBJ('/QSYS.LIB/MYLIB.LIB/F1.FILE/P1.MBR')
      NEWLNK('PGM1') LNKTYPE(*SYMBOLIC)
```

This command adds a symbolic link from the user's current directory (not in QSYS.LIB,  independent ASP QSYS.LIB,  or QDLS) to a member in a source file in QSYS.LIB.

Example 3: Adding a Hard Link

```
ADDLNK OBJ('/QOpenSys/MYDIR/FILE1') NEWLNK('FILE2')
      LNKTYPE(*HARD)
```

This command adds a hard link from the user's current directory, with the name FILE2, to FILE1 in /QOpenSys/MYDIR.

Error messages for ADDLNK

*ESCAPE Messages

CPFA085

Home directory not found for user &1.

CPFA089

Pattern not allowed in path name.

CPFA08E

More than one name matches pattern.

CPFA093

Name matching pattern not found.

CPFA0A1

An input or output error occurred.

CPFA0A7

Path name too long.

CPFA0B0

Request not allowed to operate from one file system to another.

ADDLANADPI (Add Local Area Network Adapter Information) Command Description

ADDLANADPI Command syntax diagram

Purpose

The Add Local Area Network Adapter Information (ADDLANADPI) command adds an adapter name entry to the adapter file.

Required Parameters**ADPTNAME**

Specifies the name of the adapter being added to the adapter file. The name can be a maximum of 10 characters in length.

ADPTADR

Specifies the 12-character hexadecimal adapter address.

LINETYPE

Specifies the line type of the entry.

***DDI:** A distributed data interface (DDI) line type is used.

***TRN:** A token-ring network (TRN) line type is used.

Optional Parameter

TEXT Specifies the text that briefly describes the adapter. More information is in Commonly used parameters.

***BLANK:** Text is not specified.

'description': Specify no more than 50 characters of text, enclosed in apostrophes.

Example for ADDLANADPI

```
ADDLANADPI ADPTNAME(PAYROLL) ADPTADR(00000000012B)  
LINETYPE(*TRN)
```

This command adds the adapter PAYROLL, with the address 00000000012B, to the network adapter file. The line is a token-ring line.

Error messages for ADDLANADPI

***ESCAPE Messages**

CPF8B48

Adapter name - &29 or address - &30 already in the network adapter file

CPF8B68

Line description &23 not found.

CPF8B69

Line description &23 not valid for requested action.

CPF8B74

Request to display active adapters failed.

CPF8B75

No adapter entries in network adapter file.

CPF8B76

No functional addresses for adapter.

ADDLFM (Add Logical File Member) Command Description

ADDLFM Command syntax diagram

Purpose

The Add Logical File Member (ADDLFM) command adds a named member to the specified logical file, which must already exist on the system. A member must be added to the logical file before the file can have access to data stored in any physical file member. The first member of a file can be added by entering an ADDLFM command or by specifying a member name in the MBR parameter of the Create Logical File (CRTLF) command. To add other members to the file, use the ADDLFM command to specify each one.

A logical file member can use the data from all, or a subset of, the physical files referenced by the logical file. Each member has its own set of data and can have its own access path that provides an organization to that data. The system attempts to implicitly share an access path already on the system.

The number of members that can be added to a logical file is limited to the number specified in the MAXMBRS parameter of the associated Create Logical File (CRTLF) command. Each added member has the same attributes as those defined in the logical file.

Restrictions:

1. To add a member to a keyed logical file, the user must have Object operational and object management authorities or object alter authorities for each of the physical files on which the logical file member is based (specified explicitly by the DTAMBRS parameter or implicitly by the PFILE or JFILE keyword specified in DDS). For a member added to a non-keyed logical file, object operational authority for each of the physical files is required.
2. In multithreaded jobs, this command is not threadsafe for distributed files and fails for distributed files that use relational databases of type *SNA. This command is also not threadsafe and fails for Distributed Data Management (DDM) files of type *SNA, when SYSTEM(*RMT) or SYSTEM(*FILETPYE) is specified.

Note:

An *EXCLRD lock is required on the file to add a member. Because this command adds a member to a file in a library, the library must not be locked (*SHRNUP or *EXCLRD with the Allocate Object (ALCOBJ) command) in another job.

Required Parameters

FILE Specifies the qualified name of the logical file to which this member is added.

The name of the logical file can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

If a DDM file is specified, the target system specified must be an iSeries 400 server.

logical-file-name: Specify the name of the file to which the member is added.

MBR Specifies the name of the logical file member that is being added. The member name must be unique in the file to which it is being added.

If the FILE parameter specifies a DDM file and a member name is specified as part of the remote file name in the DDM file, the member names must be the same.

Optional Parameters

DTAMBRS

Specifies the names of the physical files and members that contain the data associated with the logical file member being added by this command. A logical file member can be based on all of the physical files and members on which the logical file itself is based, specified by DTAMBRS(*ALL), or the member can be based on a subset of the total files and members, specified by DTAMBRS(qualified-file-names [member-names]).

Note:

When adding a member to a logical file that is a DDM file, the physical file, if specified, must also be a DDM file with its library and member(s) specified explicitly. *CURRENT is not supported when the logical file is a DDM file.

When a logical file is created, the physical files specified on the PFILE or JFILE DDS keyword are used to create the logical file. If no library name is specified for the physical files on the PFILE or JFILE keyword, the library list (*LIBL) at file creation time is used to find the physical files; the physical files from the library list are used to create the logical file. The qualified physical files from the PFILE or JFILE keyword (regardless of whether a library name was specified or if the library list was used to find the files) are the physical files associated with the logical file. The names of the physical files associated with the logical file are saved in the description of the logical file. When a member is added to the logical file, the DTAMBRS parameter is used to specify the physical file members associated with the logical file member. Each physical file name specified on the DTAMBRS parameter must be the name of a physical file that is associated with the logical file (saved in the description of the logical file).

***ALL:** The logical file member being added is based on all the physical files and members (that exist at the time this CRTLF command is entered) used by the logical file. At least one member must exist in at least one of the physical files. The physical file names are specified on the PFILE or JFILE parameter in the DDS.

***CURRENT:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name/: Specify the name of the library to be searched.

If a library name is not specified, the current library name (*CURRENT) from the logical file description is used. If the library name is specified, the physical file must be a physical file associated with the logical file. If the logical file is associated with more than one physical file of the same name, the library name must be specified.

Element 1: Names of Physical Files

physical-file-name: Specify the names of the physical files that contain the data being accessed by the logical file member being added.

The physical file names must match a name on the PFILE or JFILE keywords in the DDS and cannot be specified more often on the DTAMBRS parameter than on the PFILE or JFILE keywords in the DDS. For join logical files, all physical files specified on the JFILE keyword must be specified on the DTAMBRS parameter and each physical file must contain only one member. If a physical file name is not specified for a physical file that is on a PFILE or JFILE keyword in the DDS, the logical file member is not based on any member of that physical file.

Element 2: Names of Members

***NONE:** A member name is not specified.

member-name: Specify the names of the members that contain the data being accessed by the logical file member being added.

When the FILE parameter specifies a join logical file or an arrival sequence logical file, only one data member must be specified on the DTAMBRS parameter for each physical file that was specified on the PFILE or JFILE keyword in the DDS. *ALL is valid only if each based-on physical file has only one member. If any of the physical files has more than one member, the specific physical file member must be specified on the DTAMBRS parameter.

The same physical file name can be specified more than once on the JFILE keyword. In this case, each occurrence of the file name is treated as a different based-on physical file, and must be specified on the DTAMBRS parameter.

Up to 32 qualified physical file names and physical file member names can be specified. Also, the total number of member names cannot exceed 32. For example, one file can specify 32 members, two files can each have 16 members, or 32 files can each have one member specified.

For DDM file:

- The file names specified in the DTAMBRS parameter must be the names of the DDM files that represent the remote based-on physical files. If a member name was specified as part of the remote file name in the DDM file, only that member name can be specified on the DTAMBRS parameter. The member names must be the actual remote file member names.
- The based-on physical files must be at the same system location as the logical file to which the member is being added.
- When no member name is specified for the remote file name in the DDM file, all members are accessible. When only one member name is specified, only that member is accessible through that DDM file.

The following examples show the syntax for specifying single and multiple members for single and multiple physical files. In the examples, the abbreviation PF represents a physical file name, LIB represents a library qualifier, and M represents a member name.

```
Single physical file and member:  
DTAMBR((PFA M1))  
Single file with multiple members:  
DTAMBR(PFA (M1 M2 M3))  
Multiple files with single members and no members:  
DTAMBR((PFA M1) (PFB M4) (PFE *NONE))  
Multiple files with multiple members:  
DTAMBR((PFA (M1 M3 M4)) (PFB (M1 M2 M4)))  
Multiple files with the same name in different libraries:  
DTAMBR((LIBX/PFA M1) (LIBY/PFA (M1 M2)))  
Multiple files with the same name in the same library:  
DTAMBR((LIBX/PFA M1) (LIBX/PFA M1))
```

If more than one physical file member is specified for a physical file, the member names are specified in the order in which records are retrieved if duplicate key values occur across those members.

SHARE

Specifies whether the open data path (ODP) for the logical file member is shared with other programs in the routing step. When an ODP is shared, the programs accessing the file share facilities such as the file status and the buffer.

More information on shared database files is in the Database Programming topic in the Information Center.

***NO:** The ODP created by the program with this attribute is not shared with other programs in the routing step. Every time a program opens the file with this attribute, a new ODP to the file is created and activated.

***YES:** The ODP created with this attribute is shared with each program in the routing step that also specifies SHARE(*YES) when it opens the file.

Note:

When SHARE(*YES) is specified and control is passed to a program, a read operation in that program retrieves the next input record. A write operation produces the next output record.

TEXT Specifies the text that briefly describes the logical file member. More information is in Commonly used parameters.

***BLANK:** Text is not specified.

'description': Specify no more than 50 characters of text, enclosed in apostrophes.

Example for ADDLFM

```
ADDLFM FILE(INVENLIB/STOCKTXS)  
      MBR(JANUARY)  
      DTAMBR((INVENTXS JANUARY))  
      TEXT('JANUARY STOCK ACTIVITY BY LOCATION')
```

This command adds a member named JANUARY to the logical file named STOCKTXS in the INVENLIB library. The logical file has access to the data stored in the JANUARY member of the INVENTXS physical file.

Error messages for ADDLFM

*ESCAPE Messages

CPF3204

Cannot find object needed for file &1 in &2.

CPF7306

Member &1 not added to file &2 in &3.

ADDMEDIBRM (Add Media Information to BRM) Command Description

Note: To use this command, you must have the 5722-BR1 (Backup Recovery and Media Services for iSeries) licensed program installed. For detailed information on the parameters of this command, see the online help.

ADDMEDIBRM Command syntax diagram

Purpose

The Add Media Information to BRM (ADDMEDIBRM) command adds file level detail to BRMS media inventory volume content information. The files, and the volumes that contain them, can be from another tape inventory or from some other outside source. The purpose of the command is to allow user applications or another tape management system to insert data (tape file descriptions) into the BRMS media volume content information so that the volumes and their contents can be managed. The volume for which you are adding content information must exist in the media inventory and the volume, file sequence and volume sequence must be unique in the media inventory content information. Multiple volumes will write a media information record for each volume with the volume sequence being incremented as the records are added.

Note: This command adds records in the BRMS media inventory contents information based on the information you supply, particularly in regard to file sequence, volume and so on. It is critical that you are careful to apply the correct information and have a full understanding of the command before you use it.

Note: You can only add media content information to an expired volume.

Examples for ADDMEDIBRM

Example 1: Adding Contents to a Volume

```
ADDMEDIBRM TYPE(*ALLDLO) VOL(T00001) SEQNBR(1)
VOLSEQ(1)
```

In this example an entry is made in the media inventory content information for volume T00001. The volume's contents are updated to show that a save of the document library resides as file sequence number 1 on the first volume.

Error messages for ADDMEDIBRM

None

ADDMLMBRM (Add Media Library Media to BRM) Command Description

Note: To use this command, you must have the 5722-BR1 (Backup Recovery and Media Services for iSeries) licensed program installed.

ADDMLMBRM Command syntax diagram

Purpose

The Add Media Library Media to BRM (ADDMLMBRM) command adds volumes to a media library (MLB). The command adds the specified volumes to a usable category and optionally enrolls them to BRMS. If the ADDVOL parameter is *YES, you can specify a media class for the volume or volumes that you are adding. If the ADDVOL parameter and the INZ parameters are both *YES, you are supplied with additional parameters such as move policy and initialization information.

Examples for ADDMLMBRM

Example 1: Adding a Volume to a Media Library

```
ADDMLMBRM MLB(MLB01) VOL(T00001) INZ(*NO)
ADDVOL(*NO)
```

In this example, volume T00001 is added to the media library MLB01 but is not initialized. Volume T00001 must be a member of the BRMS media inventory.

Example 2: Adding and Initializing a Volume to BRMS

```
ADDMLMBRM MLB(MLB01) VOL(*INSERT) INZ(*YES)
MEDCLS(CART3490E)
```

In this example, all volumes that are in the *INSERT category are added to the media library MLB01 and the BRMS media inventory as media of class CART3490E. The volumes are initialized with the density specified in media class CART3490E.

Error messages for ADDMLMBRM

None

ADDMEDBRM (Add Media to BRM) Command Description

Note: To use this command, you must have the 5722-BR1 (Backup Recovery and Media Services for iSeries) licensed program installed. For detailed information on the parameters of this command, see the online help.

ADDMEDBRM Command syntax diagram

Purpose

The Add Media to BRM (ADDMEDBRM) command adds a volume to the BRMS media inventory. The volume can be a volume from another tape inventory that contains active data, a volume from some other outside source, or a new volume that you want to initialize. Once added, BRMS tracks the volume's characteristics, location, use and content.

When you add the volume, you must specify the media class of the volume. You can also specify how you want the volume to move, where the volume is located, its container (if any) and other miscellaneous attributes.

If the numbering scheme of the volumes that you are adding is consecutive, the ADDMEDBRM command will automatically add the volumes without having to add the volumes one at a time.

Note: OS/400 uses certain volume identifiers for special purposes. You should avoid using these volume identifiers in your volume labeling. Volume identifiers that you should avoid are:

- TAPxxx
- NLTxxx
- BLKxxx
- CLNxxx

- ERRxxx
- SLTxxx
- IMPxxx

Examples for ADDMEDBRM

Example 1: Adding a Volume to BRMS

```
ADDMEDBRM VOL(T00001) MEDCLS(QIC1000)
```

This command adds volumes to the BRMS media inventory. In this example, volume T00001 is assigned a media class of QIC1000 and is added to the BRMS media inventory. The volume is not initialized and is added as expired.

Example 2: Adding and Initializing a Volume to BRMS

```
ADDMEDBRM VOL(T00002) MEDCLS(QIC1000)
INZ(*YES) DEV(TAP01)
```

This command adds the volume T00002 to the BRMS media inventory with a media class of QIC1000. The volume is initialized using device TAP01.

Error messages for ADDMEDBRM

None

ADDMSGD (Add Message Description) Command Description

ADDMSGD Command syntax diagram

Purpose

The Add Message Description (ADDMSGD) command describes a message and stores it in a message file for later use. The message description remains in the message file until the file is deleted or until the Remove Message Description (RMVMSGD) command is used to remove it from the file. To change any of the attributes of the message description, such as its message text or severity code, use the Change Message Description (CHGMSGD) command.

Note:

A description of how to print a single message is in [Print messages](#) article in the Information Center or how to print a group of messages is in the [Printing all messages in the message queue](#) article in the Information Center.

Substitution variables can be embedded both in the first-level and second-level message text. They can be replaced later by message data fields specified in the Retrieve Message (RTVMSG), Send User Message (SENDUSRMSG), and Send Program Message (SENDPGMMSG) commands.

Note:

The *type* of message being defined is *not* specified in the ADDMSGD command. The type is specified in the command that actually sends the message.

If the message and its second-level text exceeds 512 characters, it will not fit in the prompt field. In this case, enter the command on the Command Entry panel or in a CL program.

Restriction: To add a message description to a message file, the user must have *USE and *ADD authorities to the message file.

Required Parameters

MSGID

Specifies the message identifier under which the message is stored in the message file. Every message must have a unique identifier.

The message identifier must be 7 characters long and made up of a message prefix and a message number, as shown in the following format:

pppnnnn

The first 3 characters (ppp) must consist of an alphabetic character followed by two alphanumeric (alphabetic or decimal) characters; the last 4 characters (nnnn) must consist of hexadecimal numbers ranging from 0 through 9 and A through F.

For **user-defined messages**, the same format *must* be used; in addition, the 3-character prefix should start with a U to distinguish user-defined messages from IBM-supplied messages. For example, the message identifier of a message in a payroll application message file could be UPY0027.

Following is a representative sample of the message identifier prefixes that are used to identify messages in several of the IBM-supplied message files. Shown are some of the prefixes for the OS/400 operating system, ILE COBOL/400*, ILE RPG/400*, the IDU and SDA utilities, and keyboard and machine interface (MI) messages.

CPA	OS/400 system operator action
CPC	OS/400 system completion messages
CPD	OS/400 system diagnostic messages
CPI	OS/400 system informational messages
CPX	OS/400 system titles and texts
CPZ	OS/400 system abnormal termination
CBE	COBOL run time
CBL	COBOL compiler
CBX	COBOL titles and texts
CSC	COBOL syntax checker
QRG	RPG language compiler
RPG	RPG run time
RPT	RPG auto report
RSC	RPG syntax checker
RTX	RPG auto report titles and texts
RXT	RPG relational diagnostic texts
IDU	Interactive database utilities (IDU)
IDX	IDU titles and text
SDA	Screen design aid (SDA)
SDX	Screen design aid titles and texts

KBD Keyboard

MCH iSeries 400 machine instruction interface

MSGF Specifies the qualified name of the message file where the message is to be stored. The IBM-supplied message files (for example, QCPFMSG and QRPMSG) cannot be specified unless the user entering the command has authority to update those files. When the system is installed, only the system security officer has that authority. Message file overrides in effect for the job are ignored by this command and the message is stored in the specified file.

The name of the message file can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

message-file-name: Specify the name of the message file to use.

MSG Specifies the first level of message text of the message being defined. This text is the message that is initially shown or printed, or sent to a program or log. Up to 132 characters (enclosed in apostrophes) can be specified, but the limitations of the display stations (their screen size) should be considered. The entire message must be enclosed in apostrophes if blanks are included in the message. To code an apostrophe for use in the message, enter a double apostrophe.

One or more substitution variables can be embedded in the message text string to indicate positional replacement fields. These replacement fields allow substitution of variable data in the message by the program before the message is sent. The rules below must be followed if variables are used.

- Variables must be specified in the form &n, where n is a 1- or 2-digit (1-99) number identifying the data field that is substituted.
- Variables can be *preceded* by any alphanumeric character (including blanks). For example, the variables shown in the message below are valid.

Command&34&72 &2 &99help

- Variables can be followed by any non-numeric character (the character following the variable cannot be digits 0-9). For example, the variables shown in the following example are *not* valid.

Command&345 &244 &999help

- The variables can be enclosed in apostrophes if only the variables themselves make up the message. For example, to show a two-part decimal value, the message '&1.&2' can be specified.)
- Variables for this parameter do not have to be described positionally (in ascending or descending sequence).

Note:

The data fields are described positionally in the FMT parameter and are specified positionally in the MSGDTA parameter of the Send Program Message (SNDPGMMMSG) and Send User Message (SNDUSRMSG) commands. Details on substituting data fields in message text are in

the CL Programming  book.

'message-text': Specify the first-level message text, enclosed in apostrophes.

Double-Byte Character Set Considerations

If entering double-byte characters on this parameter, several combinations of characters may cause errors to occur on this command. If the double-byte characters contain the string, X'50Fn' (where n is a 1-digit number, ranging from 0 through 9), error messages CPF2424 or CPF2431 may be sent. Examples are: X'50F0', X'50F4', X'50F9'.

Coded Character Set Identifier (CCSID) Considerations

The text supplied on the MSG parameter is assumed to be in the CCSID of the job running this command unless the CCSID parameter is coded. If the CCSID parameter is coded, the text is assumed to be in the CCSID specified. For more information about the message handler and its use of CCSIDs, see the Globalization topic in the Information Center.

Optional Parameters

SECLVL

Specifies whether any second-level message text is shown to a display station user to further explain the message specified in the MSG parameter. The user presses the Help key to request the second-level message text. Second-level message text can also be written to the job log if *SECLVL is specified on the LOG parameter of the job commands.

***NONE:** There is no second-level message text for this message description.

'second-level-message': Specify the text shown as the second-level message text if it is requested by the user. No more than 3000 characters, enclosed in apostrophes, can be specified, but display limitations must be considered. One or more substitution variables can be embedded in the second-level message text, as described in the MSG parameter.

The second-level message text can be formatted for the display station using three format control characters:

Note:

When formatting second-level message text for the display station, the format control characters must be followed by a blank space.

- &N** Forces the text to a new line (column 2). If the text is longer than one line, the next lines are indented to column 4 until the end of the text or until another format control character is found.
- &P** Forces the text to a new line, indented to column 6. If the text is longer than one line, the next lines start in column 4 until the end of the text or until another format control character is found.
- &B** Forces the text to a new line, starting in column 4. If the text is longer than one line, the next lines are indented to column 6 until the end of the text or until another format control character is found.

Double-Byte Character Set Considerations

If entering double-byte characters on this parameter, several combinations of characters may cause errors to occur on this command. If the double-byte characters contain the string, X'50Fn' (where n is a 1-digit number, ranging from 0 to 9), error messages CPF2424 or CPF2431 may result. Examples are: X'50F0', X'50F4', X'50F9'.

Coded Character Set Identifier (CCSID) Considerations

The text supplied on the SECLVL parameter is assumed to be in the CCSID of the job running this command unless the CCSID parameter is coded. If the CCSID parameter is coded, the text is assumed in the CCSID specified. For more information about message handler and its use of CCSIDs, see the Globalization topic in the Information Center.

SEV Specifies the severity code of the message. The severity code indicates the severity level of the condition that causes the message to be sent.

00: The severity code assigned to this message is 00.

severity-code: Specify the severity level associated with this message. Valid values range from 00 through 99. The assigned code for the message should correspond in importance to the IBM-predefined severity codes. Any 2-digit value can be entered, even if no severity code (either predefined or user-defined) has been defined for it. More information is in Commonly used parameters.

FMT Specifies the formats of up to 99 message data fields. Each field is described in this parameter by a list of attributes. All 99 of the message data fields can be used as substitution values in the first- and second-level message text defined in this message description. They also can be specified on the DMPLST or ALROPT parameter of this command. If specified in the MSGDTA parameter of the SNDPGMMSG or SNDUSRMSG commands, the data fields must be concatenated in one character string and must match the format and sequence specified. The length of the entire character string of concatenated message data fields cannot exceed 512 characters.

***NONE:** No format is being described for message fields. If *NONE is specified, or if this parameter is omitted, no references can be made to message data fields in the MSG, SECLVL, DMPLST, or ALROPT parameters.

type [length [decimal-positions]]: The format of each message data field (up to 99 fields) to be substituted in the message in this message description is defined by a list of attributes. These attributes specify the type of data in the field, the length of the field, and, optionally, the number of decimal digits to the right of the decimal point. Certain data types do not require a length field. Boundary alignment requirements must be considered (for example, pointers are always aligned on 16-byte boundaries).

Type of Message Data: The first value, type, specifies the type of data the substitution field contains and how the data is formatted when substituted in the message text. The contents of the second and third values vary depending on the type specified. One of the following types can be specified for each field described by this parameter:

***QTDCHAR:** A character string formatted with enclosing apostrophes ('Monday, the 1st') is used.

***CHAR:** A character string formatted without enclosing apostrophes is used. An alphanumeric string is used, for example, to specify a name (BOB). Trailing blanks are truncated.

***HEX:** A string of bytes formatted as a hexadecimal value (for example, X'C0F4') is specified.

***SPP:** A 16-byte space pointer to data in a space object is specified. If referred to in the DMPLST parameter, the data in the space object (from the offset indicated by the pointer) for the length specified, is to be dumped. *SPP is not valid as a replacement field in message text.

***DEC:** A packed decimal number that is formatted in the message as a signed decimal value with a decimal point is used. Values for length (required) and decimal positions (optional) are specified for this type (*DEC) to indicate the number of decimal digits and the number of digits to the right of the decimal point. Zeros to the left of the first significant digit are suppressed, and leading blanks are truncated (removed). If a decimal position other than zero is specified, a decimal point is shown in the result even if the decimal precision in the result is zeros; examples are 128.00 and 128.01, if FMT(*DEC 5 2) is specified. If the number of decimal positions is not specified, zero is assumed. The following gives two examples:

- If FMT(*DEC 2) is specified for a substitution field and the message data is a packed decimal value of X'058C', the message text contains a positive value of 58 with no decimal point indicated.
- If FMT(*DEC 4 2) is specified and the packed value is specified as X'05810C' (3 bytes long), the text contains the formatted decimal value of 58.10.

***BIN:** A binary value that is 2, 4, or 8 bytes long (B'0000 0000 0011 1010') and is formatted in the message as a signed decimal value (58) is specified.

***UBIN:** A binary value that is 2, 4, or 8 bytes long (B'0000 0000 0011 1010') and is formatted in the message as an unsigned decimal value (58) is specified.

***CCHAR:** A character string that can be converted. If data of this type is sent to a message queue that has a CCSID tag other than 65535 or 65534, the data is converted from the CCSID specified by the send function to the CCSID of the message queue. Conversions can also occur on data of this type when the data is obtained from the message queue using a receive or display function. See the Message Handler section of the Globalization topic in the Information Center for more details on CCSID conversions.

The following formats are valid only in IBM-provided message descriptions and should not be used for other messages.

***DTS:** An 8-byte field that contains a system date time stamp is specified. The date time stamp contains the date followed by one blank separator and the time. The output message date is formatted as specified by the job date format and job date separator. system values QDATFMT and QDATSEP. The time is formatted as hh:mm:ss.

***SYP:** A 16-byte system pointer to a system object is specified. If referred to in message text, the simple name of the system object is formatted as described in the name type, *CHAR. If referred to by the DMPLST parameter, the object itself is dumped.

***ITV:** An 8-byte binary field that contains the time interval (in seconds) for wait time-out conditions is specified. The time interval is formatted in the message as a zero-suppressed zoned decimal value (15 0) representing the number of seconds to wait.

Length of Message Data: Following the type specification, a second value (length) can be specified to indicate the number of characters or digits that are passed in the message data. How the second value is used depends on the type specified in the first value.

1. If a length is not specified for *QTDCHAR, *CHAR, *HEX, or *SPP, then *VARY is assumed for the length. If *VARY is specified or assumed, the message data field passed by the SNDUSRMSG or SNDPGMMSG commands must be preceded by a 2-byte or 4-byte binary field that indicates the actual number of bytes of data being passed. However, if *SPP is specified, the length field is contained in the first bytes pointed to by the space pointer. Therefore, the 2- or 4-byte field must precede the data pointed to by the space pointer, and must *not* precede the space pointer that is passed as part of the message data.
2. If the type *DEC is specified, the total number of decimal digits (including the fraction) *must* be specified as the second value; the number of digits in the fraction optionally can be specified as the third value.
3. If the type is *BIN or *UBIN is specified, the message data field can be only 2, 4 or 8 bytes long; the default value is 2 bytes.
4. If the type *CCHAR is specified, the message data length field can be only *VARY. A variable length field is required because as the data in this field gets converted to different coded character set identifiers (CCSIDs), its length may change.

Length Field Size/Decimal Positions: The third value is used in one of two ways, depending on the type specified in the first value. (1) If *QTDCHAR, *CHAR, *CCHAR, *HEX, or *SPP is specified, and if *VARY is specified or assumed for the second value, the third value is used with *VARY to indicate the size of the length field actually passed. The third value can be either a 2 or a 4, which is the number of bytes used to specify the length (in binary) of the passed value. (2) If *DEC is specified, the third value indicates the number of decimal positions in the decimal value. If not specified, the default is 0 decimal positions.

Note: If an object has been damaged or deleted, the substitution variable is not replaced by the object name if it is shown; instead, the variable appears as &n, where n = number. If the length of the message data that is passed to the substitution variable is shorter than the length specified for FMT, the substitution value becomes a null field.

If the message is an inquiry message (specified by *INQ in one of the send message commands) or a notify message (specified by *NOTIFY in the SNDPGMMSG command only) and a reply is expected, seven parameters can be used to specify some requirements that relate to the reply received. The seven validity checking parameters are: TYPE, LEN, VALUES, SPCVAL, RANGE, REL, and DFT.

These parameters are not necessary for a message to allow a reply, but they can be used to define valid replies that can be made to the message. Also note that the VALUES, RANGE, and REL parameters are mutually exclusive; only one of them can be specified in this command.

TYPE Specifies the type of reply that is valid to respond to an inquiry or notify message.

***CHAR:** Any character string is valid. If it is a quoted character string, the apostrophes are passed as part of the character string.

***NONE:** No reply type is specified, and no reply validity checking is performed. LEN(*NONE) must also be specified.

***DEC:** Only a decimal number is a valid reply.

***ALPHA:** Only an alphabetic (A through Z, \$, #, and @) character string is valid. Blanks are not allowed.

***NAME:** Only a simple name is a valid reply. The name does not have to be an OS/400 system object name, but it must start with an alphabetic character; the remaining characters must be alphanumeric. If all reply characters are alphabetic (A-Z), the reply is converted to uppercase.

LEN Specifies the maximum length of a reply to an inquiry or notify message. The values specified under *TYPE apply *only* if one or more of the other validity checking parameters are specified. If none of the validity checking parameters are specified, the reply type *CHAR can contain as many as 132 characters.

***TYPE:** The maximum length is determined by the type of reply specified on the TYPE parameter. The maximum length for each type of reply is as follows:

- Up to 132 characters can be specified for types *CHAR and *ALPHA. If additional validity checking is being performed (for example, if VALUES, RANGE, REL, or SPCVAL are specified), the maximum length allowed for *CHAR and *ALPHA is 32 characters.
- Up to 15 digits can be specified for *DEC, of which up to 9 digits can be to the right of the decimal point.
- Up to 10 alphanumeric characters can be specified for *NAME.

***NONE:** No reply length is specified. No reply validity checking is performed if this message is sent as an inquiry or notify message. TYPE(*NONE) must also be specified.

length [decimal-positions]: Specify the maximum length allowed for the message reply. The length specified cannot exceed the maximums specified on the *TYPE value. If the reply type is a decimal value, the number of decimal positions can be optionally specified; if it is not specified, zero decimal positions are assumed.

VALUES

Specifies a list of values of which one can be received as a valid reply to an inquiry or notify message. No more than 20 values can be specified in the list. Each value in the list must meet the requirements specified for message replies by the TYPE and LEN parameters.

If VALUES is specified, the RANGE and REL parameters cannot be specified. To be valid, a reply must match one of the values in this list.

For the reply value to match the compare value, both must be of the same keyboard shift. For example, if the program requires a reply containing uppercase characters, one of the following methods ensures a response in uppercase characters:

- Requiring a response in uppercase characters.
- Entering the compare values for the VALUES parameter in lowercase, but using the SPCVAL parameter to convert the characters to uppercase.
- Using the TYPE(*NAME) keyboard value to convert the characters to uppercase. To use this method, all reply characters must be alphabetic (A through Z).

***NONE:** No list of reply values is specified. The reply can have any value that is consistent with the other validity specification parameters.

value: Specify a list of up to 20 values to compare with a reply value that is sent in response to the message defined in this message description. The reply value must match one of the values in this list to be a valid reply to this message. The maximum length of each value is 32 characters.

SPCVAL

Specifies a list of up to 20 sets of special values of which one set (if the from-value is matched by the sent reply) is used as the reply for an inquiry or notify message. These values are special in that they may not meet all the validity checking specifications given in the other reply-oriented parameters. The reply sent is compared to the from-value in each set; if a match is found, and a to-value was specified in that set, the to-value is sent as the reply. If no to-value was specified, the from-value is sent as the reply. The to-value must meet the requirements specified in the TYPE and LEN parameters. If the reply sent does not match any from-value, the reply is checked for validity by the specifications in the other reply-oriented parameters.

***NONE:** No special values are specified for the replies to this message.

from-value [to-value]: Specify a list of up to 20 sets of special values to determine the reply to this message. Each set must have a from-value, with which the reply is compared, and an optional to-value, which is sent as the reply if its from-value matches the reply.

RANGE

Specifies the lower and upper value limits for valid replies to an inquiry or notify message. These values must meet the requirements specified for replies by the TYPE and LEN parameters, and both values must be of the same type. If both values are not of the same length, the shorter value is padded on the right with blanks. For the replies of types *CHAR and *ALPHA, the reply is padded on the right with blanks, or truncated on the right, to the length of the specified values, before the value range is checked for validity. If RANGE is specified, the VALUES and REL parameters cannot be specified.

***NONE:** No range values are specified for the replies to this message.

lower-value upper-value: Specify the upper and lower value limits for replies to an inquiry or notify message.

REL Specifies the relation that must be met for a valid reply to an inquiry or notify message. The value

specified must meet the requirements specified for replies by the TYPE and LEN parameters. For replies of the types *CHAR and *ALPHA, the reply is padded on the right with blanks, or truncated on the right, to match the length of the value specified, before the system performs the relational test on the reply value that is sent. If REL is specified, the VALUES and RANGE parameters cannot be specified.

***NONE:** No relational values are specified for the replies to this message.

operator-value: Specify one of the relational operators and the value against which the message reply is checked for validity. If the reply is valid in the relational test, it is sent to the message sender. The relational operators that can be entered are:

- *LT Less than
- *LE Less than or equal to
- *GT Greater than
- *GE Greater than or equal to
- *EQ Equal to
- *NL Not less than
- *NG Not greater than
- *NE Not equal to

DFT Specifies, if the message is an inquiry or notify message, the default reply (enclosed in apostrophes, if it contains special characters) used if the receiver of the message has indicated that all incoming messages are to receive default replies, or if a message is deleted from a message queue and no reply was specified. The default reply can also be used to answer unmonitored notify messages. The default reply must meet the requirements specified for replies by the validity specification parameters, TYPE and LEN.

***NONE:** No default reply is specified for the replies to this message.

'default-reply': Specify the default reply to send (enclosed in apostrophes if it contains special characters) to inquiry or notify messages.

DFTPGM

Specifies the qualified name of any default program called to take default action if this message is sent as an escape message to a program or procedure that is not monitoring for it. This parameter is ignored if the message is *not* sent as an escape message. If it is sent as an escape message, the following parameters are passed to the specified default program:

- Name of the program or procedure to which the message is sent (277 characters). The program name, module name, procedure name, and program type of the call message queue to which the message is sent. This name is the same as that of the program or procedure that did not monitor for the escape message.
 - Characters 1 through 10 are the name of the program to which the message is sent.
 - Characters 11 through 20 are the name of the module to which the message is sent. If the message is not sent to an Integrated Language Environment (ILE) procedure, the value *N is returned in this field, padded on the right with blanks.
 - Characters 21 through 276 are the name of the procedure to which the message is sent. If the message is not sent to an ILE procedure, the value *N is returned in this field, padded on the right with blanks.
 - Character 277 is set to the value 1 if the message is sent to an ILE procedure, or to the value 0 if the message is not sent to an ILE procedure.
- Message reference key (4 characters). The message reference key of the escape message on the program message queue.

***NONE:** No default program is specified for this message.

The name of the program can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

default-program-name: Specify the name of the default program that is called if an escape message is sent.

DMPLST

Specifies the data that is dumped if this message is sent as an escape message to a program that is not monitoring for it. This parameter can specify that data related to the job be dumped, that data from message data fields be dumped, or that a combination of these be dumped. If data from message data fields is to be dumped, this parameter specifies one or more numbers that positionally identify the data fields to be dumped.

The system objects indicated by system pointers are completely dumped. The data in a space object, indicated by a space pointer, is dumped starting from the offset indicated by the space pointer for the length indicated in the field description. The standard job dump can also be requested. Dumps are taken as part of system default actions if escape messages are not monitored by the program to which they were sent.

***JOB:** This value has the same effect as specifying JOB(*) and OUTPUT(*PRINT) on the Display Job (DSPJOB) command. See the DSPJOB command description for more information.

***JOBDMF:** The data areas of the job are dumped as specified by the DMPJOB command.

*JOBDMF can be specified by itself, with *JOB, *JOBINT, or a list of message data field numbers.


***JOBINT:** The internal machine data structures related to the machine process in which the job is running are dumped to the machine error log as specified by the DMPJOBINT command. *JOBINT can be specified by itself, with *JOBDMF, *JOB, or with a list of message data field numbers.

message-data-field-number: Specify the numbers of the message data fields that identify the data that is dumped if this escape message is sent but not monitored. Up to 99 data field numbers can be specified in the list; additionally, the list can contain the values *JOB, *JOBINT, and *JOBDMF.

***NONE:** There is no dump list for this message. No dump occurs.

Notes:

1. If any of these values are specified for DMPLST, *JOB is assumed to be part of the values. For example, DMPLST (1 2 *JOBDMF) gives the same result as DMPLST(*JOB 1 2 *JOBDMF).

2. Values specified for the DMPLST parameter may be overridden by the QSRVDMF system value. More information is in the chapter on system values in the Work Management  book.

3. The program receiving the unmonitored message either must have a name starting with 'Q' or the message severity must be 50 or higher.

4. The user of the job in which the dump is specified must be authorized to the dump command requested on this parameter.

LVL Specifies the level identifier of the message description being defined. The level identifier is made up of the date on which the message is defined and a 2-digit number that makes the identifier unique.

***CURRENT 1:** The current date and a 1 are used as the first and second parts of the message description level identifier.

creation-date level-number: Specify the date on which the message is being defined, and enter a 2-digit value (ranging from 1 through 99) that makes the level identifier of the message description unique. The date must be specified in the format defined by the system values QDATFMT and, if separators are used, QDATSEP.

ALROPT

Specifies the alert option associated with messages sent to the QHST log message queue.

***NO:** No alert is sent.

Element 1: When an Alert is Sent

***IMMED:** An alert is sent immediately, simultaneous with sending the message to QHST.

***UNATTEND:** An alert is sent immediately only if the system is running in unattended mode.

The system is considered to be unattended if the Alert Status (ALRSTS) network attribute is set to *UNATTEND.

***DEFER:** The alert is sent after local problem analysis.

*DEFER should be specified only for messages against which a problem analysis can be run. An alert is sent at the first exit from problem analysis for the problem referred to by the message.

All alerts set to *DEFER are treated as *IMMED if:

- The system is running in unattended mode
- An error log ID is not available for a problem that might be resolved using problem analysis
- The LOGPRB parameter for the message is set to *NO

Element 2: Message Data Field Format Numbers

***NONE:** No message data field format number is passed with the alert identifier.

format-number: Specify the message data field format number that is passed with the alert identifier.

LOGPRB

Specifies, for IBM-supplied messages, whether an entry is put in the problem log. If there is an error log ID for the message and *YES is specified for this parameter, the user can call for problem analysis by pressing the F14 key from the system operator message queue display (Display Messages).

***NO:** An entry is not put in the problem log.

***YES:** An entry is put in the problem log if there is a error log ID associated with the message.

CCSID

Specifies the coded character set identification (CCSID) that the text supplied on the MSG and SECLVL parameters are in. If the message file that this message description is being added to is not 65534 or 65535, the text supplied is converted from the CCSID specified to the CCSID of the message file. Otherwise, the text is not converted but the CCSID is saved in case a conversion is needed during a retrieve or display type function. For more information about the message handler and its use of CCSIDs, see the Globalization topic in the Information Center.

***JOB** The text for this message description is assumed to be in the CCSID of the job running this command.

***HEX:** The text for this message description is not converted and is tagged 65535.

coded-character-set-identifier: Specify the CCSID you want the text to be considered in. Valid values range from 1 through 65535. See the Globalization topic in the Information Center for a list of valid CCSID values. Only CCSIDs that a job can be changed to are accepted.

Examples for ADDMSGD

Example 1: Defining a Message

```
ADDMSGD MSGID(UIN0115) MSGF(INV)
MSG('Enter the name of user''s department')
SECLVL('Valid department names are:
&B X12 &B X13 &B X14')
TYPE(*CHAR) LEN(3) DFT('ZZZ')
```

This command defines a message and stores it in a file named INV under the identifier UIN0115. The message supplies second-level message text by using the &B formatting character to show the three valid department names (X12, X13, and X14) each on a separate line. The reply requires validity checking so that a valid reply can only be a 3-character identifier. A default reply of ZZZ is also provided.

Example 2: Defining a Message Description

```
ADDMSGD MSGID(UPY0047)
MSGF(PAYLIB/TIMECARD)
MSG('For the week of &1, &2 time
cards were processed. Do you have more?')
FMT((*CHAR 8) (*CHAR 3))
TYPE(*ALPHA) LEN(1) VALUES(N Y)
SPCVAL((YES Y)(NO N)) DFT(N)
```

This command defines a message description that is stored in the TIMECARD message file in the PAYLIB library. The program that processes the time cards can send a message (as an inquiry type message) telling how many time cards (in &2) have been processed for the week (specified in &1). To send this message to a user via a message queue, the program must use the SNDPGMMSG or SNDUSRMSG commands. In this example, the command specifies:

- The message identifier of this message (UPY0047)
- The file (TIMECARD) that contains this message
- The time card date in 8 characters (such as 09/15/88); this must be the first value in the MSGDTA parameter
- The number of time cards in no more than 3 digits (such as 125)

If a reply of YES is sent, it is accepted as a Y (SPCVAL parameter). If NO is sent, it is accepted as an N. If neither YES nor NO is sent, the reply is checked for validity by the TYPE, LEN, and VALUES parameters. If the user chooses, no reply is sent and the default reply (N) is assumed.

Example 3: Defining an Escape Message

```
ADDMSGD MSGID(UPY1234)
MSGF(PAYLIB/TIMECARD)
MSG('Tax for employee &1 exceeds
gross salary.') SEV(75)
FMT((*CHAR 6)(*DEC 9 2)(*CHAR 8))
DFTPGM(PAYLIB/BADTAX)
DMPLST(1 2 3 *JOB)
```

This command defines an escape message. The sender of the message passes three data values, the first of which (employee serial number) is used as replacement text in the message. If this message is sent as an escape message and the program to which the message is sent does not monitor for message UPY1234, default system action is taken. This includes dumping the three data values that were passed and the job structure. After the dump is taken, program BADTAX is called.

See the Monitor Message (MONMSG) command for more about monitoring for messages.

Error Messages for ADDMSGD

*ESCAPE Messages

CPF2401

Not authorized to library &1.

CPF2407

Message file &1 in &2 not found.

CPF2411

Not authorized to message file &1 in &2.

CPF2412

Message ID &1 already exists in message file &2 in &3.

CPF2430

Message description not added to message file

CPF2461

Message file &1 could not be extended.

CPF2483

Message file currently in use.

CPF2510

Message file &1 in &2 logically damaged.

CPF9830

Cannot assign library &1.

CPF9838

User profile storage limit exceeded.

ADDMFS (Add Mounted File System) Command Description

ADDMFS Command syntax diagram

Purpose

The Add Mounted File System (ADDMFS) command makes the objects in a file system accessible to the integrated file system (IFS) name space. The file system to be made accessible can be either a user-defined file system (*UDFS) on the local system, a remote file system accessed via a local Network File System client (*NFS), or a local or remote NetWare file system (*NETWARE). The directory that is the destination for the mount (the MNTOVRDIR parameter) must exist.

This command can also be issued using the following alternative command name:

- MOUNT

For more information about Network File System commands, see the OS/400 Network File System

Support  book.

Restrictions:

1. You must have *IOSYSCFG special authority to use this command.
2. If you are mounting a user-defined file system or a Network File System, *READ authority to the file system being mounted is required.

3. If you are mounting a NetWare file system, *EXECUTE authority to the file system being mounted is required.
4. You must have *W authority to the directory being mounted over.

Required Parameters

TYPE Specifies the type of file system being mounted. The type of mount determines the correct form for the MFS parameter.

***NFS:** The file system specified for the MFS parameter is a Network File System. The MFS parameter must be of the form *hostname:pathname* where *hostname* can either be the name of a system or an IP address, and *pathname* must be an absolute path name.

***UDFS:** The file system specified for the MFS parameter is a user-defined file system. ➤ It must be in one of the following two forms:

- */dev/QASPXX/udfsname.udfs*, where *XX* is one of the valid system or basic user auxiliary storage pool (ASP) numbers on the system, and *udfsname* is the name of the user-defined file system. All other parts of the name must appear as in the example above.
- */dev/aspname/udfsname.udfs*, where *aspname* is one of the valid independent ASP names on the system, and *udfsname* is the name of the user-defined file system. All other parts of the name must appear as in the example above.

The name part of the path must be unique within the specified *QASPXX* or *aspname* directory. ⏪

***NETWARE:** The file system specified for the MFS parameter is a NetWare file system. The MFS parameter must be one of the following forms:

- *server/volume:pathname*, where *pathname* is optional.
- NetWare Directory Services (NDS**) context to a volume, a directory map object to mount, or an alias to a volume or directory map object. The NDS context can be a distinguished or relative context. If a relative context is specified the current context for the job is searched, and if it is not found the default system context is searched. If a context to a volume or an alias to a volume is specified an optional directory path may also be specified.

Note:

On the MFS parameter, if you specify a relative context that contains no dots and no path name after the colon, you must be sure to quote the parameter value when prompting on the command. The command analyzer may interpret the MFS value as a label and remove the trailing colon.

See the *Examples* section for examples of these forms.

MFS Specifies the path name of the file system to be mounted. It can be the path to a local Block Special File (*BLKSF), a remote NFS path name, or the path of a NetWare file system. See the TYPE parameter to determine the correct format for the MFS parameter.

MNTOVRDIR

Specifies the path name of the existing directory that the file system will be mounted over. This directory gets 'covered' by the mounted file system. This directory must exist.

Multiple file systems can be mounted over the same directory, one on top of the other. However, only the topmost mounted file system is accessible, and the file systems must later be in the opposite order from which they were mounted (last-in first-out order).

Optional Parameters

OPTIONS

The options list contains a character string of mount options. The keywords are separated by

commas. For some keywords, an equal '=' and a value follow the keyword. If a keyword is not specified, the default value for that option will be used. The options list may contain spaces.

***DFT:** The default value for the options string for the mount of a Network File System (*NFS) is:

```
'rw,suid,rsize=8096,wsize=8096,timeo=20,retrans=5,  
acregmin=30,acregmax=60,acdirmin=30,acdirmax=60,hard'
```

The default value for the options string for the mount of a user defined file system (*UDFS) is:

```
'rw'
```

The default value for the options string for the mount of a NetWare file system (*NETWARE) is:

```
'rw,acregmax=60,acdirmax=60'
```

For the mount of a Network File System, all of the following options are valid. For the mount of a user defined file system, only the protection option (*ro* or *rw*) is valid. For the mount of a NetWare file system, the protection (*ro* or *rw*), *acregmax*, *acdirmax*, *noac*, and *nocto* options are valid. If options that are not valid for the file system type you are mounting are specified, they are ignored.

options-list: The following are the available options keywords and their descriptions:

rwlrw This option specifies the protection for the mounted file system. Either *ro* (read-only) or *rw* (read-write) may be specified. If neither is specified, *rw* is assumed.

suidlnosuid

For the mount of a Network File System, if *suid* is specified, setuid execution is allowed. If neither or *nosuid* is specified, setuid execution is not allowed.

hardsoft

For the mount of a Network File System, specifies whether NFS** file systems are hard or soft mounted. Hard mounted means that operations on them are retried until they are acknowledged by the server. Soft mounted means that a timeout error is returned if a remote operation fails the number of times specified on the *retrans* parameter. If neither is specified, *hard* is assumed.

rsize=n

For the mount of a Network File System, specifies the size of the read buffer in bytes. The read buffer is used for data transfer between the NFS client and the remote NFS server on an NFS read request. The allowed range is 512 to 8096. If *rsize* is not specified, the default value of 8096 is assumed. For better performance, the read buffer should be a multiple of the the application buffer size.

wsize=n

For the mount of a Network File System, specifies the size of the write buffer in bytes. The write buffer is used for data transfer between the NFS client and the remote NFS server on an NFS write request. The allowed range is 512 to 8096. If *wsize* is not specified, the default value of 8096 is assumed. For better performance, the write buffer should be a multiple of the the application buffer size.

timeo=n

For the mount of a Network File System, specifies the amount of time, in tenths of seconds, to wait for the client to respond on each try. The allowed range is 0 to 10000. If *timeo* is not specified, the default value of 20 tenths of a second (2 seconds) is assumed.

retry=n

For the mount of a Network File System, specifies the number of times to retry the mount operation. The allowed range is 0 to 10000. If *retry* is not specified, the default value of 5 retransmission attempts is assumed.

retrans=n

For the mount of a Network File System, specifies the number of times to retry the

transmission to the client. The allowed range is 0 to 10. If *retrans* is not specified, the default value of 5 retransmission attempts is assumed.

acregmin=n

For the mount of a Network File System, specifies the minimum number of seconds to hold locally stored file attributes after file updates. The allowed range is 1 to 3600. If *acregmin* is not specified, the default value of 30 seconds is assumed.

acregmax=n

For the mount of a Network File System or a NetWare file system, specifies the maximum number of seconds to hold locally stored file attributes after file updates. The allowed range is 1 to 2,000,000,000. If *acregmax* is not specified, the default value of 60 seconds is assumed.

acdirmin=n

For the mount of a Network File System, specifies the minimum number of seconds to hold locally stored directory attributes after a directory update. The allowed range is 1 to 3600. If *acdirmin* is not specified, the default value of 30 seconds is assumed.

acdirmax=n

For the mount of a Network File System or a NetWare file system, specifies the maximum number of seconds to hold locally stored directory attributes after a directory update. The allowed range is 1 to 2,000,000,000. If *acdirmax* is not specified, the default value of 60 seconds is assumed.

nocto For the mount of a Network File System or a NetWare file system, specifies whether to force the refresh of remote attributes when opening a file. If this keyword is present, attributes are not refreshed from the server when opening a file, and changes are not sent to the server on the last close. If *nocto* is not present, the default value of no suppression is assumed.

noac For the mount of a Network File System or a NetWare file system, specifies whether to suppress local storage of attributes and names. If this keyword is present, local storage of attributes and names is suppressed. If *noac* is not present, the default value of no suppression is assumed. If you specify *noac*, values specified for *acregmin*, *acregmax*, *acdirmin*, and *acdirmax* may be specified but are not used.

CCSID

Specifies, for Network File Systems, a pair of coded character set identifiers (CCSIDs) to identify a specific character representation to be used. The first CCSID specifies what encoding scheme should be assumed for data files on the remote system. The second CCSID specifies what encoding scheme should be assumed for path names on the remote system.

Element 1: Data File CCSID

***BINARY:** No conversion is used.

***ASCII:** The ASCII equivalent of the coded character set identifier (CCSID) obtained from the default job CCSID is used.

***JOBCCSID:** The coded character set identifier (CCSID) from the default job CCSID is used.

data-coded-character-set-identifier: A coded character set identifier (CCSID) to be assumed for data files on the remote system. Valid values are 1 through 65533.

Element 2: Path Name CCSID

***ASCII:** The ASCII equivalent of the coded character set identifier (CCSID) obtained from the default job CCSID is used.

***JOBCCSID:** The coded character set identifier (CCSID) from the default job CCSID is used.

path-name-coded-character-set-identifier: A coded character set identifier (CCSID) to be assumed for path names on the remote system. Valid values are 1 through 65533. Only CCSIDs that can be converted into ISO 10646(13488) are supported. See Globalization topic in the Information Center for a list of supported conversions.

CODEPAGE

Specifies, for Network File Systems, a pair of code pages. The first code page specifies what code page should be assumed for data files on the remote system. The second code page specifies what code page should be assumed for path names on the remote system.

Note: This parameter is replaced by CCSID but the CODEPAGE parameter can still be used. However, because this parameter may be removed in a future release, whenever possible use the CCSID parameter.

Element 1: Data file code page

Note: You should specify a code page that has the same number of bytes per character as the original data.

***BINARY:** No conversion is used.

***ASCII:** The ASCII equivalent of the code page obtained from the default job coded character set identifier (CCSID) associated with the current job is used.

***JOBCCSID:** The code page obtained from the default job coded character set identifier (CCSID) associated with the current job is used.

data-code-page: A code page to be assumed for data files on the remote system. Only code pages that correspond to single- or double-byte encoding schemes are supported. Code pages that correspond to mixed-byte encoding schemes are not supported.

Element 2: Path name code page

***ASCII:** The ASCII equivalent of the code page obtained from the default job coded character set identifier (CCSID) associated with the current job is used.

***JOBCCSID:** The code page from the default job CCSID associated with the current job is used.

path-name-code-page: A code page to be assumed for path names on the remote system. Only code pages whose CCSIDs are convertible into ISO 10646(13488) are supported. See the Globalization topic in the Information Center for a list of supported conversions.

Examples for ADDMFS

Example 1: Mounting a User-Defined File System

```
ADDMFS TYPE(*UDFS) MFS('/DEV/QASP03/PROD1')
      MNTOVRDIR('DIRB')
```

This command mounts a user defined file system PROD1 over the directory, DIRB. It uses the defaults for the other parameters.

Example 2: Mounting a Network File System

```
ADDMFS TYPE(*NFS) MFS('RAINFALL:/QSYS.LIB/RAY.LIB')
MNTOVRDIR('/mystuff')
```

This command mounts the */qsys.lib/ray.lib* file system from the remote system RAINFALL into the directory */mystuff*.

Example 3: Mounting a Network File System with OPTIONS

```
ADDMFS TYPE(*NFS) MFS('RAINFALL:/QSYS.LIB/RAY.LIB')
MNTOVRDIR('/mystuff')
OPTIONS('ro,nosuid,rsize=256, retrans=10')
CODEPAGE(*ASCII *JOBCCSID)

CCSID(*ASCII *JOBCCSID)
```

This command mounts the */qsys.lib/ray.lib* file system from the remote system RAINFALL into the directory */mystuff*. In addition it specifies to mount as read-only, not allow setuid execution, set the read buffer to 256 bytes, and the retransmission attempts to 10. The job CCSID is used to determine the coded character set identifier to use for remote path names.

Example 4: Mounting a NetWare File System with OPTIONS

```
ADDMFS TYPE(*NETWARE)
MFS('RCHNWSVR1/LOTUS:LOTSUITE/SMARTCTR')
MNTOVRDIR('/temp1')
OPTIONS('ro,agregmax=120')
```

This command mounts the NetWare directory *LOTSUITE/SMARTCTR* contained in the volume *LOTUS* that resides on server *RCHNWSVR1* over the directory */temp1*. In addition it specifies to mount as read-only, sets the maximum time to store file attributes locally to 120 seconds.

Example 5: Mounting using a NetWare Directory Services Context

Following are several examples of mounting a NetWare file system using NetWare Directory Services (NDS) contexts.

```
ADDMFS TYPE(*NETWARE) MFS('.LOTUS_VOL.ROCHESTER.IBM')
MNTOVRDIR('/temp1')
```

This command mounts NDS volume *LOTUS_VOL* using a distinguished context, over the directory */temp1*.

```
ADDMFS TYPE(*NETWARE)
MFS('CN=LOTUS_VOL.OU=ROCHESTER:LOTSUITE/SMARTCTR')
MNTOVRDIR('/temp1')
```

This command mounts path *LOTSUITE/SMARTCTR* on NDS volume *LOTUS* using a relative path and fully qualified names, over the directory */temp1*.

```
ADDMFS TYPE(*NETWARE)
MFS('.CN=LOTUSMAP.OU=ROCHESTER.O=IBM')
MNTOVRDIR('/temp1')
```

This command mounts a directory map object using a distinguished context and fully qualified names, over the directory */temp1*.

Error messages for ADDMFS

None

MOUNT (Add Mounted File System) Command

MOUNT syntax diagram

MOUNT Command	For the description of the MOUNT command, see the ADDMFS (Add Mounted File System) command description.
---------------	---

ADDNTWAUTE (Add NetWare Authentication Entry) Command Description

ADDNTWAUTE Command syntax diagram

Purpose

The Add NetWare Authentication Entry (ADDNTWAUTE) command adds authentication information for a server to a user profile. The information specifies how the user signs on to the server. This information is used to start authenticated connections to servers. An authenticated connection to a server is required to issue requests to the server. If an authenticated connection does not exist, the system attempts to start a connection using data stored in the authentication entries.

Required Parameters

SVRTYPE

Specifies the server type of the authentication entry.

***NETWARE3:** The entry is for a NetWare 3.x server.

***NDS:** The entry is for a NetWare Directory Services tree.

PASSWORD

Specifies the password used to authenticate the user to the server.

***NONE:** No password is needed to verify authority.

***STRNTWCNN:** No password is stored as part of the authentication entry. The password must be provided as part of a Start NetWare Connection (STRNTWCNN) request prior to issuing any requests to the server.

'password': Use the specified password.

Optional Parameters

NDSTREE

For server type *NDS, specifies the name of the NetWare Directory Services tree for which the authentication entry is being added.

SERVER

For server type *NETWARE3, specifies the name of the server for which the authentication entry is being added.

USRPRF

Specifies the user profile to which the authentication entry is to be added.

***CURRENT:** Use the current user profile.

user-profile-name: Add the entry to the specified user profile. The user profile must be the current user profile, or the user must have *USE and *OBJMGT authority to the user profile, and *SECADM special authority.

NDSCTX

For server type *NDS, specifies the NetWare Directory Services context in which the user name is defined. The NDS context and NetWare user name are combined to form the distinguished name for the user in the NDS tree.

NTWUSER

Specifies the NetWare user name used to authenticate the user to the server.

***USRPRF:** The NetWare user name is the same as the name of the user profile (USRPRF parameter) to which the entry is being added.

'NetWare-user-name': Use the specified user name.

Examples for ADDNTWAUTE

Adding a *NETWARE3 server authentication entry

```
ADDNTWAUTE SVRTYPE(*NETWARE3) SERVER(DEPT480) USRPRF(JOHN)
           NTWUSER(JOHNM) PASSWORD(XXXX)
```

This command adds an authentication entry for the NetWare 3.x named DEPT480 to the user profile JOHN. The NetWare user name is JOHNM, and the password is XXXX.

Adding a *NDS authentication entry

```
ADDNTWAUTE SVRTYPE(*NDS) NDSTREE(IBMTRREE)
           NDSCTX('OU=PROG.OU=ROCH.O=IBM') PASSWORD(*STRNTCNN)
```

This command adds an authentication entry for NDS tree IBMTRREE to the current user profile. The user name is the same as the current user profile. The full distinguished name for the user (formed by combining the NDSCTX and NTWUSER parameters, and assuming the current user profile name is JOHN) is '.CN=JOHN.OU=PROG.OU=ROCH.O=IBM'. The Start NetWare Connection (STRNTWCNN) command must be used, with the correct password specified on the PASSWORD parameter, before requests can be sent to the server.

Error messages for ADDNTWAUTE

*ESCAPE Messages

FPE0217

Authentication entry not added due to errors.

ADDNETJOB (Add Network Job Entry) Command Description

ADDNETJOB Command syntax diagram


Purpose

The Add Network Job Entry (ADDNETJOB) command adds a network job entry to the network job table on the system. The network job entry is used to determine the action that is taken when an input stream is sent to a user on this system by using the Submit Network Job (SBMNETJOB) command. This entry determines whether the input stream is automatically submitted, placed on the queue of network files for a user, or rejected. The entry also specifies the user profile that is used for checking the authority to the job description referenced by the input stream. There must be one entry for each user or distribution group who intends to submit jobs to this system.

Note: There is a network attribute, JOBACN (Job Action), that provides overall control of network job submission. Its value must be *SEARCH before the network job table is searched for an action. If the network attribute is *REJECT, all incoming jobs are rejected. If the network attribute is *FILE, all incoming

network jobs are saved in the user's queue of network files regardless of any network job entry. The network attribute can be changed with the Change Network Attributes (CHGNETA) command.

Each network job entry is identified by the two-part user ID of the sender. When an input stream arrives, the user ID of the sending user is used to find a network job entry. If no entry is found, the second part of the user ID is used to find an entry, using *ANY for the first part. If this search fails, a search is made using *ANY for both parts of the user ID. If no entry is found, the job is rejected.

For additional information on the network job table, refer to the SNA Distribution Services  book.

Restrictions:

1. This command is shipped with public *EXCLUDE authority.
2. The user must have *ALLOBJ (all object) authority.
3. The internal value for a node identifier may differ from the characters shown by the ADDNETJOB command depending on the type of work station (language) being used. If the byte-string value specified for the FROMUSRID command parameter does not match the rules for an internal node identifier value, or if it does not match the internal value for any defined node (ignoring case differences), an error may be reported.

Required Parameters

FROMUSRID

Specifies the two-part user ID of the user who submits an input stream to this system. Any input streams received from the user are handled as specified in this network job entry. Both parts of the user ID are required. A special value of *ANY can be entered for the first part or for both parts of the user ID.

Note: Depending on the type of work station being used, the internal value for a user identifier may differ from the characters shown by the Display Network Job Entry (DSPNETJOB) command. If the byte-string value specified for the FROMUSRID parameter does not match the rules for an internal user identifier value, or if it does not match the internal value for any enrolled user, an error may be reported.

ACTION

Specifies the action that is taken for the input stream controlled by this entry if the network attribute JOBACN is *SEARCH.

***REJECT:** The input stream is rejected.

***FILE:** The input stream is placed on the queue of network files for the user to whom the input stream is sent.

***SUBMIT:** The input stream is submitted to a batch job queue. The user profile specified in the network job entry is used to check for the required authority to the job queues that are used and to the job descriptions specified in the input stream.

Optional Parameters

SBMUSER

Specifies the user profile name under which jobs are submitted. This user profile name is used to check the authority to the job queues and job descriptions specified in the input stream. The value specified for this parameter will be effective if ACTION(*SUBMIT) is specified either on this command or on the Change Network Job Entry (CHGNETJOB) command.

QUSER: The IBM-supplied user profile QUSER is used to submit the jobs.

user-profile: Specify the name of the user profile that is used to submit the jobs.

MSGQ

Specifies the qualified name of the message queue to which messages are sent.

Note: The message sent to the message queue specified here notifies the recipient that the input stream arrived and whether it was submitted, placed on the user's queue of network files, or rejected. A message is also sent to the history log (QHST) when an input stream arrives.

***USRPRF:** The message queue of the user profile to whom the job was sent is used. This user is specified on the TOUSRID parameter of the SBMNETJOB command; this may or may not be the same user as is specified on the SBMUSER parameter of this command.

***NONE:** No message is sent to a user; however, a message is sent to the history log (QHST).

The name of the message queue can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

message-queue-name: Specify the qualified name of the message queue that is used to receive messages.

JOBQ Specifies the job queue on which the job entries are placed. A job entry is placed on this queue for each job in the input stream that has JOBQ(*RDR) specified on the Batch Job (BCHJOB) command. If *RDR is not specified on the BCHJOB command, the job queue specified on the BCHJOB command or in the job description is used. (The job queue for each job in the input stream can be different.) This parameter is valid only if ACTION(*SUBMIT) is specified on this command, in the existing network job entry, or in a subsequent Change Network Job Entry (CHGNETJOBE) command.

The name of the job queue can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

QBATCH: The job entry uses job queue QBATCH. The library list (*LIBL) is used to locate the named job queue; if one is found, the name of the library in which the job queue was found is placed in the job entry.

job-queue-name: Specify the name of the job queue.

Examples for ADDNETJOBE

Example 1: Submitting Input Streams Automatically

```
ADDNETJOBE FROMUSRID (JOHN SMITH)
ACTION(*SUBMIT)
SBMUSER(ANDERSON)
JOBQ(QGPL/QPGMRL)
```

This command adds a network job entry that is used to determine the action that is taken for any input stream received from a user with a user ID of JOHN SMITH. The input streams are submitted automatically. User profile ANDERSON is used to check the authority to the job queues and job descriptions specified in the input stream. Messages are sent to the message queue specified in the user profile of the user to whom the input stream was sent. If no job queue is specified in either the received // BCHJOB command or the referenced job description, the jobs are placed on job queue QPGMRL in the QGPL library.

Example 2: Sending Messages to Specific Message Queue

```
ADDNETJOBE FROMUSRID(*ANY JONES) ACTION(*FILE)
MSGQ(BROWN) SBMUSER(ANDERSON)
```

This command adds a network job entry that is used to determine the action taken for any input stream received from any user with JONES as the second part of the user ID for whom there is not a specific network job entry. The input stream is placed on the queue of received files for the user to whom the job was sent, and a message is sent to message queue BROWN.

Error messages for ADDNETJOBE

*ESCAPE Messages

CPF8050

Network job table could not be accessed.

CPF8051

*ANY not correct for second part of user ID.

CPF8052

Network job entry &1 &2 not added.

CPF9040


Wrong characters used in User ID or address, or List identifier &1 &2.

ADDNWSSTGL (Add Network Server Storage Link) Command Description

ADDNWSSTGL Command syntax diagram

Purpose

The Add Network Server Storage Link (ADDNWSSTGL) command is used to add a network storage space link to a network server description. Up to 16 network server storage spaces can be linked to a *WINDOWSNT network server description using standard static linking. ➤ When linking to a *WINDOWSNT server with Windows 2000 or later installed, an additional 32 network storage spaces can be linked. 16 of the dynamic links are reserved for shared storage spaces (ACCESS *SHRUPD) used for clustering Windows servers. When linking shared storage spaces to a Windows server in a Windows cluster, a special quorum resource disk (FORMAT *NTFSQR) must be linked before linking other shared storage spaces. When linking to a *GUEST server 64 storage links of any type are allowed. ⏪

More information about using this command is in Communications Configuration  book.

Required Parameters

NWSSTG

Specifies the name of the network server storage space.

NWSD

Specifies the name of the network server description to which this link is added.

Optional Parameters



DYNAMIC

Specifies if this storage is to be linked as dynamic storage using the next available location, or linked as static storage using the drive sequence number specified in the DRVSEQNBR parameter.

***NO:** The storage space is linked in the standard linking method using the DRVSEQNBR parameter as the value to link the storage. The NWSD must be varied off to perform this link.

***YES:** The storage space is linked using dynamic linking and will be linked in the next available sequence number of the dynamic linking area of the NWSD.

Note:

This parameter is valid only if the network server description (NWSD parameter) was created as TYPE(*WINDOWSNT). *YES can only be specified when Windows 2000  or later is installed on the network server. 


DRVSEQNBR

Specifies the order the network storage spaces are presented to the server. Each storage space must be given a unique sequence number.


Note:

This parameter is valid only if the network server description (NWSD parameter) was created as TYPE(*WINDOWSNT).

***CALC:** The system will assign the sequence number for network servers that were created as TYPE(*WINDOWSNT). The system assigns the lowest available sequence number.

 ***QR:** The storage space is linked as the quorum resource for the clustered Windows network server description.

Note:

Only one storage space may be linked as the quorum resource for any server, but the storage space may be linked to multiple servers joined together as a cluster. This value can be specified only if the network server description (NWSD parameter) was created as TYPE(*WINDOWSNT) and when Windows 2000 or later is installed. 

sequence-number: Specify a number in the range 1 through 64. If a non-sequential number is specified, the number may not match the logical unit number on Windows and unexpected results may occur on drive assignments and applications. For static links (DYNAMIC(*NO)) to Windows servers the valid sequence range is 3 through 18. For non-cluster dynamic links (DYNAMIC(*YES)) to Windows servers the valid sequence range is 1 through 16. For cluster links (DYNAMIC(*YES) ACCESS(*SHRUPD)) the valid range is 1 through 15. For any link to a *GUEST server the valid range is 1 through 64.

ACCESS

Specifies the server's access method to the storage space.

Note:

This parameter is valid only if the network server description (NWSD parameter) was created as TYPE(*WINDOWSNT) and is installed with Windows 2000 or later installed . Access modes of *UPDATE, *READ, or *SHRUPD can be linked as either static storage or dynamic storage. Windows NT 4.0 servers can only link storage as static storage using *UPDATE access mode.

***UPDATE:** The storage space is being linked to allow an NWSD to have an exclusive read/write lock to this storage space. No other NWSDs can link to this storage space while the storage is linked as *UPDATE.

***READ:** The storage space is being linked to allow an NWSD to have a read only lock to this storage space. Write requests to the storage space will not be allowed from this server. Other NWSDs have the option to link to this same storage space as *READ. **Note:** This value is valid for *WINDOWSNT servers when Windows .NET (*WIN2002) or later is installed and the disk is not linked to a *GUEST server. All servers linking the storage space must have *SHRRD as their ACCESS. <<

>> ***SHRUPD:** The storage space is being linked to allow an NWSD to have a shared read/write lock to this storage space.

Note:

This value is valid for *WINDOWSNT servers when DRVSEQNBR is *CALC or *QR and DYNAMIC is *YES and the disk is not linked to a *GUEST server. This value is valid for *GUEST servers at any DRVSEQNBR as long as the disk is not linked to a *WINDOWSNT server.

TYPE Describes the type of network server description to which this link is added.

Note:

This parameter is present only for compatibility with previous releases. The value specified is not syntax checked and no verification is done to ensure that the network server description matches the specified TYPE value.

>> Examples for ADDNWSSTGL

Example 1:

```
ADDNWSSTGL NWSSTG(PARTS) NWSD(WNTSVR)
  DRVSEQNBR(3)
ADDNWSSTGL NWSSTG(DATA) NWSD(WNTSVR)
  DRVSEQNBR(*CALC)
```

These commands link storage space PARTS to the Windows NT Server at drive sequence 3, and storage space DATA at the next available sequence number.

Example 2:

```
ADDNWSSTGL NWSSTG(DATAUPD) NWS(D2KSVR)
DYNAMIC(*YES)
```

This command dynamically links the storage to the Windows 2000 server to the next available dynamic storage sequence.

Example 3:

```
ADDNWSSTGL NWSSTG(QUORUM) NWS(DNETSVR1) DYNAMIC(*YES) ACCESS(*SHRUPD) DRVSEQNBR(*QR)
ADDNWSSTGL NWSSTG(QUORUM) NWS(DNETSVR2) DYNAMIC(*YES) ACCESS(*SHRUPD) DRVSEQNBR(*QR)
ADDNWSSTGL NWSSTG(DATASHR) NWS(DNETSVR1) DYNAMIC(*YES) ACCESS(*SHRUPD) DRVSEQNBR(*CALC)
ADDNWSSTGL NWSSTG(DATASHR) NWS(DNETSVR2) DYNAMIC(*YES) ACCESS(*SHRUPD) DRVSEQNBR(*CALC)
```

These commands allow two Windows 2002 (Dot Net) servers DNETSVR1 and DNETSVR2 to share storage space DATASHR with read/write access.

Example 4:

```
ADDNWSSTGL NWSSTG(DATASRC) NWS(DNETSVR) DYNAMIC(*YES) ACCESS(*SHRUPD) DRVSEQNBR(*CALC)
ADDNWSSTGL NWSSTG(DATASRC) NWS(LINUX1) DYNAMIC(*YES) ACCESS(*SHRUPD) DRVSEQNBR(1)
ADDNWSSTGL NWSSTG(DATASRC) NWS(LINUX2) DYNAMIC(*YES) ACCESS(*SHRUPD) DRVSEQNBR(*CALC)
```

These commands allow two LINUX servers, LINUX1 and LINUX2, and a Windows server, DNETSVR, to share storage space DATASRC with read access. <<

Error messages for ADDNWSSTGL

*ESCAPE Messages

CPF26BA

Add network server storage link command failed.

ADDNETBLE (Add Network Table Entry) Command Description

ADDNETBLE Command syntax diagram

Purpose

The Add Network Table Entry (ADDNETBLE) command is used to add a network entry to the network table. You can use the network table to manage a list of your networks and their associated Internet addresses.

Restriction: You must have system configuration (*IOSYSCFG) special authority to use this command.

Required Parameters

NETWORK

Specifies the name of the network to be added to the table.

Note:

The combination of the values on the NETWORK and INTNETADR parameters must be unique.

INTNETADR

Specifies the Internet address of the network. Internet addresses are expressed in the decimal form

`nnn.nnn.nnn.nnn`

where *nnn* is a number ranging from 0 through 255.

Optional Parameters

ALIAS Specifies the alternate name for the network. You can specify a maximum of 4 aliases. No checking is done to ensure that an alias is unique.

***NONE:** The network has no alternate name.

alias: Specify an alternate network name.

TEXT Specifies the text that briefly describes the network entry. More information on this parameter is in Commonly used parameters.

***BLANK:** Text is not specified.

'network-description': Specify no more than 50 characters of text, enclosed in apostrophes.

Example for ADDNETTBLE

```
ADDNETTBLE NETWORK(NETONE) INTNETADR(9.5.0.0)
```

This command adds an entry for the network NETONE to the network table. The Internet address for NETONE is 9.5.0.0.

Error messages for ADDNETTBLE

*ESCAPE Messages

TCP1901

Internet address &1 not valid.

TCP290C

Network entry already exists in table. Entry was not added.

TCP2916

Network entry contains characters that are not valid. Entry was not added.

ADDNCK (Add Nickname) Command Description

ADDNCK Command syntax diagram

Purpose

The Add Nickname (ADDNCK) command is used to add a nickname to the system distribution directory. The nickname must be unique if it is a public nickname. The nickname must be unique only for the owner if it is a private nickname.

A **nickname** is a short version of either a directory entry or a distribution list name. You can specify a nickname instead of the full directory entry or distribution list name on many OfficeVision displays. More

information about nicknames is in the SNA Distribution Services  book.

Required Parameters

NCK Specifies the nickname to be added and the ability of users to access the nickname.

Element 1: Nickname


nickname: Specify the nickname you are adding.

Element 2: Nickname Access

***PRIVATE:** The nickname cannot be shared with other users. It can be accessed and changed only by the owner.

***PUBLIC:** The nickname can be shared with other users. It can be accessed by any user on the local system, but it can be changed only by a user with security administrator (*SECADM) authority or by the owner.

USRID

Specifies the two-part user identifier for which a user nickname is being added. Both the user ID and address elements must be specified. More information about specifying the user ID and address is in the SNA Distribution Services  book.

Note:

This parameter cannot be specified when the LSTID parameter is specified.

Element 1: User ID

user-ID: Specify the user ID for this nickname. A maximum of 8 characters can be specified.

Element 2: User Address

user-address: Specify the address for this nickname. A maximum of 8 characters can be specified.

LSTID Specifies the two-part list identifier of the distribution list for which a list nickname is being added. Both the list identifier and qualifier elements must be specified.

Note:

This parameter cannot be specified when the USRID parameter is specified.

Element 1: List Identifier

list-ID: Specify the list identifier (ID) of the distribution list.

Element 2: List Qualifier

list-ID-qualifier: Specify the list ID qualifier of the distribution list.

Note:

The distribution list identifier has two parts, the ID and the qualifier, separated by at least one space. If lowercase characters are specified, the system changes them to uppercase.

The naming rules for the two-part list ID are identical to the rules for the user ID and address. A complete description of these rules is in the SNA Distribution

Services  book.

Optional Parameter

TEXT Specifies the description of the nickname.

Note:

If a user nickname is specified when sending a note using the OfficeVision program, the text for the TO or COPYLIST fields is filled with the nickname text rather than the directory entry description.

***DFT:** The default description is used for the text. The default description is the first description associated with the specified USERID or LISTID parameter.

'nickname-description': Specify a description to further identify the nickname. A maximum of 50 characters can be specified and must be enclosed in apostrophes.

Example for ADDNCK

```
ADDNCK NCK(SEC44A *PUBLIC) USRID(XZWS44A RCHAS1)
      TEXT('Secretary for Department 44A')
```

This command adds a public user nickname which is a short version of the User ID and Address XZWS44A RCHAS1. If this nickname is unique it is added to the directory.

Error messages for ADDNCK

*ESCAPE Messages

CPF8360

Not enough storage for commitment control operation.

CPF8AA1

Library QUSRSYS not completely installed.

CPF905C

Error occurred trying to find a translation table.

CPF9838

User profile storage limit exceeded.

CPF9A89

Nickname function not successful.

ADDNODLE (Add Node List Entry) Command Description

ADDNODLE Command syntax diagram

Purpose

The Add Node List Entry (ADDNODLE) command adds a new entry to an existing node list object.

Required Parameter

NODL Specifies the qualified name of the node list object to which the entry is added.

The name of the node list can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

node-list-name: Specify the name of the node list to which the entry is added.

Optional Parameters

RMTLOCNAME

Specifies the name and address type of the system to add to the list object. The name can be an SNA network ID and control point name, an internet protocol host name, or an internet address.

An SNA node name is specified using the format *nnnnnnnn.ccccccc*, where *nnnnnnnn* is the network ID and *ccccccc* is the control point name. If only the control point name is specified, the local network ID (LCLNETID) network attribute is used as the value of the network identifier (ID) of the system being added to the node list. If the LCLNETID network attribute is changed, the new value does not affect the existing entries.

A host name must follow these conventions:

- The first character must be either an English alphabetic character or a numeric character.
- The last character must be either an English alphabetic character or a numeric character.
- Blanks () are not allowed.
- The special characters, period (.) and minus (-), are allowed.
- Parts of the name separated by periods (.) cannot exceed 63 characters in length.
- Internet address names (in the form *nnn.nnn.nnn.nnn*) are not allowed.
- Names must be from 1 to 255 characters in length.

The internet address is specified in the form *nnn.nnn.nnn.nnn*, where *nnn* is a decimal number ranging from 0 through 255. An internet address is not valid if it has a value of all binary ones or all binary zeros for the network identifier (ID) portion or the host ID portion of the address. If the internet address is entered from a command line, the address must be enclosed in apostrophes.

Element 1: Name or Address

remote-location-name: Specify the remote location name to add to the node list.

Element 2: Address Type

***SNA:** The node name has a Systems Network Architecture (SNA) address type.

***IP:** The node name has an Internet Protocol (IP) address type.

CPNAME

Specifies the SNA node name that is being added to the node list object. This system is specified as two elements: the network ID and the control point name.

Notes:

1. The RMTLOCNAME parameter is recommended for use in specifying the network ID and the control point name.
2. When the RMTLOCNAME parameter is used to specify the name of a system to add to the node list, *RMTLOC must be specified for this parameter.

***RMTLOC:** The network ID and control point name are specified using the RMTLOCNAME parameter.

Element 1: Network ID

***NETATR:** The local network ID (LCLNETID) network attribute is used as the value of the network identifier (ID) of the system being added to the node list. If the LCLNETID network attribute is changed, the new value does not affect the existing entries.

network-ID: Specify the network ID of the system being added to the node list.

Element 2: Control Point Name

control-point-name: Specify the control point name of the system being added to the node list.

Note: This field is left blank when *RMTLOC is specified as the network ID.

TEXT Specifies the text that briefly describes the node list entry. More information is in Commonly used parameters.

***BLANK:** Text is not specified.

'description': Specify no more than 50 characters of text, enclosed in apostrophes.

Examples for ADDNODLE

Example 1: Adding a System in the Local Network to a Node List

```
ADDNODLE  NODL(MYLIB/NODL02)
           RMTLOCNAME(AS400A01 *SNA)
           TEXT('THE FIRST ISERIES 400 SERVER IN MY NETWORK')
```

This command adds an entry for system AS400A01, which is in the local network, to the node list NODL02 in library MYLIB. The entry has an address type of SNA. The text description for the entry is THE FIRST ISERIES 400 SERVER IN MY NETWORK.

Example 2: Adding a Host Name to a Node List

```
ADDNODLE  NODL(MYLIB/NODL02)
           RMTLOCNAME(MYSYS.NET1.LOCAL *IP)
           TEXT('SYSTEM AT HEADQUARTERS')
```

This command adds an entry for host name MYSYS.NET1.LOCAL to the node list NODL02 in library MYLIB. The entry has an address type of IP. The text description for the entry is SYSTEM AT HEADQUARTERS.

Example 3: Adding an Internet Address to a Node List

```
ADDNODLE  NODL(MYLIB/NODL02)
           RMTLOCNAME('9.13.156.8' *IP)
           TEXT('MINNEAPOLIS OFFICE')
```

This command adds an entry for internet address 9.13.156.8 to the node list NODL02 in library MYLIB. The entry has an address type of IP. The text description for the entry is MINNEAPOLIS OFFICE.

Error messages for ADDNODLE

*ESCAPE Messages

CPF7AD4

Network ID &1 not in correct format.

CPF7B18

Control point &1 not in correct format.

CPF813E

Node list &4 in &9 damaged.

CPF96B3

Node list entry already exists.

CPF96B5

Remote location name not in correct format.

CPF9801

Object &2 in library &3 not found.

CPF9802

Not authorized to object &2 in &3.

CPF9803

Cannot allocate object &2 in library &3.

CPF9807

One or more libraries in library list deleted.

CPF9808

Cannot allocate one or more libraries on library list.

CPF9810

Library &1 not found.

CPF9820

Not authorized to use library &1.

CPF9830

Cannot assign library &1.



ADDOBJCRQA (Add Object CRQ Activity) Command Description

Note: To use this command, you must have the 5722-SM1 (System Manager for iSeries) licensed program installed.

ADDOBJCRQA Command syntax diagram

Purpose

The Add Object Change Request Activity (ADDOBJCRQA) command adds an object distribution activity to a change request description. The object referred to in the activity can be an OS/400 object identified by an OS/400 object name or a global name, or a non-OS/400 object such as a PS/2 file which is identified by a global name. The object can also be an installable object which is identified by a global name. An installable object is an object created with more than one OS/400 object.

The activity can be conditioned so that it only runs after one or more other activities have completed (successfully or unsuccessfully). The activity can also be scheduled to run at a future date and time.

Restrictions:

1. You must have *CHANGE authority to change the request description and *EXECUTE authority to the library.
2. If a NODL value is specified, the node list can only contain entries that have a value of *SNA for the address type.
3. An object can be specified using an OS/400 object name or a global name but not both.
4. The global name can be a maximum of 65-*n* characters in length, where *n* is the number of tokens. A maximum of 10 tokens can be specified.
5. The object to be distributed cannot reside in the QTEMP library.

Notes:

The following notes provide information on how the command works.

1. All conditions must be satisfied before the activity can run.
2. Authorization to the object specified on the activity is not verified until the activity runs.
3. If a global name is used, the Add Distribution Catalog Entry (ADDDSTCLGE) command can be used to indicate where the object is located or stored.
4. The start times indicate when the activity can be started. Actual start times can be later due to network and system delays.
5. Only OS/400 program objects or file members such as CL and REXX can be run.
6. The save and restore history for the object is not updated when it is sent or retrieved.
7. Active message queues are not saved when libraries (*LIB) are sent or retrieved.
8. Activities are not guaranteed to run in the same jobs. Therefore, references to objects in library QTEMP should be avoided because the scope of QTEMP is only within the job.
9. If you need to cancel an installation of a NetView Distribution Manager (NetView DM) change file or other non-OS/400 object, you must use the Add Change Management Activity (QNSADDCM) API.

Required Parameters

CRQD Specifies the change request description object name.

The possible library values are:

***LIBL:** All of the libraries in the user and in the system portions of the job's library list are searched.

***CURLIB:** The current library for the job used to locate the object.

library-name: Specify the name of the library to be searched.

change-request-description: Specify the name of the change request description object.

ACTIVITY

Specifies the name of the activity to add to the change request description.

***GEN:** An activity name is generated. The activity ID is of the form QACTxxxxxx, where xxxxxx is the first multiple of ten not already being used.

***LAST:** The activity is the last to run in the change request. When *LAST is specified for the activity (ACTIVITY) parameter, the condition (COND) parameter and the start time (STRTIME) parameter cannot be specified. Only one activity named *LAST can exist in the change request description.

activity-name: Specify a 10-character activity name.

ACTION

Specifies the object distribution function to be performed.

***SND:** Sends the specified object to the specified managed system or systems.

***RTV:** Retrieves the specified object from the specified managed system or systems. To retrieve an object from more than one system, a global name with an *ANY token is required, so that each retrieved object has a unique global name. Global names with unspecified tokens (*ANY, *HIGHEST, or *LOWEST) are stored in the distribution repository when they are retrieved.

***DLT:** Deletes the specified object on the specified managed system or systems.

***RUN:** Runs the specified program on the specified managed system or systems. OS/400 program objects (*PGM), REXX programs or file members containing a CL input stream or a REXX procedure can run on an iSeries managed system.

***SNDRUN:** Sends the specified program and runs it on the specified managed system or systems. The program that is sent is deleted on completion.

***INS:** Installs the objects previously packaged for installation on the specified managed system or systems. Only installable objects can be installed.

***SNDINS:** Sends the objects, previously packaged for installation, and installs them on the specified managed system or systems. Only objects identified by global names can be installed.

***UNINS:** Removes the objects installed on the specified managed system or systems. Only objects installed through an installable object can be uninstalled.

OBJ Specifies the name of the object that runs, is sent, retrieved, or deleted. For send and retrieve actions, the object name represents the name of the object on both the central site system and the managed systems.

***GLOBAL:** The object is identified by the global name specified on the GLBNAME parameter.

***COMPNAME:** Indicates the component name that is used to identify the installable objects that must be uninstalled from the managed system.

The possible library values are:

***LIBL:** All of the libraries in the user and in the system portions of the job's library list are searched.

***CURLIB:** The current library for the job is used to locate the object.

library-name: Specify the name of the library to be searched.

object-name: Specify the object name. Only characters A through Z and 0 through 9 can be used in the object names.

Optional Parameters

GLBNAME

Specifies a global name, which is a series of tokens that uniquely identify an object in an SNA network. The global name represents the name that is used to locate the appropriate catalog entry on both the central site system and the managed systems. The catalog entry specifies the object that is to be used on that system. For example, if a retrieve action is specified, the global name is used to determine the object that is to be retrieved on the managed system. Also, the global name shows the location where it is to be stored on the central site system.

Special values in a token position indicate how to search for the object. By specifying *ANY in a token position, the token is ignored when searching for the correct object. If multiple objects are found matching the tokens specified, an error is returned.

If an object is sent, the global name must have been previously cataloged so that it is associated with a local object name or associated with an object in the distribution repository. Retrieved objects, for which no catalog entry exists, are placed in the distribution repository. The GLBNAME parameter cannot be specified if the object name is not *GLOBAL.

The GLBNAME parameter is not valid when ACTION(*UNINS) is specified. When the OBJ is *GLOBAL and the global name maps to an installable object, the global name must have the following structure:

```
ComponentName REF RefreshLevel
```

In the previous example:

- Component name are the tokens before the token with the REF value. It is used to distinguish objects from an installable object from those from another. The component can be between 1 and 7 tokens.
- The REF token is required to identify the global name as an installable object, and can only be specified from the second to the eighth token in the global name.
- The refresh level is a token with a numeric value. The refresh level shows the level of the installable object and must follow the token with the REF value.

Element 1: Token 1

***NETID:** The first global name token value is a network ID generated by the command from the network attributes. The network ID is determined by the current value of the LCLNETID network attribute value.

***SERVER:** This token is stored within the change request activity with the value &SERVER, and is replaced by the short name of the change control server when the object is distributed.

***TARGET:** This token is stored within the change request activity with the value &TARGET, and is replaced by the short name of the target when the object is distributed.

***MDDATE:** This token is stored within the change request activity with the value &DATE, and is replaced when distributed by the date that the object was last changed.

***MDTIME:** This token is stored within the change request activity with the value &TIME, and is replaced when distributed by the time that the object was last changed.

global-name-token: Specify the first token of the global name. The first token is recommended to be the registered enterprise ID or network ID.

Element 2-10: Token 2-10

***ANY:** Any token value matches when searching for the object where the action is to be performed. This is useful when retrieving objects for which some of the tokens in the global name are not known or vary between systems.

***HIGHEST:** The object with the highest token value has the action performed on it. The token must be ordered. This is useful when a token in a global name is used to indicate a different version of the object and you need to manipulate the object with the highest version level.

***LOWEST:** The object with the lowest token value has the action performed on it. The token must be ordered. This is useful when a token in a global name is used to indicate a different version of the object and you need to manipulate the object with the lowest version level.

***NETID:** The network ID of this system is used. The network ID is determined by the current value of the LCLNETID network attribute value.

***CPNAME:** The control point name of this system is used. The control point name is determined by the current value of the LCLCPNAME network attribute value.

***SERVER:** This token is stored within the change request activity with the value &SERVER, and is replaced by the short name of the change control server when the object is distributed.

***TARGET:** This token is stored within the change request activity with the value &TARGET, and is replaced by the short name of the target when the object is distributed.

***MDDATE:** This token is stored within the change request activity with the value &DATE, and is replaced when distributed by the date that the object was last changed.

***MDTIME:** This token is stored within the change request activity with the value &TIME, and is replaced when distributed by the time that the object was last changed.

global-name-token: Specify one of a series of 1 to 16-character tokens that uniquely identify the object on which the action is to be performed. Characters A through Z and 0 through 9 can be used. Other special values (@, #, and \$) can be used for tokens that represent network IDs and system names.

OBJTYPE

Specifies the object type. It cannot be specified if a global name or a component name is used.

***FILEDATA:** A file member should be transferred without the file attributes. This is used to move files between an iSeries server and another iSeries server. The *FILE object type can be used with iSeries servers to preserve the file attributes.

object-type: Specify the OS/400 object type. For a list of valid object types, see Commonly used parameters.

MBR Specifies the physical file member name. This cannot be specified unless the object type is *FILE or *FILEDATA.

***ALL:** The action should be performed on all members within the physical file. The object type must be *FILE. *ALL must be used for file types that do not have members such as device files.

Note: *ALL cannot be specified if the action is *RUN or *SNDRUN.

***FIRST:** The action should be performed on the first member (by date added) in the physical file. The member name is determined when the activity runs. *FIRST is not allowed when the action is *RUN.

***LAST:** The action should be performed on the last member (by date added) in the physical file. The member name is determined when the activity runs. *LAST is not allowed when the action is *RUN.

member-name: Specify the member name on which the action should be performed. The member name should be specified when action is *RUN.

DATATYPE

Specifies the data type of the member. It is used to specify the type of source file that is run on the managed system. This parameter is ignored when a file is not being sent or run. In addition, this parameter cannot be specified when the object type is not *FILEDATA.

***UNSPEC:** Unspecified file member type. If the data type cannot be determined at the managed system, or if the name of the file where this member resides is QCLSRC, then the file member is treated as a CL batch input stream. If the source file is named QREXSRC, the file member is treated as a REXX procedure.

***CL:** The file member contains control language, in other words, an OS/400 CL batch input stream.

***REXX:** The file member contains a REXX procedure.

COMPNAME

Component name, which is the set of tokens of a global name previous to the REF token. The component name is used to identify the installable objects that will be uninstalled. COMPNAME is only valid when ACTION(*UNINS) and OBJ(*COMPNAME) are specified.

Element 1: Token 1

The only special value allowed for the first token is *NETID.

***NETID:** The network ID of this system is used. The network ID is determined by the current value of the LCLNETID network attribute value.

component-name-token: One of a series of 1 to 16 character tokens that uniquely identifies the object on which the action is to be performed. Characters A through Z and 0 through 9 can be used. Other special values (@, #, and \$) can be used for tokens that represent network IDs and system names.

Elements 2-7: Tokens 2-7

***NETID:** The network ID of this system is used. The network ID is determined by the current value of the LCLNETID network attribute value.

***CPNAME:** The control point name of this system is used. The network ID is determined by the current value of the LCLCPNAME network attribute value.

component-name-token: One of a series of 1 to 16 character tokens that uniquely identifies the object on which the action is to be performed. Characters A through Z and 0 through 9 can be used. Other special values (@, #, and \$) can be used for tokens that represent network IDs and system names.

REFLVL

The refresh level is the level of the installable object that will be uninstalled. REFLVL is only valid when ACTION(*UNINS) and OBJ(*COMPNAME) are specified.

***ALL:** All the installable objects matching the component name with any refresh level will be uninstalled.

refresh-level: Specify the level of the installable object to be uninstalled. The level is a numeric value up to 16 characters.

NODL Specifies that the node list parameter is the object name that contains a list of systems that are the destinations for the activity. This parameter cannot be specified if the control point name (CPNAME) parameter is also specified.

***NONE:** The systems on which this activity is to be performed are not specified by a node list. Individual control point names must be specified.

The possible library values are:

***LIBL:** All of the libraries in the user and system portions of the job's library list are searched for the node list object.

***CURLIB:** The current library for the job is used to locate the node list object.

library-name: Specify the name of the library to be searched.

node-list-name: Specify the node list object name containing the list of systems on which the activity is to be performed.

CPNAME

Specifies the APPN control point names of the managed systems on which this activity is to be performed. Control point names cannot be specified if the node list (NODL) parameter is specified.

***NONE:** The systems on which this activity is to be performed are not identified individually. A node list must be specified.

***NETATR:** The network ID of the local system is used. This is useful when the node being specified is in the same network as the local system.

network-identifier: Specify the APPN network identifier of the managed system on which the activity is to be performed.

control-point-name: Specify the APPN control point name of the managed system on which the activity is to be performed. For NetView Distribution Management Agents, the control point name is the change control client which supports numeric characters (0-9) in the first position of control point names that are valid in other platforms.

TGTRLS

Specifies the release of the operating system on which you intend to use the object. This parameter is ignored for objects with global names that are in the SystemView distribution repository or for actions other than send or retrieve.

***CURRENT:** The object is used on the release of the operating system currently running on your system. If V5R2M0 is running on your system, *CURRENT means that you intend to use the object on a system with V5R2M0 installed. The object can also be used on a system with any later release of the operating system installed.

***PRV:** The object is intended for a system that is at the previous release level compared to the local system.

Note:

Modification levels are not supported. To specify a previous release with a modification level other than 0, such as V4R1M4, specify the value for the release rather than the special value of *PRV.

release-level: Specify the release level in the VxRxMx format. The object is used on a system with the specified release or with any later release of the operating system installed.

Valid values depend on the current version, release, and modification level and they change with each new release.

REPLACE

Specifies if the object should be replaced if it already exists. This parameter cannot be specified for actions other than for *SND, *SNDRUN, and *RTV.

***NO:** An error is returned if the object already exists.

***YES:** The object is replaced if it already exists.

DTACPR

Specifies that data is compressed when sending or receiving. This parameter cannot be specified for actions other than *SND, *SNDRUN, and *RTV. SNA compression with a prime compression character of blank is performed.

***NONE:** The file data is not compressed when sent or when retrieved.

***SNA:** The file data is compressed when sent or when retrieved.

Note:

Objects that are globally named can have compression information specified when they were added to the distribution repository using Add Distribution Catalog Entry (ADDDSTCLGE) command.

KEEPCLGE

Specifies if the catalog entry and associated save file corresponding to the installable object will be kept in the specified system or systems. The KEEPCLGE parameter is only valid when ACTION(*SNDINS) or ACTION(*INS) is specified.

***NO:** The catalog entry and associated save file are not kept.

***YES:** The catalog entry and associated save file are kept.

PARM Specifies the parameters to be passed when starting the program. This cannot be specified if the action is not *RUN or *SNDRUN. Up to 20 parameters can be specified.

parameter: Specify a 1 to 253 character parameter. The prompt panel initially allows 50 characters to be entered. By entering an ampersand (&) in position 1, the field expands for larger parameters.

COND Specifies which conditions must be met before this activity can be performed. Each condition identifies an activity that must run before this activity and the value the end code from that activity must have to allow this activity to run. The default condition is that the previous activity (in alphabetical order) must complete successfully before this activity can be run.

Single value:

***NONE:** There are no conditions for this activity.

Element 1: Conditioning Activity

The activity that must be run before this activity.

***PRV:** This activity is conditioned on the previous activity. Activities are ordered alphabetically by activity name. If the activity being added is the first activity, a previous activity does not exist and any condition with *PRV is marked as having been met.

conditioning-activity-name: Specify the name of the activity that must be run before this activity. The activity name specified in the activity (ACTIVITY) parameter cannot be specified in the conditioning activity name. An activity cannot be conditioned on itself.

generic-conditioning-activity-name:* Specify the generic name of the activities that must be run before this activity.

Element 2: Relational Operator

This element is the relational operator to use when comparing the end code from the conditioning activity.

***EQ:** Equal

***GT:** Greater than

***LT:** Less than

***NE:** Not equal

***GE:** Greater than or equal

***LE:** Less than or equal

Element 3: Condition Code

This element is the value compared to the actual end code of the conditioning activity.

***SUCCESS:** The activity ended successfully (0 <= end code <= 9). This end code can only be specified with relational operator *EQ or *NE.

***FAIL:** The activity failed (10 <= end code <= 89). This end code can only be specified with relational operator *EQ or *NE.

***NOTRUN:** The activity never started (90 <= end code <= 99). This end code is only specified with relational operator *EQ or *NE.

***ANY:** The activity ended with any end code. This end code is only specified with relational operator *EQ.

end-code: Specify an integer value (0-99) that indicates the result of an activity (success or failure). The end code ranges and descriptions are:

00 Activity completed successfully.

01-09 Activity completed with warning messages.

10-29 Activity did not complete successfully.

30-39 Activity was canceled by a user before it completed.

- 30 = Activity ended with *CNTRLD option
- 35 = Activity ended with *IMMED option
- 39 = Activity ended with *FRCFAIL option

40-49 Activity was not run due to errors detected by the application.

- 40 = Activity not run for security reasons

90-99 Activity was not run because conditions or schedules were not met.

- 95 = Scheduled start time expired
- 99 = Conditions cannot be met

Element 4: Condition Mode

This element indicates which systems the conditioning activity must have completed on before this activity can be performed.

***ALLNODES:** The conditioning activity specified must complete on all nodes before this activity runs.

***SAMENODE:** When the conditioning activity specified completes for a given node, the activity specified on the ACTIVITY parameter may run for that same node even though the conditioning activity specified cannot have completed for all other nodes. In the case where this activity lists a node not in the conditioning activity, this activity may run for that node; the condition is ignored.

***NONE:** There are no conditions for this activity.

STRTIME

Specifies the date and time when this activity can be started on the central site system. The current date and time values, and next date values are determined when the change request is submitted.

Element 1: Start After Time

***CURRENT:** This activity can start any time on or after the time when the change request is submitted.

start-after-time: Specify the time when this activity can start. The time can be entered as 4 or 6 digits (hhmm or hhmmss) where hh = hours, mm = minutes, and ss = seconds. Seconds are optional. The time can be specified with or without a time separator such as a colon (:). With a time separator, specify a string of 5 or 8 digits (hh:mm or hh:mm:ss).

Element 2: Start After Date

***CURRENT:** This activity can start on or after the date on which the change request is submitted.

***NEXT:** The activity can start on any date after the date when the change request is submitted.

start-after-date: Specify the date after which this activity can start. The date must be specified in the job date format.

Element 3: Start Before Time

This element is ignored if the start before date is *ANY.

***ANY:** The activity can start at any time on or before the start before date.

***CURRENT:** The activity must start before the time when the change request was submitted on the date specified on the start before data element.

start-before-time: Specify the time before the activity must be started. If the activity cannot be started before this time, it never starts. The time can be entered as 4 or 6 digits (hhmm or hhmmss) where hh = hours, mm = minutes, and ss = seconds. Seconds are optional. The time can be specified with or without a time separator such as a colon (:). With a time separator, specify a string of 5 or 8 digits (hh:mm or hh:mm:ss).

Element 4: Start Before Date

***ANY:** The activity can start at any time after the start after time and the start after date.

***CURRENT:** The activity must start on the date the change request is submitted.

***NEXT:** The activity must start by the day after the date the change request is submitted.

start-before-date: Specify the date before the activity must start. If the activity cannot be started by this date, it never starts. The date must be specified in the job date format.

RMTSTRTIME

Specifies the date and time when the activity can begin running on the managed system. The current date and time values and next date values are determined when the activity begins running at the central site system based on the central site date and time.

Element 1: Time Zone

The time zone of the remote start time.

***LCLSYS:** The remote start time is specified in the time zone of the central site system.

***MGDSYS:** The remote start time is specified in the time zone of the managed system.

Element 2: Start After Time

This is the definition of the time after which the activity is to start.

***CURRENT:** This function can start on the managed system at any time on or after the time this activity is started on the central site system on the date specified in element 3.

start-after-time: Specify the time when this function can start on the managed system. The time can be entered as 4 or 6 digits (hhmm or hhmmss) where hh = hours, mm = minutes, and ss = seconds. Seconds are optional. The time can be specified with or without a time separator. With a time separator, specify a string of 5 or 8 digits (hh:mm or hh:mm:ss).

Element 3: Start After Date

***CURRENT:** This function starts on the managed system on any date on or after the activity starts on the central site system.

***NEXT:** This function starts on the managed system on any date after the activity starts on the central site system.

start-after-date: Specify the date after the functions start on the managed system. The date must be specified in the job date format.

Notes:

1. The special values *CURRENT and *NEXT cannot be specified for the date and the time when the time zone value *MGDSYS is specified.
2. This parameter can only be specified if *RUN, *SNDRUN, *INS, *SNDINS, and *UNINS actions are specified.

TEXT Specifies the activity description.

***GEN:** A description is generated based on the action specified.

text-description: Specify a 50-character description of the activity.

HOLD Specifies that the activity be held when the change request is submitted.

***NO:** The activity is not held. It runs when all conditions and the start time are met.

***YES:** The activity is held for all nodes when the change request is submitted. It must be released by you before it runs.

Examples for ADDOBJCRQA

Example 1: Adding an Activity to a Change Request Description

```
ADDOBJCRQA CRQD(MYLIB/CR1) ACTIVITY(ACT01) ACTION(*RTV)
OBJ(QGPL/QXYZ) OBJTYPE(*JOB) CPNAME(((*NETATR SYS1))
```

This example shows how to add an activity to change request description CR1 in library MYLIB which retrieves the QGPL/QXYZ job description from the iSeries server SYS1 in the local network.

Example 2: Adding an Activity to an Accounting System

```
ADDOBJCRQA CRQD(MYLIB/CR2) ACTIVITY(ACT02) ACTION(*SND)
OBJ(ACCTLIB/TAXFILE) OBJTYPE(*FILE) MBR(DEDUCTIONS)
TGTRLS(*PRV) STRTIME((23:00:00 9/30/02))
NODL(NETLIB/ACCTSYS)
```

This example shows how to add an activity to send a tax table to all of the iSeries accounting servers in the ACCTSYS node list at 11 p.m. on 30 September 2002. The accounting systems are at the previous release level.

Example 3: Adding an Activity to Retrieve a Program

```
ADDOBJCRQA CRQD(MYLIB/CR3) ACTIVITY(*GEN) ACTION(*RTV)
OBJ(*GLOBAL) GLBNAME(CUSTNET PCSOFT WDAPP VER5 020314)
CPNAME((CUSTNET DEVPS2))
```

```

ADDOBJCRQA CRQD(MYLIB/CR3) ACTIVITY(*GEN) ACTION(*SND)
  OBJ(*GLOBAL) GLBNAME(CUSTNET PCSOFT WDWAPP VER5 020314)
  NODL(NETLIB/PS2SE)
ADDOBJCRQA CRQD(MYLIB/CR3) ACTIVITY(*GEN) ACTION(*RUN)
  OBJ(*GLOBAL) GLBNAME(CUSTNET PCSOFT WDWAPP VER5 020314)
  COND((*PRV *EQ *SUCCESS *SAMENODE))
  RMTSTRTIME((*MGDSYS (23:00:00 10/20/02))
  NODL(NETLIB/PS2SE)

```

This command adds activities to retrieve a program from a PS/2, and sends it to all of the PS/2s in the southeast area. The PS/2s are identified in the PS2SE node list. It runs it on the PS/2s at 11 p.m. in the time zone where the PS/2 is located only if the send to the PS/2 was successful. The activity names are generated.

Example 4: Adding an Activity to Retrieve a File from Multiple Systems

```

ADDOBJCRQA CRQD(MYLIB/CR4) ACTIVITY(ONLY) ACTION(*RTV)
  OBJ(*GLOBAL) GLBNAME(CUSTNET SALES *ANY *HIGHEST)
  STRTIME((22:00:00 *CURRENT) (06:00:00 *NEXT))
  NODL(NETLIB/STORES)

```

This command adds an activity to retrieve the most recent nightly sales file from each system identified in the STORES node list. The files are cataloged as CUSTNET SALES system-name date-created. The file must be retrieved after 10 p.m. on the day the request is submitted but before 6 a.m. the next morning when the stores open.

Example 5: Adding an Activity to Send an Installable Object

```

ADDOBJCRQA CRQD(MYLIB/CR1) ACTIVITY(01) ACTION(*SND)
  OBJ(*GLOBAL) GLBNAME(CUSTNET ASOBJ PCKOBJ 01269 REF 0001)
  CPNAME((*NETATR SYS1))

```

This command adds an activity to send an object that was previously packaged for installation in the central site to the iSeries server SYS1.

Example 6: Adding an Activity to Send and Install an Installable Object

```

ADDOBJCRQA CRQD(MYLIB/CR2) ACTIVITY(ACT02) ACTION(*SNDINS)
  OBJ(*GLOBAL) GLBNAME(CUSTNET ASOBJ PCKOBJ 01270 REF 0002)
  NODL(NETLIB/ACCTSYS) KEEPCLGE(*NO)

```

This command adds an activity to send and install an installable object that was previously packaged for installation to the node list ACCTSYS. The catalog entry in the managed system pointing to the installable object is not kept.

Example 7: Uninstalling an Installable Object

```

ADDOBJCRQA CRQD(MYLIB/CR4) ACTIVITY(ACT04) ACTION(*UNINS)
  OBJ(*COMPNAME) COMPNAME(CUSTNET ASOBJ PCKOBJ 03000)
  REFLVL(*ALL) CPNAME((*NETATR SYS4)

```

This command adds an activity to uninstall all the installable objects from the system SYS4 whose global name matches the component name CUSTNET ASOBJ PCKOBJ 03000, no matter what release level it has.

Error messages for ADDOBJCRQA

*ESCAPE Messages

None <<

ADDOPTCTG (Add Optical Cartridge) Command Description

ADDOPTCTG Command syntax diagram

Purpose

The Add Optical Cartridge (ADDOPTCTG) command adds an optical disk cartridge and its volumes to an optical media library. Each optical cartridge contains two volumes.

A 12-character date and time stamp is assigned by the system to a new volume when it is added to the optical library. This stamp is used as the volume identifier to track each volume until it is read. You can keep the system-generated volume identifier or you can specify a new volume identifier when the volume is initialized.

Restriction: The user must have *USE authority to use this command. The command is shipped with *EXCLUDE public authority.

Required Parameter

MLB Specifies the name of the optical media library where the optical cartridge is to be added.

Optional Parameters

AUTL Specifies the authorization list used to verify authority to the optical cartridge and its volumes.

***PRV:** The previous authorization list is used. If no previous authorization list was saved or if the previous authorization list does not exist, the default authorization list, QOPTSEC, is used.

QOPTSEC: The default authorization list for the optical volumes, QOPTSEC, is used.

***NONE:** No security checking is performed for the optical volumes.

authorization-list-name: Specify the name of the authorization list used.

DIR Specifies whether the optical directory index is built for the optical volumes being added. The directory index is required to run Work with Optical Directories (WRKOPTDIR) and Display Optical (DSPOPT) when displaying directories. If not built now, the index will be built later the first time one of these commands is issued.

***NO:** The optical directory index will not be built for the optical volumes being added but instead will be built at a later time if needed. Using this option may result in better performance for the command than if DIR(*YES) were specified.

***YES:** The optical directory index is created now for each volume.

Example for ADDOPTCTG

```
ADDOPTCTG  MLB(OPT01)  AUTL(MYAUTH)
```

This command adds an optical cartridge and its volumes to the system in optical media library OPT01. The optical cartridge is secured with the authorization list MYAUTH.

Error messages for ADDOPTCTG

*ESCAPE Messages

OPT1245

Error processing directories for optical volume &1.

OPT1480

Add optical disk cartridge failed to complete successfully.

OPT1530

&1 does not represent a valid optical device.

OPT1555

Optical device &1 in use.

OPT1652

Device &1 is not an optical media library.

OPT1790

Operation conflicts with another request.

OPT1815

Internal program error occurred.

OPT1860

Request to optical device &1 failed.

OPT1861

No device description configured for resource &1.

OPT1862

No active device description for resource &1.

OPT1863

Optical libraries need to be reclaimed.

OPT2040

Error accessing backup control file.

OPT2301

Internal system object in use.

OPT2410

Authorization list &1 for volume &2 was not found.

OPT7740

User not authorized to object &2 in library &3 type &4.

ADDOPTSVR (Add Optical Server) Command Description

ADDOPTSVR Command syntax diagram

Purpose

The Add Optical Server (ADDOPTSVR) command enables OS/400, using the hierarchical file system (HFS) APIs, to access a remotely attached optical server. This command retrieves a list of optical volumes in each server and adds them to the optical index database allowing the volumes to be accessed using the HFS APIs. If a remote optical server is already enabled, you can use this command to refresh the volume list for that server.

Restrictions:

1. You must have *USE authority to use this command. It is shipped with *EXCLUDE public authority.
2. To use a remote optical server, the users must have the library that contains the communications side information for that destination in their library list.

Required Parameter

CSI Specifies the communications side information object name of the optical server to be added to the optical configuration. The communications side information object name is also referred to as the optical server name or the optical destination name. A maximum of 16 qualified names of servers can be specified.

The name of the communications side information object can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

communications-side-information-object-name: Specify the name of the communications side information object representing the optical server.

Example for ADDOPTSVR

```
ADDOPTSVR  CSI((QGPL/LAN01))
```

This command enables OS/400 to access the optical server defined by the communications side information object name LAN01 in library QGPL.

Error messages for ADDOPTSVR

*ESCAPE Messages

OPT0125

Command &1 completed with errors, more information in job log.

OPT6712

Remote optical server volume list rebuild failed.

ADDPEXDFN (Add Performance Explorer Definition) Command Description

ADDPEXDFN Command syntax diagram

Purpose

The Add Performance Explorer Definition (ADDPEXDFN) command adds a new performance explorer definition to the system. Each definition is stored as a member in the QAPEXDFN file in library QUSRSYS. A performance explorer definition identifies the performance data that is to be collected during a performance explorer session. A session can be started using the STRPEX (Start Performance Explorer) command. When starting a new session, a performance explorer definition name must be provided.

Additional information about the performance explorer tool can be found in the Performance Tools for

iSeries  book.

➤ Restrictions:

1. This command is shipped with PUBLIC *EXCLUDE authority.
2. You must have execute authority to the PGM library if PGM is specified.
3. To use this command you must have *SERVICE special authority or be authorized to the Service Trace function of Operating System/400 through iSeries Navigator's Application Administration support. The

Change Function Usage Information (QSYCHFUI) API, with a function ID of QIBM_SERVICE_TRACE, can also be used to change the list of users that are allowed to perform trace operations.

4. The following user profiles have private authorities to use the command:
 - QPGMR
 - QSRV



Required Parameter

DFN Specifies the name of the performance explorer definition being added. If the specified definition already exists in the QAPEXDFN file in library QUSRSYS, an error condition will occur. The user can either change the definition name or remove the existing definition using the Remove Performance Explorer Definition (RMVPEXDFN) command, and try this command again.

definition-name: Specify the name of the new performance explorer definition.

Optional Parameters

TYPE Specifies the type of performance data to be collected.

***STATS:** General performance program statistics are collected to help identify problem areas. This mode is mainly used as a map to help determine if and where more detailed information should be collected and analyzed.

In addition to collecting the information specified above, statistics mode also provides the option of counting the occurrences of specific types of events. These can be counted any of 4 counters provided.

To count the occurrences of a particular type of event, specify SLTEVT(*YES) and then choose the event-identifier(s) to be counted from the various event categories.

For example, TYPE(*STATS) SLTEVT(*YES) JOBEVT((*ALL 1)) would count all job events in counter 1.

***TRACE:** Detailed trace information is collected. This is the most detailed type of performance data collection available.

***PROFILE:** Collects relative cpu consumption data either by program or by job. This allows a user to determine what program/procedure is using excessive cpu time, or, what section of code within a program/procedure is using excessive cpu time.

PRFTYPE

Specifies the type of profile to collect. This parameter is only valid if TYPE(*PROFILE) is specified.

***PGM:** Specific programs are sampled to identify sections of code that are using larger amounts of resources. This information is very valuable when the user wants to improve the performance of a specific program or application.

***JOB:** Specific jobs are sampled to identify programs and procedures that are using larger amounts of resources. This mode can provide a view of all the programs and procedures in the entire system, and is equivalent to *TRACE mode of *PMCO events with a specified INTERVAL.



JOB Specifies which jobs are included in the performance explorer data collection session.

Single Value

***:** Only the job that issues the STRPEX (Start Performance Explorer) command is included.

***ALL:** All jobs on the system are included.

Element 1: Job Identifier

job-name: Specify the name of the job to be included in the performance explorer data collection session.

generic-job-name*: Specify the generic name of the job to be included. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. For more information on the use of generic names, refer to generic names.

» Job User Name Qualifier «

***ALL**: All jobs that match the specified job name are included.

user-name: Specify the name of the user of the job to be included.

generic-user-name*: Specify the generic user name of the jobs to be included.

» Job Number Qualifier «

***ALL**: All jobs that match the specified job name and user name are included.

job-number: Specify the job number to further qualify the job name and user name.

» Element 2: Thread Identifier

***ALL**: All threads of the specified job are included.

***INITIAL**: Only the initial thread of the specified job is included.

***SELECT**: Select the threads from a list of threads for the specified job. This value is only valid if the command is run in an interactive job.

thread-id: Specify the thread identifier of the job to be included. This is the thread id as shown on the WRKJOB command. «

TASK Specifies which licensed internal code (LIC) tasks are included in the performance explorer data collection session.

Note: LIC tasks can be obtained from the Performance Tools reports and WRKSYSACT command. There is no guarantee that LIC task names will remain the same or exist from system to system or release to release

***NONE**: No LIC tasks on the system are included.

***ALL**: All LIC tasks on the system are included.

task-name: Specify the name of the task to be included in the performance explorer data collection session.

generic-task-name*: Specify the generic name of the task to be included. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. For more information on the use of generic names, refer to generic names.

PGM Specifies the program whose performance profile data is collected. This parameter is only valid if TYPE(*PROFILE) is specified. Up to 16 programs may be specified.

Element 1: Program Name

The program name can be qualified by one of the following library values:

***LIBL:** The library list of the job that issues the STRPEX command is searched to find the specified program or service program.

library-name: Specify the library which contains the program or service program.

program-name: Specify the name of the program to be sampled.

Element 2: Module Name

***ALL:** All modules in the program or service program will be sampled. If sampling an OPM (Original Program Model), specify *ALL for this element.

module-name: Specify a specific module within the program or service program that is to be sampled.

Element 3: Procedure Name

***ALL:** All procedures in the specified module are sampled.

procedure-name: Specify a specific procedure within the specified module that is to be sampled. Specify the procedure name within single quotes if the procedure name contains lower case characters.

Element 4: Program Type

Indicate the type of program being specified.

***PGM:** The program being specified is a program (*PGM) object.

***SRVPGM:** The program being specified is a service program (*SRVPGM) object.

Element 5: Pane Size

Specifies the pane size, which is the number of consecutive program instruction addresses assigned to each counter. The smaller the pane size, the more fine-grained the program profile information will be.

4: The default pane size is 4.

numeric value: Specify the pane size to use for the program. Valid values are 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096.

DTAORG

Specifies how the data is organized.

Note:

This parameter is only valid if TYPE(*STATS) is specified.

***FLAT:** The performance explorer tool will not collect data for a parent-child relationship.

***HIER:** The performance explorer tool will collect data for a parent-child relationship.

MAXSTG

Specifies the maximum amount of storage, in kilobytes (K), that the performance explorer tool uses for collecting trace data.

Note: This parameter is only valid if TYPE(*TRACE) is specified.

10000: Up to 10000 kilobytes of storage is used.

maximum-K-bytes: >> Specifies the requested maximum amount of storage, in kilobytes (K), to use for the collected trace records. The system will calculate the minimum amount of storage that is necessary for the trace. This minimum storage size calculation is dependent upon the system's processor configuration. The minimum amount of storage may be significantly larger than the size specified on the MAXSTG parameter. The system will use the larger of the two values. <<

TRCFULL

Specifies whether the trace records wrap (replace oldest records with new records) or whether the trace stops when all of the storage specified by the MAXSTG parameter has been used.

Note: This parameter is only valid if TYPE(*TRACE) is specified.

***STOPTRC:** Tracing stops when the trace file is full of trace records.

***WRAP:** When the trace file is full, the trace wraps to the beginning. The oldest trace records are written over by new ones as they are collected.

MRGJOB

Specifies whether the data from different jobs should be merged in one data area or kept separate (for example, one data area per job).

Note: This parameter is only valid if DTAORG(*FLAT) is specified.

***YES:** The data from individual jobs is merged.

***NO:** The data from individual jobs is kept separate.

PGMBKTEVT

Specifies which program call flow events are included in the statistics mode definition.

Note: This parameter is only valid if TYPE(*STATS) is specified.

***DFT:** Statistics are to be collected on *MISTREND's, on all hooked *MIENTRYEXIT's, and on all hooked *JVA's.

***MISTREND:** Statistics are to be collected on all machine instructions.

***MIENTRYEXIT:** Statistics are to be collected on programs and procedures that have been explicitly hooked via the ENBPFRCOL parameter on the various compile and change job commands.

***JVA:** Statistics are to be collected on Java methods and Java native methods that have been explicitly hooked via the ENBPFRCOL parameter (or its equivalents) on the Java and JIT compile commands.

***PRC:** Statistics are to be collected on programs and procedures that have been implicitly hooked. This includes any program that has been compiled at optimization level 30 or below. Optimization level 40 programs require explicit compiler options which activate the trace job (trcjob) style hooks.

*MIENTRYEXIT and *PRC are mutually exclusive.

INTERVAL

Specifies the interval at which samples are taken of the program. A low interval will cause a high number of samples to be taken, and will also cause higher overhead. A low interval will also provide a significant amount of data.

Note:

This parameter is only valid if TYPE(*PROFILE) or TYPE(*TRACE) is specified. ➤ It is used in conjunction with BASEVT(*PMCO) to control the interval between *PMCO events. ⏪

milliseconds: Specify the interval at which samples are taken of the program. Valid values range from 0.1 to 200.0 milliseconds.


TRCTYPE

Specifies what type of trace performance data to be collected.

Note:

This parameter is only valid if TYPE(*TRACE) is specified.

Note:

Additional information about the TRCTYPE options can be found in the Performance Tools for iSeries  book.

***CALLRTN:** Specifies that call return events are included in the trace definition. Call return events occur when a program is entered and exited as well as when certain machine instruction are started and completed. The list of events included has been expanded in V5R1 from PGMEVT(*MIENTRY *MIEXIT *MISTR *MIEND) to include PGMEVT(*JVAENTRY *JVAEXIT *JVANTVMTHSTR *JVANTVMTHEND *JVAPRECALL *JVAPOSTCALL *PASEPRCENTRY *PASEPRCEXIT).

***BASIC:** Specifies that events relative to general performance analysis are included in the trace definition. This option should be used when it is unclear as to what type of performance problem determination is necessary.

***DSKIO1:** Specifies that events associated with disk input/output operations are included in the trace definition.

***DSKIO2:** Specifies that events associated with disk input/output operations plus higher level requests to do input/output operations are included in the trace definition.

***DSKSVR:** Specifies that events associated with disk server operations are included in the trace definition.

***DSKSTG:** Specifies that events associated with disk storage consumption are included in the trace definition.

***VRTADR:** Specifies that events associated with virtual address assignment are included in the trace definition.

***PGMACT:** Specifies that events associated with program activations and deactivations are included in the trace definition.

***FILEOPEN:** Specifies that events associated with file (*FILE) opens are included in the trace definition.

***PRFDTA:** > Specifies that the profiling event (the *PMCO event) are to be included in the trace definition. The INTERVAL parameter also needs to be set to a value other than *NONE to specify the interval at which to collect profile events. <<

***TASKSWT:** Specifies that events associated with tasking are included in the trace definition.

Single Value

***SLTEVT:** Only selected individual events (xxxEVT parameters) and machine instructions (MCHINST parameter) are included in the trace definition.

Note: If TRCTYPE(*SLTEVT) is specified, SLTEVT(*YES) must also be specified.

SLTEVT

For trace mode (TYPE(*TRACE)) collections, SLTEVT allows individual machine instructions and events to be specified in addition to the categories of events available with the TRCTYPE parameter. For statistics mode (TYPE(*STATS)) collections, SLTEVT allows individual events to be counted.

Note: This parameter is only valid if TYPE(*TRACE) or TYPE(*STATS) is specified.

***NO:** Do not allow selection of specific events.

***YES:** Allow selection of specific events.

TEXT Specifies the text that briefly describes the performance explorer definition. More information is in Commonly used parameters.

***BLANK:** Text is not specified.

'description': Specify no more than 50 characters of text, enclosed in apostrophes.

MCHINST

Specifies which machine instructions are included in the performance explorer data collection session. Use this parameter in conjunction with the PGMEVT parameter. MCHINST controls which of the possible machine instructions are to be included in the collection. PGMEVT(*MISTR and *MIEND) controls whether any machine instructions are included in the collection.

Note: This parameter is only valid if TYPE(*TRACE) and SLTEVT(*YES) are specified.

***ALL:** All machine instructions that are available for collection are included.

***NONE:** No machines instructions available for collection are included.

machine-instruction-name: Specify the name of the machine instruction to be included in the performance explorer data collection session.

BASEVT

Specifies which base events are included in the definition.

Note: This parameter is only valid if SLTEVT(*YES) is specified.


Single Value

***NONE:** No base events are included in the definition.

Element 1: Event Identifier

***ALL:** All base events are included in the trace mode definition or counted in the statistics mode definition.

In a statistics mode definition (TYPE(*STATS)), you can specify which event-identifiers are to be counted. But, some of the base event event-identifiers cannot be counted. An informational message will be shown if a definition attempts to count all base events.

event-identifier: Additional information about the BASEVT options can be found in the Performance Tools for iSeries  book.

Element 2: Event Counter

***NONE:** No base events are counted in the statistics mode definition.

event-counter: In a statistics mode definition (TYPE(*STATS)), events can be counted in one of four counters (1,2,3 or 4).


PGMEVT

Specifies which program call flow events are included in the trace definition.

Note: This parameter is only valid if TYPE(*TRACE) and SLTEVT(*YES) are specified.

***NONE:** No program call flow events are included in the trace definition.

***ALL:** All program call flow events are included in the trace definition.

event-identifier: Additional information about the PGMEVT options can be found in the Performance Tools for iSeries  book.

STGEVT

Specifies which auxiliary storage management events are included in the definition.

Note:


This parameter is only valid if SLTEVT(*YES) is specified.

Single Value

***NONE:** No auxiliary storage management events are included in the definition.

Element 1: Event Identifier

***ALL:** All auxiliary storage management events are included in the trace mode definition or counted in the statistics mode definition.

event-identifier: Additional information about the STGEVT options can be found in the Performance Tools for iSeries  book.

Element 2: Event Counter

***NONE:** No auxiliary storage management events are counted in the statistics mode definition.

event-counter: In a statistics mode definition (TYPE(*STATS)), events can be counted in one of four counters (1,2,3 or 4).

DSKEVT

Specifies which disk events are included in the definition.

Note:

This parameter is only valid if SLTEVT(*YES) is specified.


Single Value

***NONE:** No disk events are included in the definition.

Element 1: Event Identifier

***ALL:** All disk events are to be included in the trace mode definition or counted in the statistics mode definition.

***ALLSTR:** All disk start events are included in the trace mode definition or counted in the statistics mode definition.

event-identifier: Additional information about the DSKEVT options can be found in the Performance Tools for iSeries  book.

Element 2: Event Counter

***NONE:** No disk events are counted in the statistics mode definition.

event-counter: In a statistics mode definition (TYPE(*STATS)), events can be counted in one of four counters (1,2,3 or 4).

FAULTEVT

Specifies which page fault events are included in the trace definition.

Note:


This parameter is only valid if SLTEVT(*YES) is specified.

Single Value

***NONE:** No page fault events are included in the definition.

Element 1: Event Identifier

***ALL:** All page fault events are included in the trace mode definition or counted in the statistics mode definition.

event-identifier: Additional information about the FAULTEVT options can be found in the Performance Tools for iSeries  book.

Element 2: Event Counter

***NONE:** No page fault events are counted in the statistics mode definition.

event-counter: In a statistics mode definition (TYPE(*STATS)), events can be counted in one of four counters (1,2,3 or 4).

JOBEVT

Specifies which job or process related events are included in the definition.

Note:


This parameter is only valid if SLTEVT(*YES) is specified.

Single Value

***NONE:** No job or process related events are included in the definition.

Element 1: Event Identifier

***ALL:** All job or process related events are included in the trace mode definition or counted in the statistics mode definition.

event-identifier: Additional information about the JOBEVT options can be found in the Performance Tools for iSeries  book.

Element 2: Event Counter

***NONE:** No job or process events are counted in the statistics mode definition.

event-counter: In a statistics mode definition (TYPE(*STATS)), events can be counted in one of four counters (1,2,3 or 4).

LCKEVT

Specifies which lock or seize events are included in the definition.

Note:


This parameter is only valid if SLTEVT(*YES) is specified.

Single Value

***NONE:** No lock or seize events are included in the definition.

Element 1: Event Identifier

***ALL:** All lock and seize events are included in the trace mode definition or counted in the statistics mode definition.

event-identifier: Additional information about the LCKEVT options can be found in the Performance Tools for iSeries  book.

Element 2: Event Counter

***NONE:** No lock or seize events are counted in the statistics mode definition.

event-counter: In a statistics mode definition (TYPE(*STATS)), events can be counted in one of four counters (1,2,3 or 4).

SAREVT

Specifies which segment address register events are included in the definition.

Note:

This parameter is only valid if SLTEVT(*YES) is specified.


Single Value

***NONE:** No segment address register events are included in the definition.

Element 1: Event Identifier

***ALL:** All segment address register events are included in the trace mode definition or counted in the statistics mode definition.

***ALLSTR:** All segment address register start events are included in the trace mode definition or counted in the statistics mode definition.

event-identifier: Additional information about the SAREVT options can be found in the Performance Tools for iSeries  book.

Element 2: Event Counter

***NONE:** No segment address register events are counted in the statistics mode definition.

event-counter: In a statistics mode definition (TYPE(*STATS)), events can be counted in one of four counters (1,2,3 or 4).

DSKSVREVT

Specifies which disk server events are included in the definition.

Note:


This parameter is only valid if SLTEVT(*YES) is specified.

Single Value

***NONE:** No disk server events are included in the definition.

Element 1: Event Identifier

***ALL:** All disk server events are included in the trace mode definition or counted in the statistics mode definition.

event-identifier: Additional information about the DSKSVREVT options can be found in the Performance Tools for iSeries  book.

Element 2: Event Counter

***NONE:** No disk server events are counted in the statistics mode definition.

event-counter: In a statistics mode definition (TYPE(*STATS)), events can be counted in one of four counters (1,2,3 or 4).

OSEVT

Specifies which operating system events are included in the definition.

Note:


This parameter is only valid if SLTEVT(*YES) is specified.

Single Value

***NONE:** No operating system trace events are included in the definition.

Element 1: Event Identifier

***ALL:** All operating system events are included in the trace mode definition or counted in the statistics mode definition.

event-identifier: Additional information about the OSEVT options can be found in the Performance Tools for iSeries  book.

Element 2: Event Counter

***NONE:** No operating system events are counted in the statistics mode definition.

event-counter: In a statistics mode definition (TYPE(*STATS)), events can be counted in one of four counters (1,2,3 or 4).

JVAEVT

Specifies which Java events are included in the definition.

Note:


This parameter is only valid if SLTEVT(*YES) is specified.

Single Value

***NONE:** No Java events are included in the definition.

Element 1: Event Identifier

***ALL:** All Java events are included in the trace mode definition or counted in the statistics mode definition.

event-identifier: Additional information about the JVAEVT options can be found in the Performance Tools for iSeries  book.

Element 2: Event Counter

***NONE:** No Java events are counted in the statistics mode definition.

event-counter: In a statistics mode definition (TYPE(*STATS)), events can be counted in one of four counters (1,2,3 or 4).

CMNEVT

Specifies which communication events are included in the definition.


Note: This parameter is only valid if SLTEVT(*YES) is specified.

Single Value

***NONE:** No communication events are included in the definition.

Element 1: Event Identifier

***ALL:** All communication events are included in the trace mode definition or counted in the statistics mode definition.

event-identifier: Additional information about the CMNEVT options can be found in the Performance Tools for iSeries  book.

Element 2: Event Counter

***NONE:** No communication events are counted in the statistics mode definition.

event-counter: In a statistics mode definition (TYPE(*STATS)), events can be counted in one of four counters (1,2,3 or 4).

» APPEVT

Specifies which application events are included in the definition.


Note: This parameter is only valid if SLTEVT(*YES) is specified.

Single Value

***NONE:** No application events are included in the definition.

***ALL:** All application events are included in the trace mode definition or counted in the statistics mode definition.

Element 1: Event Identifier

event-identifier: Additional information about the APPEVT options can be found in the Performance Tools for iSeries  book.

Element 2: Event Counter

***NONE:** No application events are counted in the statistics mode definition.

event-counter: In a statistics mode definition (TYPE(*STATS)), events can be counted in one of four counters (1,2,3 or 4).

PASEEVT

Specifies which Portable Application Solution Environment (PASE) events are included in the definition.


Note: This parameter is only valid if SLTEVT(*YES) is specified.

Single Value

***NONE:** No PASE events are included in the definition.

***ALL:** All PASE events are included in the trace mode definition or counted in the statistics mode definition.

Element 1: Event Identifier

event-identifier: Additional information about the PASEEVT options can be found in the Performance Tools for iSeries  book.

Element 2: Event Counter

***NONE:** No PASE events are counted in the statistics mode definition.

event-counter: In a statistics mode definition (TYPE(*STATS)), events can be counted in one of four counters (1,2,3 or 4).

SYNCEVT

Specifies which synchronization events are included in the definition.


Note: This parameter is only valid if SLTEVT(*YES) is specified.

Single Value

***NONE:** No synchronization events are included in the definition.

***ALL:** All synchronization events are included in the trace mode definition or counted in the statistics mode definition.

Element 1: Event Identifier

event-identifier: Additional information about the SYNCEVT options can be found in the Performance Tools for iSeries  book.

Element 2: Event Counter

***NONE**: No synchronization events are counted in the statistics mode definition.

event-counter: In a statistics mode definition (TYPE(*STATS)), events can be counted in one of four counters (1,2,3 or 4).

JRNEVT

Specifies which Journal events are included in the definition.

Note:


This parameter is only valid if SLTEVT(*YES) is specified.

Single Value

***NONE**: No Journal events are included in the definition.

***ALL**: All Journal events are included in the trace mode definition or counted in the statistics mode definition.

Element 1: Event Identifier

event-identifier: Additional information about the JRNEVT options can be found in the Performance Tools for iSeries  book.

Element 2: Event Counter

***NONE**: No Journal events are counted in the statistics mode definition.

event-counter: In a statistics mode definition (TYPE(*STATS)), events can be counted in one of four counters (1,2,3 or 4).

FILSVREVT

Specifies which iSeries NetServer, File Server and Network File System Server and Client events are included in the definition.

Note:

This parameter is only valid if SLTEVT(*YES) is specified.

Single Value

***NONE**: No iSeries NetServer, File Server and Network File System Server and Client events are included in the definition.

***ALL**: All iSeries NetServer, File Server and Network File System Server and Client events are included in the trace mode definition or counted in the statistics mode definition.

Element 1: Event Identifier

event-identifier: Additional information about the FILSVREVT options can be found in the Performance Tools for iSeries book.

Element 2: Event Counter

***NONE**: No iSeries NetServer, File Server and Network File System Server and Client events are counted in the statistics mode definition.

event-counter: In a statistics mode definition (TYPE(*STATS)), events can be counted in one of four counters (1,2,3 or 4).

EXPCHEVT

Specifies which expert cache events are included in the definition.


Note: This parameter is only valid if SLTEVT(*YES) is specified.

Single Value

***NONE**: No expert cache events are included in the definition.

***ALL**: All expert cache events are included in the trace mode definition or counted in the statistics mode definition.

Element 1: Event Identifier

event-identifier: Additional information about the EXPCHEVT options can be found in the Performance Tools for iSeries  book.

Element 2: Event Counter

***NONE**: No expert cache events are counted in the statistics mode definition.

event-counter: In a statistics mode definition (TYPE(*STATS)), events can be counted in one of four counters (1,2,3 or 4).<<

Examples for ADDPEXDFN

Example 1: Using TYPE(*TRACE)

```
ADDPEXDFN DFN(TEST1) TYPE(*TRACE)
          JOB(*) MAXSTG(5000)
```

This command adds a new performance definition named TEST1, which will result in a member named TEST1 being added to file QAPEXDFN in library QUSRSYS. When this definition is used to start a performance explorer session (STRPEX command), detailed trace information will be collected for the job that invoked the STRPEX command. A maximum of 5000 kilobytes of trace data will be collected. When the trace record storage area is full no more trace records will be collected.

Example 2: Using TYPE(*PROFILE)

```
ADDPEXDFN DFN(TEST2) TYPE(*PROFILE)
          PGM((MYLIB/MYSRVPGM1 *ALL *ALL *SRVPGM))
```


This command adds a new performance explorer definition named TEST2. When this definition is used to start a performance explorer session (STRPEX command), performance profile information for service program MYSRVPGM1 in library MYLIB will be collected.

Example 3: Using TYPE(*TRACE)

```
ADDPEXDFN DFN(TEST3) TYPE(*TRACE) JOB(*ALL)
  TRCTYPE(*CALLRTN *DSKIO1)
  TEXT('Trace definition example')
```

This command adds a new performance explorer definition named TEST3. When this definition is used to start a performance explorer session (STRPEX command), performance trace information for program call/return and disk input/output operation will be collected.

Example 4: Counting SAR Events

```
ADDPEXDFN DFN(TEST4) TYPE(*STATS)
  SLTEVT(*YES) SAREVT( (*ALL 1) )
  TEXT('Count all SARs in counter 1')
```

This command adds a new performance explorer definition named TEST4. When this definition is used to start a session (STRPEX command), performance statistics for program and procedure call/return operations will be collected. In addition, all segment address register (SAR) events that occur will be counted in counter 1.

Error messages for ADDPEXDFN

None >>

ADDPEXFTR (Add Performance Explorer Filter) Command Description

ADDPEXFTR Command syntax diagram

Purpose

The Add Performance Explorer Filter (ADDPEXFTR) command adds a new performance explorer (PEX) filter to the system. Each filter is stored as a member in the QAPEXFTR file in library QUSRSYS. A performance explorer filter identifies the performance data that is to be collected during a performance explorer session, and is meant to limit the amount of data collected by specifying a compare value for specific events. If the data in the event matches the compare value, then the data will be collected. If not, the data is discarded. The filter is specified on the STRPEX (Start Performance Explorer) command.

Restrictions:

1. This command is shipped with PUBLIC *EXCLUDE authority.
2. To use this command you must have *SERVICE special authority, or be authorized to the Service Trace function of Operating System/400 through iSeries Navigator's Application Administration support. The Change Function Usage Information (QSYCHFUI) API, with a function ID of QIBM_SERVICE_TRACE, can also be used to change the list of users that are allowed to perform trace operations.
3. You must have execute authority to the libraries for PGMTRG, PGMFTR, and *X authority to the directories for JVAFTR, and PATHFTR if these parameters are specified.
4. The following user profiles have private authorities to use the command:
 - QPGMR
 - QSRV

Required Parameter

FTR Specifies the name of the performance explorer filter to be added. If the specified filter already exists in the QAPEXFTR file in library QUSRSYS, an error condition will occur. The user can either change the filter name or remove the existing filter using the Remove Performance Explorer Filter (RMVPEXFTR) command, and try this command again.

filter-name: Specify the name of the new performance explorer filter.

Optional Parameters

PGMTRG

If a procedure entry event (*PRCENTRY) occurs that matches this trigger specification, then PEX will begin collecting all events specified in the PEX definition used for the active PEX session. The events will be collected only for the thread where the trigger occurs. When the procedure exit event (*PRCEXIT) occurs that matches the trigger specification, PEX will stop collecting the events specified in the PEX definition.

If the *PRCENTRY/*PRCEXIT events are not specified in the PEX definition, this trigger will still function correctly. However, you will not see any *PRCENTRY/*PRCEXIT events in your data. If you have specified *PRCENTRY/*PRCEXIT events in your PEX definition, you will see those events in your data only after this trigger has been activated due to the trigger procedure being called.

The *PRCENTRY/*PRCEXIT events are enabled in procedures that are compiled at optimization level 30 and under. For optimization level 40, procedures have these events enabled if the LICOPT on CHGPGM has been set to 'CallTracingAtHighOpt' and the procedure stacks a frame on the invocation stack when called (non-leaf procedures).

Element 1: Program Name

The program name can be qualified by one of the following library values:

***LIBL:** The library list of the job that issues the ADDPEXFTR command is searched to find the specified program or service program.

library-name: Specify the library which contains the program or service program.

program-name: Specify the name of the trigger program.

Element 2: Module Name

module-name: Specify the module within the program or service program that contains the procedure that is to be the trigger.

Element 3: Procedure Name

***PEP:** The program entry procedure will act as the trigger. This is not valid for programs of type *SRVPGM.

procedure-name: Specify a specific procedure within the specified module that is to be the trigger. Specify the procedure name within single quotes if the procedure name contains lower case characters.

Element 4: Program Type

Indicate the type of program being specified.

***PGM:** The program being specified is a program (*PGM) object.

***SRVPGM:** The program being specified is a service program (*SRVPGM) object.

Element 5: Trigger Option

***ENTRYEXIT:** The specified trigger procedure enables the collection of events at procedure entry time. At procedure exit, the collection of events is disabled.

***ENTRY:** The specified trigger procedure enables the collection of events at procedure entry time. The collection of events is enabled for the duration of the PEX session.

PGMFTR

Specifies the program comparisons to use for this filter.

Element 1: Relational Operator

***EQ:** Events having program data that matches the specified program are included in the data collected by PEX.

***NE:** Events having program data that matches the specified program are excluded from the data collected by PEX. These events will not show up in the *MGTCOL object or the PEX database.

Element 2: Program Name

The program name can be qualified by one of the following library values:

***LIBL:** The library list of the job that issues the ADDPEXFTR command is searched to find the specified program or service program.

library-name: Specify the library which contains the program or service program.

***ALL:** All programs in the specified library will pass the program filter.

program-name: Specify the name of the program to be used as a compare value for the program filter.

Element 3: Module Name

***ALL:** All modules in the program or service program will pass the filter. If filtering an OPM (Original Program Model), specify *ALL for this element.

module-name: Specify a specific module within the program or service program to be used as a compare value for the program filter.

Element 4: Procedure Name

***ALL:** All procedures in the specified module are used as a compare value for the program filter.

procedure-name: Specify a procedure to use as the filter compare value. Specify the procedure name within single quotes if the procedure name contains lower case characters.

Element 5: Program Type

Indicate the type of program being specified.

***PGM:** The program being specified is a program (*PGM) object.

***SRVPGM:** The program being specified is a service program (*SRVPGM) object.

JVAFTR

Specifies the Java package, class, and methods to be used as compare values for the Java filter.

Element 1: Relational Operator

***EQ:** Events having Java data that match the specified packages, classes, and methods are included in the data collected by PEX.

***NE:** Events having Java data that matches the specified packages, classes, and methods are excluded from the collection and will not show up in the *MGTCOL object or the PEX database.

Element 2: Package Name

package-name: Specify the name of the Java package to be used as a compare value for the filter.

Element 3: Class Name

***ALL:** All classes in the specified package will pass the Java filter.

class-name: Specify a class within the package to be used as a compare value for the filter.

Element 4: Method Name

***ALL:** All methods in the specified class and package will pass the filter.

method-name: Specify a method to use as the filter compare value.

PATHFTR

Specifies the Integrated File System object path name comparisons to use for this filter.

Element 1: Relational Operator

***EQ:** All events that have an object path that matches the specified object path are included in the data collected by PEX.

***NE:** All events that have an object path data that matches the specified object path are discarded. These events will not show up in the *MGTCOL object or the PEX database.

Element 2: Path Identifier

object-path: The object path to use as a compare value for this filter.

MEMFTR

Specifies the memory pool comparisons to use for this filter.

Element 1: Relational Operator

***EQ:** All events that have pool identifier data that matches the specified pool are included in the data collected by PEX.

***NE:** All events that have pool identifier data that matches the specified pool are discarded. These events will not show up in the *MGTCOL object or the PEX database.

Element 2: Pool Identifier

pool-identifier The system pool id to use as a compare value for this filter. This pool ID corresponds to the pool identifier as shown on the WRKACTJOB command or on the output of PRTPEXRPT of type *TRACE.

DSKFTR

Specifies the disk unit comparisons to use for this filter.

Element 1: Relational Operator

***EQ:** All events that have disk identifier data that matches the specified disk are included in the data collected by PEX.

***NE:** All events that have disk identifier data that matches the specified disk will be discarded. These events will not show up in the *MGTCOL object or the PEX database.

Element 2: Disk Identifier

disk-identifier: The disk identifier used as a compare value for this filter. This disk identifier corresponds to the disk unit as shown on the WRKDSKSTS command or the output of PRTPEXRPT of type *TRACE. If a disk is mirrored, this identifier applies to both disks in the mirrored pair.

ASPFTR

Specifies the ASP (auxiliary storage pool) comparisons to use for this filter.

Element 1: Relational Operator

***EQ:** All events with an ASP identifier that matches the specified ASP will be included in the data collected by PEX.

***NE:** All events with an ASP identifier that matches the specified ASP will be discarded. These events will not show up in the *MGTCOL object or the PEX database.

Element 2: ASP Identifier

ASP-identifier: The ASP identifier to use as a compare value for this filter. This ASP identifier can be set to the name of an independent ASP or to the ASP number that corresponds to the ASP value as shown on the WRKDSKSTS command or in the output of PRTPEXRPT of type *TRACE.

IPFTR Specifies the IP (internet protocol) information to use as a compare value for this filter.

Element 1: Relational Operator

***EQ:** All events with IP data that match the filter compare values will be included in the data collected by PEX.

***NE:** All events with IP data that match the filter compare values will be discarded. These events will not show up in the *MGTCOL object or the PEX database.

Element 2: Address Family

***INET:** The Internet protocol will be used as part of the compare value.

***INET6:** The Internet protocol version 6 will be used as part of the compare value.

***UNIX:** The Unix protocol will be used as part of the compare value.

Element 3: Communication Type

***ALL:** All communication types will pass this part of the IP filter.

***STREAM:** A communication type of SOCK_STREAM will be used as the compare value.

***DGRAM:** A communication type of SOCK_DGRAM will be used as the compare value.

***RAW:** A communication type of SOCK_RAW will be used as the compare value.

***SEQPACKET:** A communication type of SOCK_SEQPACKET will be used as the compare value.

Element 4: Local IP Address

***ALL:** All local IP addresses will pass this part of the IP filter.

local-IP-address: The local IP address to be used as part of the IP compare value.

Element 5: Remote IP Address

***ALL:** All remote IP addresses will pass this part of the IP filter.

remote-IP-address: The remote IP address to be used as part of the IP compare value.

Element 6: Local Port Number

***ALL:** All local ports for the specified local address will pass this part of the IP filter.

local-port: The local port number to be used as a compare value.

Element 7: Remote Port Number

***ALL:** All remote ports for the specified address will pass this part of the IP filter.

remote-port: The remote port number to be used as the compare value.

USRDFNFTR

Specifies user defined comparisons to use for this filter. This type of filter will require help from IBM service.

Element 1: Relational Operator

***EQ:** All events for the specified user filter will be collected if the event data matches the compare value.

***NE:** All events for the specified user filter will be discarded if the event data matches the compare value. These events will not show up in the *MGTCOL object or the PEX database.

***GT:** All events for the specified user filter will be collected if the event data is greater than the compare value.

***LT:** All events for the specified user filter will be collected if the event data is less than the compare value.

Element 2: Event Type

event-type: The event type (1-31) for the event to filter.

Element 3: Event Subtype

event-subtype: The event subtype (1-31) for the event to filter.

Element 4: Data Offset

data-offset: The offset into the event data to be compared against the compare value.

Element 5: Data Type Specifies how to compare the event data to the compare value.

***CHAR:** Compare as two character strings, left adjusted and padded on the right with blanks. The maximum length is 30 bytes.

***HEX:** Compare as hexadecimal strings, left adjusted and padded on the right with hexadecimal zeroes. The maximum length is 30 hexadecimal digits.

***INT1:** The first byte of event data at the specified data offset and the compare value are compared as a signed 1-byte integers.

***INT2:** The first two bytes of event data at the specified data offset and the compare value are compared as signed 2-byte integers.

***INT4:** The first four bytes of event data at the specified data offset and the compare value are compared as signed 4-byte integers.

***INT8:** The first eight bytes of event data at the specified data offset and the compare value are compared as signed 8-byte integers.

***UINT1:** The first byte of event data at the specified data offset and the compare value are compared as a unsigned 1-byte integers.

***UINT2:** The first two bytes of event data at the specified data offset and the compare value are compared as unsigned 2-byte integers.

***UINT4:** The first four bytes of event data at the specified data offset and the compare value are compared as unsigned 4-byte integers.

***UINT8:** The first eight bytes of event data at the specified data offset and the compare value are compared as unsigned 8-byte integers.

Element 6: Compare Value

compare-value: The value used to compare against the event data. Up to five compare values can be specified. If multiple values are specified, the comparison will be made with each compare value. If any comparison is true, the event will be filtered.

TEXT Specifies the text that briefly describes the performance explorer filter. More information on this parameter is in *Commonly used parameters*.

***BLANK:** Text is not specified.

'description': Specify no more than 50 characters of text, enclosed in apostrophes.

Examples for ADDPEXFTR

Example 1: Adding Disk and Memory Pool Filters

```
ADDPEXFTR  FTR(FILTER1)
           DSKFTR(*EQ (1 2))
           MEMFTR(*EQ 3)
```

This command adds a new performance explorer filter named FILTER1 which will result in a member named FILTER1 being added to file QAPEXFTR in library QUSRSYS. If this filter is used when starting a performance explorer session (STRPEX command), then events will be collected if they contain disk device identifier data of '1' or '2'. In addition, the pool data for that event must contain '3'. If either the disk data or the memory pool data do not match the specified filter, then that event will not be recorded.

If an event does not contain disk device or memory pool data, then the filter does not apply to that event and those events will be collected. For example, the base event *TASKSWTIN does not contain any disk or memory pool data, so this event would still be collected.

Example 2: Adding a Disk Filter

```
ADDPEXFTR  FTR(DISKFILTER) DSKFTR(*NE (1 2))
```

This command adds a new performance explorer filter named DISKFILTER. If this filter is used when starting a performance explorer session (STRPEX command), then events will be collected if the event contains disk device name data that does not match '1' and does not match '2'.

If an event does not contain disk device name data, then the filter does not apply to that event and those events will be collected. For example, the base event *TASKSWTIN does not contain any disk data, so this event would still be collected.

Example 3: Adding an IP Filter

```
ADDPEXFTR  FTR(IPFILTER)
           IPFTR(*EQ (*INET *STREAM '1.2.3.4'))
```

This command adds a new performance explorer filter named IPFILTER. If this filter is used when starting a performance explorer session (STRPEX command), then events will be collected if a communications event has an address family of *INET, the communication type is *STREAM, and the local IP address is '1.2.3.4'.

Example 4: Adding a User-Defined Filter

```
ADDPEXFTR  FTR(USERFILTER)
           USRDNFTR((*EQ 1 2 20 *CHAR ('BOB' 'SAM')))
```

This command adds a new performance explorer filter named USERFILTER. If this filter is used when starting a performance explorer session (STRPEX command), then events will be collected if the event type is '1', the event subtype is '2' and the data at offset 20 is either 'BOB' or 'SAM'.

Error messages for ADDPEXFTR

*ESCAPE Messages◀

ADDPFCST (Add Physical File Constraint) Command Description

ADDPFCST Command syntax diagram

Purpose

The Add Physical File Constraint (ADDPFCST) command can be used to add constraint relationships to a specified physical file. The four types of constraint relationships that you can add are referential constraints, unique constraints, primary key or check constraints. All constraints are defined at the file level.

You can use referential constraint relationships to define dependencies between files. The relationships that you define are enforced by the system when changes occur to information in the files. When you define constraint relationships you control the **referential integrity** of the data being processed.

To define or establish a referential constraint, the parent file and the dependent file must exist. However, if the parent or dependent file has no members, the constraint only is defined (not established).

When a referential constraint is established, either an access path is created or an existing access path with matching attributes is shared. A maximum of 300 constraint relationships can be established for a file. However, only one primary key constraint can be established for a file.

You can remove a constraint by using the Remove Physical File Constraint (RMVFCST) command. You can view all constraints for a dependent file by using the Display File Description (DSPFD) command.

Restrictions:

1. You cannot add constraint relationships to system files or to program described files.
2. You cannot add a constraint relationship to a file that your user job has open.
3. Referential constraints cannot span auxiliary storage pools (ASPs).
4. Constraints cannot be added to a file in the temporary library QTEMP.
5. If a referential constraint is added with this command and the established referential constraint has records that are in check pending, the constraint is automatically changed to the disabled state.
6. In multithreaded jobs, this command is not threadsafe for distributed files and fails for distributed files that use relational databases of type *SNA.

Required Parameters

FILE Specifies the file to which a constraint is being added. The file must be a physical file and it must allow a maximum of one member (MAXMBRS(1)).

If a referential constraint is being added, this parameter specifies the dependent file and the library containing the dependent file. The parent file is specified on the PRNFILE parameter.

The name of the physical file can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

physical-file: Specify the name of the physical file.

TYPE Specifies the type of constraint being added to the physical file.

***REFCST:** A referential constraint is being added.

Notes:

1. Referential constraints cannot span multiple ASPs (auxiliary storage pools).
2. Referential constraints cannot be added while either the parent or the dependent file is open.
3. Duplicate and multiple referential constraints can be added between the same dependent and parent files if the constraint name is unique. However, the results may not match your expectations. See the Database Programming topic in the Information Center for advisory information on duplicate or multiple referential constraints.

***UNQCST:** A unique constraint is being added.

Note: Duplicate unique constraints are not allowed.

***PRIKEY:** A primary key constraint is being added. A primary key constraint is a special case of a unique constraint.

Note: Only one primary key constraint is allowed per physical file.

***CHKCST:** A check constraint is being added. A check constraint is a field-level validity check of the data in a record of a physical file.

KEY Specifies the constraint key, which is the definition of the access path for the type of constraint specified on the TYPE parameter. The constraint key is one or more fields that exist in the file specified on the FILE parameter. For all constraint types, the fields specified can allow nulls (ALWNULL). If a primary key has at least one null capable field, a check constraint will also be added which will prevent null values from being inserted into the null capable fields. This check constraint cannot be disabled and will be removed when the primary key is removed.

***REFCST**

The foreign key of a referential constraint is defined. If a referential constraint is established, a foreign key access path is added to the dependent file.

***UNQCST**

The key of a unique constraint is defined. If a unique constraint is established, a unique key access path is added to the physical file.

***PRIKEY**

The key of a primary key constraint is defined. If a primary key constraint is established, a primary key access path is added to the physical file.

field-name: Specify the name of the field for the constraint key you are defining. Each field name must exist in the file specified on the FILE parameter. You can specify a maximum of 120 (but no duplicate) field names to define the constraint key, where:

- The field names are of the object type *NAME and are a maximum length of 10.

- The fields must be specified in ascending order.
- The maximum number of bytes in a key is 2000 bytes (see the Database Programming topic in the Information Center for more information on this limitation).

Optional Parameters

CST Specifies the name of the constraint being added.

***GEN:** The system generates a constraint name.

constraint-name: Specify the name of the constraint. The constraint name must be unique to the library of the physical file specified on the FILE parameter. You can specify a maximum of 128 characters without delimiters, or 256 characters with quotation mark (") delimiters. The case is preserved when lowercase characters are specified. See the Database Programming topic in the Information Center for more information on naming conventions for constraints.

PRNFILE

Specifies the parent file and qualifying library of a referential constraint. The file must be a physical file and it must allow a maximum of one member (MAXMBRS(1)).

The name of the parent file can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

parent-file: Specify the name of the parent file of a referential constraint.

PRNKEY

Specifies the parent key, which is the definition of the access path on a parent file of a referential constraint. Only unique constraints or the primary key constraint of a parent file can be used to define a referential constraint. The parent key is one or more fields that exist in the file specified on the PRNFILE parameter.

***PRNFILE:** The access path of the parent file is used when the access path is either a primary key constraint or a unique constraint.

field-name: Specify the name of the field for the constraint key you are defining. Each field name must exist in the file specified on the PRNFILE parameter. You can specify a maximum of 120 (but no duplicate) field names to define the parent key, where:

- The fields can allow nulls.
- The field names are of the object type *NAME and are a maximum length of 10.
- The fields must be in ascending order.
- The fields must match the type and length attributes of the fields specified for the foreign key.
- The maximum number of bytes in a key is 2000 bytes (see the Database Programming topic in the Information Center for more information on this limitation).

DLTRULE

Specifies the delete rule for a referential constraint between a parent file and dependent file. The **delete rule** restricts or defines the effect of deleting a record in both the parent file and the dependent file.

***NOACTION:** The no action delete rule is used. The delete rule is enforced at the end of the delete request. The following are attributes of the no action delete rule:

1. Deleting a record in a parent file is permitted (not restricted) if data for a non-null parent key does not match data for a foreign key.
2. Deleting a record in a parent file is restricted (does not occur) if data for a non-null parent key matches data for a foreign key.

***RESTRICT:** The restrict delete rule is used. The delete rule is enforced at the beginning of the delete request. The following are attributes of the restrict delete rule:

1. Deleting a record in a parent file is permitted if data for a non-null parent key does not match data for a foreign key.
2. Deleting a record in a parent file is restricted if data for a non-null parent key matches data for a foreign key.

***CASCADE:** The cascade delete rule is used. Deleting a record in a parent file causes matching records in the dependent file to be deleted when data for a non-null parent key matches data for a foreign key.

***SETNULL:** The set null delete rule is used. Deleting a record in a parent file updates matching records in a dependent file if data for a non-null parent key matches data for a foreign key. If the matching foreign key field is null-capable, the value is set to null. If the matching foreign key field is not null-capable, the field is not updated.

Note:

To use this rule, a minimum of one field in the foreign key access path must be null-capable.

***SETDFT:** The set default delete rule is used. The following are attributes of the set default delete rule:

1. Deleting a record in the parent file updates matching records in the dependent file when data for a non-null parent key matches data for a foreign key. The matching foreign key values are set to the default value as defined by the default.
2. The default foreign key value must match the corresponding parent key value when there are no null-capable fields.

UPDRULE

Specifies the update rule for a referential constraint between a parent file and dependent file. The **update rule** restricts or defines the effect of updating a record in both the parent file and the dependent file.

***NOACTION:** The no action update rule is used. The update rule is enforced at the end of the update request. The following are attributes of the no action update rule:

1. Updating a record in a parent file is permitted (not restricted) if data for a non-null parent key does not match data for a foreign key.
2. Updating a record in a parent file is restricted (does not occur) if data for a non-null parent key matches data for a foreign key.

***RESTRICT:** The restrict update rule is used. The update rule is enforced at the beginning of the update request. The following are attributes of the restrict update rule:

1. Updating a record in a parent file is permitted if data for a non-null parent key does not match data for a foreign key.
2. Updating a record in a parent file is restricted if data for a non-null parent key matches data for a foreign key.

CHKCST

Specifies a check constraint expression that results in a check constraint. This parameter is only valid for TYPE(*CHKCST).

A check constraint is a validity check placed on fields of a database physical file. The data being inserted or updated into fields with a check constraint must meet the validity check prior to the insert or update of a record. If not all of the validity checks are met, then the write or update request is not performed and a message will be signaled back to the program of the requesting function indicating a check constraint violation.

The check constraint expression has the same syntax as used for SQL. The corresponding SQL term for check constraint expression is a check constraint search condition. For syntax rules, refer to the SQL Reference topic in the Information Center.

Examples for ADDPFCST

Example 1: Adding a Unique Constraint

```
ADDPFCST FILE(MYLIB/LOCATIONS) TYPE(*UNQCST)
KEY(REGION) CST(Personnel_by_REGION)
```

This command adds a unique constraint to the LOCATIONS file in the MYLIB library. The field that defines the access path is REGION. The name of the access path is Personnel_by_REGION.

Example 2: Adding a Referential Constraint

```
ADDPFCST FILE(ADMN/PERSONNEL) TYPE(*REFCST)
KEY(REGION) CST(1994Hires)
PRNFILE(MYLIB/LOCATIONS) PRNKEY(REGION)
DLTRULE(*CASCADE) UPDRULE(*RESTRICT)
```

This command adds a referential constraint to the PERSONNEL file in the ADMN library. The field that defines the access path is REGION, which is also the key for the parent file LOCATIONS in the MYLIB library. The name of the access path is 1994Hires. According to the delete rule of cascade, if a record in the LOCATIONS file is subsequently deleted, and that record matches a record in the PERSONNEL file, the record also will be deleted from the PERSONNEL file. According to the update rule of restrict, subsequent changes to the LOCATION file records defined in the constraint are restricted at the beginning of the update request.

Example 3: Adding a Check Constraint

```
ADDPFCST FILE(PERSONNEL/SALARY) TYPE(*CHKCST)
CST(Upper_Salary_Limit) CHKCST('EMPSAL <= 100000')
```

This command adds a check constraint to the SALARY file in the PERSONNEL library. The check constraint will ensure an employee's salary may be a maximum of 100,000.

Error messages for ADDPFCST

*ESCAPE Messages

CPF32B0

Constraint cannot be added to file &1.

CPF32B7

&3 constraint(s) added to file &1 but constraint(s) in error.

ADDPFM (Add Physical File Member) Command Description

ADDPFM Command syntax diagram

Purpose

The Add Physical File Member (ADDPFM) command adds a named member to the specified physical file, which must already exist on the system. A member must be added to the physical file before the file can have data stored in it. The first member of a file can be added by entering an ADDPFM command or by specifying a member name in the MBR parameter of the Create Physical File (CRTPF) command. To add other members to the file, use the ADDPFM command to specify each one.

The number of members that can be added to the physical file is limited to the number specified in the MAXMBRS parameter of the associated CRTPF command. Each member added has the same attributes as those defined in the physical file, its own set of data records, and its own access path, as specified in the data description specifications (DDS). The access path determines the order in which the records in that member are processed.

Note:

An *EXCLRD lock is required on the file to add a member. Because this command adds a member to a file in a library, the library must not be locked (*SHRNUP or *EXCLRD in the Allocate Object command) for another job.

Restrictions

1. In multithreaded jobs, this command is not threadsafe for distributed files and fails for distributed files that use relational databases of type *SNA. This command is also not threadsafe and fails for Distributed Data Management (DDM) files of type *SNA, when SYSTEM(*RMT) or SYSTEM(*FILETPYE) is specified.

Required Parameters

FILE Specifies the qualified name of the physical file to which this member is added.

The name of the physical file can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

physical-file-name: Specify the name of the file to which the member is added.

If a DDM file is specified, the request is valid only to a target System/38 or iSeries 400.

MBR Specifies the name of the physical file member being added. The member name must be unique in the file to which it is added.

If a DDM file is specified on the FILE parameter, and a member name is specified as part of the remote file name of the DDM file, the MBR name specified must match the member name in the remote file name in the DDM file.

Optional Parameters

SRCTYPE

Specifies the source type of a member if this is a source file. The source type option is a character string of no more than 10 characters representing a name. The first character must be alphabetic (including the characters \$, @, or #), and the remaining characters must be alphanumeric or an underscore character.

Notes:

1. The user of this command must ensure the validity of the source type option.
2. The source type option can only be used with the Add Physical File Member (ADDPFM) command to add the source type attribute for a source file member.

***NONE:** No source type is specified.

source-type: Specify the source type of a member.

EXPDTE

Specifies the expiration date. The files cannot be overwritten until the expiration date. The expiration date must be later than or equal to the current date.

Note:

An attempt to open a file member that has exceeded its expiration date causes an error message to be sent. (The RMVM command is used to remove the member from the file.)

***NONE:** No expiration date is specified.

expiration-date: Specify the date after which the member cannot be used. The expiration date must be specified in the format defined by the job attributes, DATFMT and DATSEP. The date must be enclosed in apostrophes if special characters are used in the format.

SHARE

Specifies whether the open data path (ODP) for the physical file member is shared with other programs in the routing step. When an ODP is shared, the programs accessing the file share facilities such as the file status and the buffer.

More information on shared database files is in the Database Programming topic in the Information Center.

***NO:** The ODP created by the program with this attribute is not shared with other programs in the routing step. Every time a program opens the file with this attribute, a new ODP to the file is created and activated.

***YES:** The ODP created with this attribute is shared with each program in the routing step that also specifies SHARE(*YES) when it opens the file.

Note:

When SHARE(*YES) is specified and control is passed to a program, a read operation in that program retrieves the next input record. A write operation produces the next output record.

TEXT Specifies the text that briefly describes the physical file member. More information is in Commonly used parameters.

***BLANK:** Text is not specified.

'description': Specify no more than 50 characters of text, enclosed in apostrophes.

Example for ADDPFM

```
ADDPFM FILE(INVENTX) MBR(MONDAYTX)
      TEXT('Monday's Inventory Transactions')
```

This command adds a member named MONDAYTX to the physical file named INVENTX. The library list (*LIBL) is used to find the file because the FILE value is not qualified by a library name. The size of the member and the storage allocation values assigned to this member were specified in the CRTPF command that created the physical file. The text, *Monday's Inventory Transactions*, describes this member of the INVENTX file.

Error messages for ADDPFM

*ESCAPE Messages

CPF3204

Cannot find object needed for file &1 in &2.

CPF7306

Member &1 not added to file &2 in &3.

ADDPFTRG (Add Physical File Trigger) Command Description

ADDPFTRG Command syntax diagram

Purpose

The Add Physical File Trigger (ADDPFTRG) command adds a system trigger to a specified physical file. A trigger defines a program that is called when a delete, insert, update or read operation occurs for a file.

The trigger program can be specified to be called before or after an operation occurs. The operation can be an insert, update, delete or read operation through any interface. Operations do not include clearing, initializing, moving, applying journal changes, removing journal changes, or changing end of data operations.

A maximum of 300 triggers can be added to one physical file. The trigger program to be called can be the same for each trigger or it can be a different program for each trigger.

An exclusive-no-read lock is held on the physical file when adding a trigger to that file. All logical files which are built over the physical file are held with an exclusive-no-read lock.

Once a trigger is added to the physical file, all members of that specified file are affected by the trigger. When a change operation occurs on a member of the specified file, the trigger program is called. The trigger program is also called when a change operation occurs by way of either a dependent logical file or a Structured Query Language (SQL) view that is built over the physical file.

More information on the trigger program is in the Database Programming topic in the Information Center.

Restrictions

1. You must have read authority, object operational, and object management or object alter authority to the physical file, execute authority to the file library, update and object operational authority to the physical file if ALWREPCHG(*YES) has been specified, and execute authority to the trigger program and its library.
2. If the physical file or a dependent logical file or SQL view is opened in this or another job, a trigger cannot be added.
3. While this command is running, neither the physical file nor any dependent logical files can be opened.

4. The trigger program must be a program of object type *PGM. It cannot be an Integrated Language Environment* (ILE*) service program of object type *SRVPGM.
5. In multithreaded jobs, this command is not threadsafe for distributed files and fails for distributed files that use relational databases of type *SNA.
6. A trigger program cannot be added if the program is in the QTEMP library.

Required Parameters

FILE Specifies the qualified name of the physical file to which this trigger program is added. The file must exist on the system.

The name of the physical file can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

physical-file-name: Specify the name of the file to which the trigger program is added.

TRGTIME

Specifies the time when the trigger program is called.

***BEFORE:** The trigger program is called before the change operation on the specified physical file.

***AFTER:** The trigger program is called after the change operation on the specified physical file.

TRGEVENT

Specifies the event (the operation to the physical file) that calls the trigger program. Only one event can be specified for each command issued.

***INSERT:** An insert operation calls the trigger program.

Note:

If the physical file is not read and write capable, the *INSERT value cannot be specified.

***DELETE:** A delete operation calls the trigger program.

Notes:

1. If the physical file is not read and delete capable, the *DELETE value cannot be specified.
2. If the physical file has a referential constraint with a delete rule of CASCADE, the *DELETE value cannot be specified.

***UPDATE:** An update operation calls the trigger program.

Notes:

1. If the physical file is not read and update capable, the *UPDATE value cannot be specified.
2. If the physical file is a dependent file which has a foreign key with a delete rule of SET NULL or SET DEFAULT, the *UPDATE value cannot be specified.

***READ:** A read operation calls the trigger program.

Notes:

1. If the physical file is not read capable, the *READ value cannot be specified.
2. *READ can be specified as the trigger event only when *AFTER has been specified for the trigger time.

PGM Specifies the name of the program that is called when the specified event occurs on the physical file. The program must exist on the system and be of object type *PGM.

The name of the trigger program can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

Note: The special values *LIBL and *CURLIB are the values of the job running when the trigger program is added.

program-name: Specify the name of the program to be called when the specified event occurs on the specified physical file.

Optional Parameters

TRG Specifies the name of the trigger. This is to distinguish between triggers with the same trigger time and trigger event. The trigger name must be unique to the library of the trigger. You can specify a maximum of 128 characters without delimiters or 258 characters with quotation mark delimiters.

***GEN:** A trigger name will automatically be generated.

trigger-name: Specify the name of the trigger to be added.

TRGLIB

Specifies the name of the library where the trigger is to be added.

***FILE:** The library for the file specified on the FILE parameter is used.

***CURLIB:** The current library is used. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be used.

RPLTRG

Specifies whether an existing trigger is replaced by the trigger to be added when the triggers have the same trigger name or trigger time.

***NO:** The existing trigger is not replaced.

***YES:** The existing trigger is replaced. If *GEN was specified for the trigger name and the time and event match a single entry, the trigger will be replaced. If a trigger name was specified and it matches an existing entry, the trigger will be replaced. If a trigger with the specified trigger name does not exist, the new trigger is added to the physical file.

ALWREPCHG

Specifies whether repeated changes to a record within a trigger are allowed.

***NO:** Repeated changes to a record within a trigger are not allowed.

***YES:** Repeated changes to a record within a trigger are allowed.

This value is ignored if *READ is specified for the TRGEVENT parameter.

TRGUPDCND

Specifies the condition under which an update event calls the trigger program.

Note:

This parameter applies only when *UPDATE is specified on the TRGEVENT parameter.

***ALWAYS:** The trigger program is called whenever a record is updated, whether or not a value changes.

***CHANGE:** The trigger program is called only when a record is updated and a value is changed.

THDSAFE

Specifies whether the trigger program is threadsafe. This is intended for documentation purposes only. It may be used in determining the MLTTHDACN value, but there is no direct relationship between the THDSAFE and MLTTHDACN keywords.

***UNKNOWN:** The threadsafe status of the trigger program is not known.

***NO:** The trigger program is not threadsafe.

***YES:** The trigger program is threadsafe.

MLTTHDACN

Specifies the action to take when the trigger program is called in a multithreaded job. The THDSAFE attribute of the trigger program can be used in determining the action, however, there is no direct relationship between the THDSAFE and MLTTHDACN keywords.

***SYSVAL:** Use the QMLTTHDACN system value to determine the action to take.

***MSG:** Run the trigger program in a multithreaded job, but send a diagnostic message.

***NORUN:** Do not run the trigger program in a multithreaded job. Send an escape message.

***RUN:** Run the trigger program in a multithreaded job.

If you do use the THDSAFE value to determine the value for MLTTHDACN, please read the following recommendations:

1. If the THDSAFE value is *NO, MLTTHDACN should be set to *NORUN.
2. If the THDSAFE value is *UNKNOWN, MLTTHDACN should be set to *SYSVAL.
3. If the THDSAFE value is *YES, MLTTHDACN should be set to *RUN.

Examples for ADDPFTRG

Example 1: Adding a Trigger for an Insert Event

```
ADDPFTRG FILE(EMP) TRGTIME(*AFTER) TRGEVENT(*INSERT)
PGM(LIB2/INSTRG)
```

This command adds a trigger with trigger program INSTRG in library LIB2 to the physical file named EMP. When an insert operation occurs on the EMP file, the program INSTRG is called after the insert operation. The library list (*LIBL) is used to find the file because the FILE value is not qualified by a library name.

Example 2: Setting Multiple Trigger Events to Call One Trigger Program

```
ADDPFTRG FILE(EMP) TRGTIME(*AFTER) TRGEVENT(*INSERT)
PGM(LIB2/INSTRG)
```

```
ADDPFTRG FILE(EMP) TRGTIME(*AFTER) TRGEVENT(*UPDATE)
PGM(LIB2/INSTRG)
```

These two commands add triggers to call the trigger program INSTRG in library LIB2 when an insert or update operation occurs on the EMP file.

Example 3: Adding a Trigger Only When an Update Event Changes Values

```
ADDPFTRG FILE(EMP) TRGTIME(*BEFORE) TRGEVENT(*UPDATE)
PGM(LIB2/UPDTRG) TRGUPDCND(*CHANGE)
```

The trigger program UPDTRG in library LIB2 is called before a value for a field of a record in the EMP file changes during an update.

Example 4: Replacing an Existing Trigger

```
ADDPFTRG FILE(EMP) TRGTIME(*BEFORE) TRGEVENT(*UPDATE)
PGM(LIB2/NEWPGM) RPLTRG(*YES) TRGUPDCND(*CHANGE)
```

The trigger program NEWPGM being added to the file EMP has the same trigger time (*BEFORE) and trigger event (*UPDATE) as the trigger program UPDTRG that was added in Example 3. Therefore, the added trigger program NEWPGM replaces the existing trigger program UPDTRG.

Example 5: Replacing a Trigger with a Trigger for a Different Update Condition

```
ADDPFTRG FILE(EMP) TRGTIME(*BEFORE) TRGEVENT(*UPDATE)
PGM(LIB2/NEWPGM) RPLTRG(*YES) TRGUPDCND(*ALWAYS)
```

The trigger added in Example 4 that calls the trigger program NEWPGM only if the values are changed, is replaced by a trigger that always calls the trigger program NEWPGM regardless of the values.

Error messages for ADDPFTRG

*ESCAPE Messages

CPF32C6

Trigger operation not successful.

ADDTCPPTP (Add Point-to-Point TCP/IP Profile) Command Description

ADDTCPPTP Command syntax diagram

Purpose

The Add Point-to-Point TCP/IP Profile (ADDTCPPTP) command is used to create a simple PPP (point-to-point protocol) connection profile. Profiles can be created to answer incoming calls by specifying OPRMODE(*ANS), or to dial remote systems by specifying OPRMODE(*DIAL).

The preferred method of creating point-to-point profiles is through iSeries Navigator, since it supports all features of PPP. In cases where it is not feasible to create a point-to-point profile using iSeries Navigator, the ADDTCPPTP command can be used to create a simple point-to-point profile.

The ADDTCPPTP command supports only a limited set of PPP profile features. The following assumptions and restrictions apply:

- Profiles will be created for use with PPP protocol only.

- Profiles created will only support one incoming or outgoing call at a time.
- Profiles created will be defined for use with a single switched asynchronous PPP line. The PPP line description to be used with this profile will be created based on information in the ADDTCPPTP command and will be created with the following name: QPPPnnnnxx, where 'nnnn' is the hardware resource name and 'xx' is the resource number. If the resource name is too large to use then a resource name of 'CMN' will be used. If a line with this name already exists when profile is created then the previously created PPP line will be used.
- Answer profiles with PPP authentication enabled will only support up to 10 remote user validation list entries using this command. Additional entries can be added with iSeries Navigator.

If the ADDTCPPTP command fails then the PPP line description or validation list associated with the profile will not be created if they did not previously exist.

Restrictions:

You must have *IOSYSCFG special authority to use this command.

Required Parameters

CFGPRF

Specifies the name of the Point-to-point configuration profile to create.

configuration-profile-name: Specify the name of a point-to-point configuration profile.

OPRMODE

The mode of operation for this point-to-point connection profile.

***ANS:** The profile is defined to answer calls from a remote system.

***DIAL:** The profile is defined to dial a remote system.

Optional Parameters

RSRCNAME

Specifies the communications hardware resource to be used by this profile.

***CALC:** The resource name will be determined as follows:

- Look for resources being used by the 2771 integrated modem. If only one 2771 is defined, use that resource for this PPP line. *CALC is not valid if more than one 2771 modem is defined.
- If a 2771 cannot be used, determine if any resources are defined for use by ECS (Electronic Customer Support). If an ECS resource is available, use that resource for this PPP line.
- If neither a single 2771 integrated modem or ECS resource is available, the resource cannot be calculated and it will have to be explicitly defined.

resource-name: Specify the resource name.

Note:

Use the Work With Hardware Resources (WRKHDWRSC) command with *CMN specified for the TYPE parameter to help determine the resource name.

MODEM

The name of the modem description to use for this point-to-point profile.

***RSRCNAME:** The modem name will be determined based on the value defined for the RSRCNAME parameter.

- If the resource is defined to use the 2771 integrated modem, the '2771 Internal Modem' description will be used.

- If the resource is defined to use the 2772 integrated modem, the '2772 Internal Modem' description will be used.
- If the resource is defined to use the 2761 internal modem, the '2761 Internal Modem' description will be used.
- If the ECS resource was chosen, the 'IBM 7852-400' modem description will be used.
- Otherwise, if the resource does not have a pre-defined modem description, MODEM(*RSRCNAME) cannot be used and the modem description will have to be explicitly defined.

***SELECT:** A list of modems is shown from which you will select the modem to use. This option is only valid when running the ADDTCPPTP CL command in interactive mode, otherwise an error will occur. If you are running interactively, it is recommended that you use the *SELECT value to help ensure that you properly select the modem to use.

'modem-identification': Specify the name of the modem to use. Note that modem names are case sensitive and must match exactly to the modems defined for the system.

'generic-modem-identification':* Specify the generic name of the modem you wish to use. A generic modem name is a character string of one or more characters followed by an asterisk (*); for example, 'abc*'. If a generic name is specified, then the **first** modem name that matches with the generic name will be used. It is recommended that you include as many characters in the modem name string as possible to avoid any ambiguity. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete modem name. The actual modem name chosen will be posted in a message in the job log.

CALLNBR

Specifies the telephone number of the remote system to call. This parameter is only used when OPRMODE(*DIAL) is defined.

***NONE:** No telephone number is defined. If OPRMODE(*DIAL) is specified, a value other than *NONE must be defined for this parameter.

telephone-number: Specify the telephone number to call to connect to the remote system. If additional numbers are required to establish an outside call, they must also be specified. Special character ',' (comma) may be used to signify if a delay is required before dialing the next number. Typically this delay is one second for most modems.

LCLINTNETA

Specifies the internet address of the iSeries 400 to be used for this PPP connection.

***OPRMODE:** The mode of operation will determine the local IP address. If OPRMODE(*DIAL) is specified, the local IP address will be defined as *DYNAMIC. If OPRMODE(*ANS) is specified, the local IP address will be defined as *CURRENT.

***DYNAMIC:** The IP address will be defined by the remote system during PPP negotiations.

***CURRENT:** The current local IP address will be used. This address is determined as follows:

- If a local host name has been defined using the Change TCP/IP Domain (CHGTCPDMN) command, this host name will be resolved to an IP address (either defined in the local host table or by a domain name server). If this IP address is found to also exist on the local iSeries 400, it will be used.
- If no local host name is defined, or could not be resolved to, the first valid local iSeries 400 IP address found will be used.
- If no valid local IP addresses are found, *CURRENT will not be allowed.

internet-address: Specify the internet address to use as the local IP address for this PPP profile. The address specified here can already exist on the iSeries 400 or a unique IP address can be defined.

RMTINTNETA

Specifies the internet address of the remote system to use for this PPP connection.

***OPRMODE:** The mode of operation will determine the remote IP address. If OPRMODE(*DIAL) is specified, the remote IP address will be defined as *DYNAMIC. If OPRMODE(*ANS) is specified, the remote IP address will be defined as '169.254.x.x' where 169.254.x.x is a reserved IANA LINKLOCAL network address. The actual host portion (x.x) of this address will be determined at runtime.

***DYNAMIC:** The IP address will be defined by the remote system during PPP negotiations.

internet-address: Specify the internet address to use as the remote IP address for this PPP profile. The address specified here MUST be an unique IP address for this iSeries 400.

ENBPPPAUT

Specifies whether PPP authentication will be enabled for this profile.

***NO:** No PPP authentication will be required to either connect to the remote system or to allow the remote system to connect to the iSeries 400.

***YES:** PPP authentication will be required to either connect to the remote system or to allow the remote system to connect to the iSeries 400.

PPPAUT

Specifies the PPP authentication values to use for PPP authentication. This parameter is only in effect if ENBPPPAUT(*YES) is defined.

Element 1: User Name and Password

Specifies the user names and passwords to use for PPP authentication. Only one user name and password can be specified for OPRMODE(*DIAL) profiles. Up to ten user names and passwords can be specified for OPRMODE(*ANS) profiles. Additional entries can be added with iSeries Navigator.

user-name: Specify the user name to be used for PPP authentication. Each user name can be up to 64 characters in length. User names are case sensitive and will be stored exactly as they are entered.

password: Specify the password associated with the user name. Each password can be up to 64 characters in length. Passwords are case sensitive and will be stored exactly as they are entered.

Element 2: Authentication Protocol

Specifies which authentication protocol to use for PPP authentication. This value also specifies which authentication protocol to associate with each specified user name and password.

***ENCRYPTED:** Specifies that only PPP authentication methods using encrypted passwords will be used. This value will support both EAP (Extended Authentication Protocol) and CHAP (Challenge Authentication Protocol - using MD-5 hashing algorithm).

***UNENCRYPTED:** Specifies that only PPP authentication methods using unencrypted passwords will be used. Currently this is limited to PAP (Password Authentication Protocol). Note that this method of authentication is not as secure as CHAP or EAP since it allows user name and password information to flow over the link unprotected.

Element 3: Validation List

Specifies the validation list used to store remote user name and password information for OPRMODE(*ANS) profiles. This element is ignored if OPRMODE(*DIAL) is specified. All validation lists defined for use by PPP must exist in library QUSRSYS.

***CFGPRF:** Specifies that the validation list to store user name and password information is the same name as the point-to-point profile. If the validation list does not exist, it will be created in library QUSRSYS.

validation-list-name: Specifies the name of the validation list in library QUSRSYS. If the validation list does not exist, it will be created.

INACTTMR

Specifies the time (in seconds) that the system waits for user data activity for this profile before disconnecting. This timer is started once LCP (Link Control Protocol) and NCP (Network Control Protocol) negotiations have completed successfully, and restarted when user data is sent or received. LCP and NCP packets do not cause this timer to be restarted.

***NOMAX**: The inactivity timer is disabled.

timer-value: Specify a value ranging from 15 through 65535 seconds.

TEXT Allows you to specify a text description for the point-to-point profile.

***BLANK**: No text is specified.

'description': Specify no more than 50 characters of text, enclosed in apostrophes.

FULLMASQ

Specifies whether full masquerading will be used. If enabled, all IP addresses will be hidden behind the remote IP address for the dial PPP connection. This parameter is only in effect if OPRMODE(*DIAL) is specified. This feature allows all outbound IP traffic to have its source IP address translated to the remote IP address defined for the PPP link. The source port is also modified, so that return IP traffic can be properly associated with the correct conversation and have its IP address and destination port changed back to the correct values.

This feature is particularly useful to allow other hosts on the same network as the iSeries 400 to use the iSeries 400 as a gateway to the internet. If the iSeries 400 is connected to the internet using an ISP (Internet Service Provider), the other hosts, such as PCs, could also gain access to the internet and 'hide' behind the iSeries 400's assigned PPP IP address.

***NO**: No IP addresses will be hidden (masqueraded) behind the iSeries 400's PPP IP address.

***YES**: All IP address will be hidden (masqueraded) behind the iSeries 400's PPP IP address when traffic flows out the PPP link.

IPDTGFWD

Specifies whether IP datagram forwarding is enabled for this PPP connection. This parameter is only in effect if OPRMODE(*ANS) is specified.

***NO**: Internet Protocol (IP) will discard those datagrams from the remote system that are not destined for any addresses local to this iSeries 400.

***YES**: This allows Internet Protocol (IP) datagrams not destined for this iSeries 400 to pass through this system onto a connected network. Enabling IP datagram forwarding essentially enables the iSeries 400 to act as router for this connection. Careful security considerations should be reviewed prior to enabling IP forwarding for the PPP link. Note that this will only take effect if system wide IP datagram forwarding is enabled, otherwise it will be ignored even if marked. System-wide IP datagram forwarding is controlled by the IPDTGFWD parameter on the CHGTCPA (Change TCP/IP Attributes) command.

ALWRMTOVR

Specifies whether remote systems will be allowed to override the remote IP address defined in RMTINTNETA. This parameter is only in effect if OPRMODE(*ANS) is specified and RMTINTNETA(*DYNAMIC) is not specified.

***NO**: If a specified remote IP address is specified for RMTINTNETA, remote systems will not be allowed to define their own address. The remote system must use the address defined by the iSeries 400 or the PPP connection will be terminated.

***YES**: If a specified remote IP address is specified for RMTINTNETA, remote systems will still be allowed to define their own address. This is useful if you want to allow more than one type of remote client to be able to dial into the iSeries 400. Typically the remote system dialing in will

request that it be told what its IP address is. By specifying ALWRMTOVR(*YES), you tell remote clients what their IP address should be, but also allow other remote clients to specify their own address without the need for an additional profile or resource.

Examples for ADDTCPPTP

Example 1: Create an Answer Profile

```
ADDTCPPTP CFGPRF(ANSPROFILE) OPRMODE(*ANS)
```

This command will create answer PPP profile with the following properties

- The resource will be calculated and the modem description will be determined by the resource. Assuming a 2771 integrated modem is found then the '2771 Internal modem' modem description will be used.
- Line description 'QPPPCMNxx' will be created, where CMNxx is the 2771 resource.
- An existing IP address on the iSeries 400 will be defined as the local IP address. If there is an IP address associated with the local host name then this address will be used. If not, then the first local IP address found for the iSeries 400 will be used.
- The remote IP address (address that is assigned to the remote system) will be defined as 169.254.x.x, where x.x is determined at runtime.
- Authentication is not enabled.

Example 2: Create a PPP Dial Profile

```
ADDTCPPTP CFGPRF(DIALPROF) OPRMODE(*DIAL) RSRNAME(CMN14)
MODEM('2761 Internal Modem') CALLNBR('1,,9876543')
ENBPPAUT(*YES)
PPAUT(((dialuser dialpw)) *ENCRYPTED *CFGPRF)
FULLMASQ(*YES)
```

This command will create a dial PPP profile with the following properties:

- The profile will use a PPP line named 'QPPPCMN14', defined to use communication resource CMN14.
- The 2761 internal modem will be used (Modem name as seen in CFGTCPPTP, option 11).
- When calling the remote system, a '1' will be dialed first (possibly to reach an outside line), then there will a 2 second delay (approximately), then telephone number '9875432' will be called.
- Authentication is enabled and an authentication protocol using encryption will be used (EAP - extended authentication protocol or CHAP - Challenge authentication protocol (MD-5)). The user name and password defined will be used for authentication.
- The local and remote IP addresses will be defined as *DYNAMIC, which means the addresses will be defined by the remote system during the IPCP (Internet Protocol Control Protocol) negotiation phase of the PPP connection.
- All IP traffic going out the PPP link will appear as if it originated from the iSeries 400.

Example 3: Create Profile Using Predefined IP Addresses

```
ADDTCPPTP CFGPRF(ANSPROFILE) OPRMODE(*ANS) RSRNAME(CMN10)
MODEM('USRobotics 56K*')
LCLINTNETA('10.9.8.1') RMTINTNETA('10.9.8.2')
ENBPPAUT(*YES)
PPAUT(((RmtID1 RmtPW1) (RmtID2 RmtPW2)) *ENCRYPTED PPPVLDL)
TEXT('PPP *ANS profile')
IPDTGFW(*YES)
```

This command will create an answer PPP profile with the following properties:

- The profile will use a PPP line named 'QPPPCMN10', defined to use communication resource CMN10.

- Modem name of 'USRobotics 56K V.90 Sportster' will be used assuming it is the first modem name found starting with the string 'USRobotics 56K'. The actual modem selected will be posted to the joblog in a message.
- Authentication is enabled and an authentication protocol using encryption will be used (EAP - extended authentication protocol or CHAP - Challenge authentication protocol (MD-5)). Both user RmtID1 and RmtID2 are authorized to connect using this profile.
- Validation list PPPVLDL in library QUSRSYS will be used to store the user names and passwords.
- The local IP address will be 10.9.8.1 and the remote IP address will be 10.9.8.2. The 10.9.8.1 local address is an existing IP address on the iSeries 400 and is attached to the 10.9.8.0 network.
- The remote system will be allowed to directly access the 10.9.8.0 network.

Error messages for ADDTCPPTP

*ESCAPE Messages

TCP83D0

Point-to-point profile &1 not added.

ADDPJE (Add Prestart Job Entry) Command Description

ADDPJE Command syntax diagram

Purpose

The Add Prestart Job Entry (ADDPJE) command adds a prestart job entry to the specified subsystem description. The entry identifies prestart jobs that are started when the subsystem is started or when the start prestart job command is entered.

Restriction: This command is restricted to a user with *USE and object management authorities for the subsystem description and *USE authority for the user profile and the job description.

Required Parameters

SBSD Specifies the qualified name of the subsystem description to which the prestart job entry is being added.

The name of the subsystem description can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

subsystem-description-name: Specify the name of the subsystem description.

PGM Specifies the qualified name of the program run by the prestart job. This program name is used to match an incoming request with an available prestart job. Two entries with the same program name can exist in a single subsystem description, but they must have different library names. If the program does not exist when the entry is added, a library qualifier must be specified because the qualified name is retained in the subsystem description.

The name of the the program can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.


program-name: Specify the name of the program to be run by the prestart job.

Optional Parameters

USER Specifies the name of the user profile under which the prestart job is initiated. In addition, the current user profile of the prestart job is set to this user whenever the job waits for a request to handle.

Note:

When a prestart job is given a request to handle, the current user profile of the job is updated. Refer to the

Work Management  book for information on how this profile is determined. This change in current user profile is for authority checking only. None of the other attributes of the user profile, such as the current library (CURLIB) or the initial program to call (INLPGM), are given to the prestart job.

QUSER: The IBM-supplied QUSER user profile is used.

user-profile-name: Specify the name of the user profile used for the prestart job.

JOB Specifies the name of the prestart job that is started.

***PGM:** The prestart job name is the same as the program name specified by the PGM parameter.

job-name: Specify the name of the prestart job.

JOB Specifies the qualified name of the job description used for the prestart job. If the job description does not exist when the entry is added, a library qualifier must be specified because the qualified job description name is retained in the subsystem description.

***USRPRF:** The job description name specified in the user profile for the USER parameter is used.

***SBSD:** The job description having the same name as that of the subsystem description named in the SBSDB parameter is used.

The name of the job description can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

job-description-name: Specify the name of the job description being used for this prestart job. If no library is specified, the library list (*LIBL) of the job where this command is run is used to find the job description.

STRJOBS

Specifies whether prestart jobs are started at the time the subsystem is started.

***YES**: The prestart jobs are started when the subsystem is started.

***NO**: The prestart jobs are not started when the subsystem is started. The Start Prestart Jobs (STRPJ) command must be used to start these prestart jobs.

INLJOBS

Specifies the initial number of prestart jobs started when the subsystem specified in the SBSDD parameter is started.

Notes:

1. The value of this parameter must be less than or equal to the value of the MAXJOBS parameter.
2. The value of this parameter must be greater than or equal to the value of the THRESHOLD parameter.
- 3**: Three prestart jobs are started when the subsystem is started.

initial-active-jobs: Specify the initial number of prestart jobs that are started when the subsystem is started. Valid values range from 1 through 9999.

THRESHOLD

Specifies the threshold number of available prestart jobs at which additional prestart jobs are started. When the pool of available prestart jobs (jobs available to service requests) is reduced below this number, more prestart jobs (specified by the ADLJOBS parameter) are started and added to the pool of available prestart jobs.

Note:

The value of this parameter must be less than or equal to the value of the INLJOBS parameter.

2: When one prestart job is available, start the number of jobs specified by the ADLJOBS parameter.

threshold-value: Specify the minimum number of prestart jobs that must be available before additional prestart jobs are started. Valid values range from 1 through 9999.

ADLJOBS

Specifies the additional number of prestart jobs started when the number of prestart jobs drops below the THRESHOLD parameter.

Note:

The value of this parameter must be less than the value of the MAXJOBS parameter.

2: Two additional prestart jobs are started.

additional-active-jobs: Specify the number of additional prestart jobs to be started. Valid values range from 0 through 999.

MAXJOBS

Specifies the maximum number of prestart jobs that can be active on the subsystem at the same time under this prestart job entry.

Notes:

1. The value of this parameter must be greater than or equal to the value of the INLJOBS parameter.
2. The value of this parameter must be greater than the value of the ADLJOBS parameter.

***NOMAX:** There is no maximum number of jobs that can be active at the same time.

maximum-jobs: Specify the maximum number of prestart jobs that can be active at the same time. Valid values range from 1 through 9999.

MAXUSE

Specifies the maximum number of requests that can be handled by each prestart job in the pool before the job controlled by the system is ended.

200: A prestart job for this entry can service up to 200 requests before it is ended. If *NOMAX is specified, the prestart jobs may end abnormally for one of the following reasons:


- The job log has exceeded the maximum size.
- The allowed maximum for spooled files has been reached.
- The allowed maximum for CPU time has been reached.
- The allowed maximum for temporary storage has been reached.

***NOMAX:** There is no maximum number of jobs that can be active at the same time.

maximum-uses: Specify the maximum number of requests that a prestart job can handle before it is ended. Valid values range from 1 through 1000.

WAIT Specifies whether program start requests either wait for a prestart job to become available or are rejected if a prestart job is not immediately available when the program start request is received.

Note:

Refer to the CL Programming  book to determine the time-out considerations for the communications type being used.

***YES:** Program start requests wait either until there is an available prestart job, or until a prestart job is started, to service the request.

***NO:** Program start requests are rejected if a prestart job is not immediately available when the program start request is received.

POOLID

Specifies the identifier of the subsystem pool in which the prestart jobs run.

1: The prestart jobs run in the first pool.

pool-identifier: Specify the identifier of the subsystem pool in which the prestart jobs run. Valid values range from 1 through 10.

CLS Specifies the names of the classes under which the prestart jobs run and how many prestart jobs are allowed to run under each class. Jobs start by using the first class. After the number of jobs specified for the first class is reached, jobs are started under the second class. If the class does not exist when the entry is added, a library qualifier must be specified because the qualified class name is retained in the subsystem description.

Element 1: Class Name of First Class

***SBSD:** The class having the same name as the subsystem description specified in the SBSD parameter is used for prestart jobs.

The name of the class can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

class-name: Specify the name of the class being used for prestart jobs. If no library qualifier is specified, the library list (*LIBL) of the job where this command is run is used to find the class.

Element 2: Number of Jobs Using First Class

***CALC:** The system calculates how many prestart jobs use this class. If one class is specified and *CALC is specified, all of the jobs use the specified class. If two classes are specified and *CALC is specified for both, the first class is the value of the MAXJOBS parameter divided by two, and the second class is the value of the MAXJOBS parameter minus the value calculated for the first class. If a specific number of jobs is specified for one class and *CALC is specified for the other class, the system calculates the difference between MAXJOBS and the specific number of jobs for the *CALC designation.

***MAXJOBS:** All of the prestart jobs use the specified class.

number-of-jobs: Specify the number of prestart jobs that use this class. The sum of the values specified for both classes must total the value specified for the MAXJOBS parameter.

Element 3: Class Name of Second Class

***NONE:** One class is used.

***SBSD:** The class having the same name as the subsystem description specified in the SBSD parameter is used for prestart jobs.

The name of the class can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

class-name: Specify the name of the class being used for prestart jobs. If no library qualifier is specified, the library list (*LIBL) of the job where this command is run is used to find the class.

Element 4: Number of Jobs Using Second Class

***CALC:** The system calculates how many prestart jobs use this class. If one class is specified and *CALC is specified, all of the jobs use the specified class. If two classes are specified and *CALC is specified for both, the first class is the value of the MAXJOBS parameter divided by two, and the second class is the value of the MAXJOBS parameter minus the value calculated for the first class. If a specific number of jobs is specified for one class and *CALC is specified for the other class, the system calculates the difference between MAXJOBS and the specific number of jobs for the *CALC designation.

***MAXJOBS:** All of the prestart jobs use the specified class.

number-of-jobs: Specify the number of jobs that use this class. The sum of the values specified for both classes must total the value specified for the MAXJOBS parameter.

Examples for ADDPJE

Example 1: Specifying Additional Prestart Jobs

```
ADDPJE SBSD(QGPL/PJSBS) PGM(QGPL/PGM1) INLJOBS(15)
      THRESHOLD(5) ADLJOBS(10) WAIT(*NO)
```

This command adds a prestart job entry for the PGM1 program in the QGPL library to the PJSBS subsystem description contained in the QGPL library. The entry specifies that 15 prestart jobs (program PGM1 in the QGPL library) are started when subsystem PJSBS in the QGPL library is started. When the pool of available prestart jobs is reduced to four (because the prestart jobs are servicing requests specified for program PGM1 in the QGPL library), ten additional jobs are started. If no prestart jobs are available for this entry when a request is received, the request is rejected.

Example 2: Specifying Maximum Number of Prestart Jobs

```
ADDPJE SBSD(QGPL/PJSBS) PGM(QGPL/PGM2)
      USER(PJUSER) MAXJOBS(100)
      MAXUSE(50) CLS(QGPL/CLS1 75 QGPL/CLS2 *CALC)
```

This command adds a prestart job entry for the PGM2 program in the QGPL library to the PJSBS subsystem description contained in the QGPL library. The entry specifies that the prestart job for this entry runs under the PJUSER user profile. The maximum number of prestart jobs that can be active at the same time for this entry is 100. Each prestart job in the pool can handle 50 requests before the job is ended. If 100 prestart jobs are active at the same time for this entry, 75 of them would use CLS1 in the QGPL library, and 25 of them would use CLS2 in the QGPL library. If 50 prestart jobs are active at the same time for this entry, all 50 of them would use class CLS1 in the QGPL library.

Error messages for ADDPJE

*ESCAPE Messages

CPF1691

Active subsystem description may or may not have changed.

ADDPRBACNE (Add Problem Action Entry) Command Description

ADDPRBACNE Command syntax diagram

Purpose

The Add Problem Action Entry (ADDPBACNE) command adds an entry to the specified problem filter. This entry describes the actions to take for a problem entry. A problem entry is assigned to the specified group by a selection entry in the specified problem filter.

Required Parameters

FILTER

Specifies the name of the filter.

The name of the filter can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

problem-filter-name: Specify the name of the filter.

GROUP

Specifies the group for which the actions are applied. The group name is assigned from selection criteria from a selection entry in the filter. Selection entries are added to the filter with the ADDPRBSLTE command.

Optional Parameters

ASNUSER

Specifies the user assigned to the problem log entry.

***NOCHG:** No new value is assigned to the problem log entry.

***NONE:** No user is assigned to the problem log entry.

assigned-user: Specify a user name.

SNDDTAQ

Specifies the data queue for the problem notification record. Keyed data queues are supported.

***NONE:** No data queue is used.

Element 1: Data Queue Name

The name of the data queue can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

data-queue-name: Specify the name of the data queue.

Element 2: Data Queue Key

***NONE:** No key is used on the data queue.

data-queue-key: Specify the data queue key.

Notes:

1. If an exact match is not found for the group name when a filter is applied, then the default action entry is used to assign actions. The default action entry is automatically added to the filter when it is created. The default values are ASNUSER(*NONE) and SNDDTAQ(*NONE).
2. A keyed data queue is a queue with a key assigned to each entry on the queue. When retrieving entries, a key can be specified and the entries with that key are retrieved on a FIFO order. The key that is specified on the *SNDDTAQ parameter is assigned to the problem notification record when placed on a keyed data queue.

An 80-byte record is enqueued on the data queue specified by the user. This record is received when the QRCVDTAQ program is called. The data queue does not have to be used solely for problems; alerts and problems can share the same data queue.

If a key is specified, it is used when enqueueing the record on the queue. If the data queue is non-keyed, the record is enqueued without a key.

Note: The time stamp used is the system standard time stamp. This time is already stored in the problem record.

The following table describes the record format.

Table 1. Record Format

Position	Type	Value	Description
1-10	CHAR	*PRBFTR	Problem filtering notification
11-11	CHAR	Function	Function performed
			1 - Problem created
			2 - Problem changed
			3 - Problem deleted
12-19	CHAR	Function TOD	TOD time stamp for function
20-29	CHAR	Group	Group problem was filtered into
30-39	CHAR	Problem ID	Problem ID number
40-59	CHAR	Origin System	System where problem originated
60-60	CHAR	Last Event	Last event committed into the history log (see note)
61-68	CHAR	Event TOD	TOD time stamp for Last Event
69-80	CHAR	Reserved	Reserved for future use

Position	Type	Value	Description
----------	------	-------	-------------

Note: Valid Last Event values are the following:

'01'X	Problem entry opened		
'02'X	Request received		
'03'X	Opened by Alert		
'10'X	Problem analyzed		
'11'X	Verification test ran		
'12'X	Recovery procedure ran		
'20'X	Prepared to report		
'21'X	Service request sent		
'22'X	Problem answered		
'23'X	Response sent		
'24'X	Reported by voice		
'25'X	Fixes transmitted		
'30'X	Fix verified		
'41'X	Analyzed remotely		
'42'X	Remote verification ran		
'43'X	Remote recovery ran		
'50'X	Alert created		
'51'X	APAR created		
'52'X	APAR data saved		
'54'X	APAR data restored		
'55'X	APAR data deleted		
'60'X	Problem changed by Change Problem (CHGPRB) command		
'61'X	Problem deleted by Delete Problem (DLTPRB) command		
'99'X	Problem entry closed		

Example for ADDPRBACNE

```
ADDPRBACNE FILTER(MYLIB/MYFILTER) GROUP(IOWA)
ASNUSER(SYSOPR) SNDDTAQ(*LIBL/PROBDTAQ)
```

The actions defined for group IOWA are: enqueue the problem on data queue PROBDTAQ; and assign the problem to user SYSOPR.

Error messages for ADDPRBACNE

*ESCAPE Messages

CPF2150

Object information function failed.

CPF2151

Operation failed for &2 in &1 type *&3.

CPF7A82

Error occurred while applying the problem filter.

CPF812F

Filter damaged.

CPF91DB

Group &4 already exists.

CPF91DE

Filter &1/&2 at maximum size.

CPF91EB

Filter type &3 not correct for this operation.

CPF91EC

Internal processing error occurred.

CPF91E8

Internal processing error occurred.

CPF9802

Not authorized to object &2 in &3.

CPF9803

Cannot allocate object &2 in library &3.

CPF9807

One or more libraries in library list deleted.

CPF9808

Cannot allocate one or more libraries on library list.

ADDPBSLTE (Add Problem Selection Entry) Command Description

ADDPBSLTE Command syntax diagram

Purpose

The Add Problem Selection Entry (ADDPBSLTE) command defines selection criteria that categorize a group of problem log entries. You can add a problem log selection entry to a problem log filter that was created using the Create Filter (CRTFTR) command.

Required Parameters

FILTER

Specifies the name of the filter.

The name of the filter can be qualified by one of the following library values:

***LIBL:** All libraries in the job's library list are searched until the first match is found.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

problem-log-filter-name: Specify the name of the filter.

SELECT

Specifies that a problem log entry is selected or not selected based on whether information in the problem log entry satisfies a specified relationship.

You can specify a single value (*ANY) or all four elements that define a relationship. When you specify the four elements, the attribute and attribute value are compared for the relationship specified by the relational operator.

***ANY:** Any problem log entry is selected.

Element 1: Logical Operator

***IF:** The specified relationship must be satisfied for a problem log entry to be selected.

***AND:** The specified relationship must be satisfied in addition to the *IF relationship for a problem log entry to be selected.

***OR:** The specified relationship must be satisfied in addition to or instead of the *IF relationship for a problem log entry to be selected.

Element 2: Attribute

***EVENT:** The filter is applied when the problem log entry is created (a value of 1), changed (a value of 2), or deleted (a value of 3). If the entry has been created and is changed before being committed, use the value of 1.

***ORIGIN:** The problem log entry was locally generated (a value of L) or was received from another system (a value of R).

***ORGNETID:** The network identifier (ID) of the system in which the problem log entry originated is specified. This information is displayed using the Work with Problems (WRKPRB) command which shows the details for a specific problem. Specify the value in the following form:

'nnnnnnnnnn'

***ORGCNAM:** The control point name of the system in which the problem log entry originated is specified. This information is displayed using the Work with Problems (WRKPRB) command which shows the details for a specific problem. Specify the value in the following form:

'cccccccccc'

***RCVNETID:** The network identifier of the remote system from which the problem log entry was received is specified. This information is displayed using the Work with Problems (WRKPRB) command which shows the details for a specific problem. Specify the value on the following form:

'nnnnnnnnnn'

***RCVCPNAM:** This attribute specifies the Remote System Control Point name in which the problem log entry received from. This information is displayed using the Work with Problems (WRKPRB) command and shows the details for a specific problem. The value specified for this attribute should be of the following form:

'cccccccccc'

***PROBTYPE:** The type of problem entry created. Possible problems are machine-detected (a value of 1), user-detected (a value of 2), PTF order (a value of 3), application-detected (a value of 4), PC machine-detected (a value of 5), or PC user-detected (a value of 6).

Note: User-Detected Remote Hardware problems are grouped with number 2 User-Detected problems.

***SEV:** The severity of the problem log entry created. Possible choices are high (a value of 1), medium (a value of 2), low (a value of 3), none (a value of 4), or not assigned (a value of 5).

Note: Problems do not have a severity level when locally created.

***MSGID:** The message ID found in the problem log entry. This is usually an iSeries 400 message ID from an iSeries 400.

***ORGHDW:** The origin hardware resource information in the problem log entry. This information is displayed using the Work with Problems (WRKPRB) command and shows the details for a specific problem. Specify the value in the following form:

```
'tttt mmm ss-sssssss'  
'tttt mmm ss-sssss'  
'tttt mmm sssssss'  
'tttt mmm sssss'
```

where tttt is the machine type, mmm is the model number and ssssssss is the serial number. Use this exact format to match a particular hardware resource exactly, or use a part of the hardware value with the Contains (*CT) relation to provide a partial match.

***RSCHDW:** The failing hardware resource information in the problem log entry. This information is displayed using the Work with Problems (WRKPRB) command and shows the details for a specific problem. Specify the value in the following form:

```
'tttt mmm ss-sssssss'  
'tttt mmm ss-sssss'  
'tttt mmm sssssss'  
'tttt mmm sssss'
```

where tttt is the machine type, mmm is the model number and ssssssss is the serial number. Use this exact format to match a particular hardware resource exactly, or use a part of the hardware value with the Contains (*CT) relation to provide a partial match.

***RSCSFW:** The failing software resource information in the problem log entry. This information is displayed using the Work with Problems (WRKPRB) command and shows the details for a specific problem. Specify the value in the following form:

```
'ppppppp vv rr mm'
```

where ppppppp is the licensed program ID, vv is the version number, rr is the release number, and mm is the modification level. Use this exact format to match a particular software resource exactly, or use a part of the software value with the Contains (*CT) relation to provide a partial match.

Element 3: Relational Operator

The value specified for Element 2 must have the following relationship to Element 4:

- *EQ** Equal to
- *GT** Greater than
- *LT** Less than
- *NE** Not equal to
- *GE** Greater than or equal to
- *LE** Less than or equal to
- *CT** Contains

Element 4: Attribute Value

attribute-value: Specify a value to compare with the contents of the attribute specified for Element 2. A maximum of 30 characters can be specified. The value must be specified in character format and must be enclosed in apostrophes if it contains blanks or special characters. If a CL variable is specified for the value, it must be a character variable.

generic-attribute-value*: Specify the generic attribute value.

Optional Parameters

SEQNBR

Specifies the sequence number of the problem log selection entry. Selection entries in a filter are numbered by sequence number. When a filter is applied, the selection entries are tested in order of ascending sequence number.

***GEN**: The system generates the sequence number.

sequence-number: Specify a number from 1 through 9999.

GROUP

Specifies the group to which a problem log entry is assigned if it matches the criteria specified on the SELECT parameter.

***DEFAULT**: The problem log entry is assigned to the default group.

group-name: Specify a group name.

Examples for ADDPRBSLTE

Example 1: Adding a Selection Entry

```
ADDPRBSLTE FILTER(PROBLIB/PROBFILTER)
SELECT((*IF *EVENT *EQ 1) (*AND *SEV *EQ 1))
SEQNBR(*GEN) GROUP(HIGHPROB)
```

This command adds an entry to the filter PROBFILTER in library PROBLIB. Any problems that have been created and are of severity 1 are assigned to group HIGHPROB.

Example 2: Assigning Entries by Origin System Network ID

```
ADDPRBSLTE FILTER(PROBLIB/PROBFILTER)
SELECT((*IF *ORGNETID *EQ 'IOWA'))
SEQNBR(*GEN) GROUP(IOWA)
```

This command assigns any problems with a origin system network ID of IOWA to group IOWA.

Example 3: Assigning Entries by Problems for Messages

```
ADDPRBSLTE FILTER(PROBLIB/PROBFILTER)
SELECT((*IF *MSGID *EQ 'CPF89*')) SEQNBR(*GEN)
GROUP(MSGCPF89)
```

This command assigns any problems for message CPF8901, CPF8902, and so on, to group MSGCPF89.

Example 4: Assigning Entries by Hardware Problems

```
ADDPRBSLTE FILTER(PROBLIB/PROBFILTER)
SELECT((*IF *RSCHDW *CT 9404) (*OR *RSCHDW *CT 9406)
(*OR *RSCHDW *CT 9402)) SEQNBR(*GEN) GROUP(AS400USER)
```

All problems for iSeries 400 hardware (the hardware resource information *containing* machine type 9402, 9404 or 9406) are assigned to group AS400USER.

Caution must be taken when using the contains operation. In this example if the sending machine had a serial number containing 9402, 9404, or 9406 it would also match this selection entry even if the machine type was not 9402, 9404, or 9406. A better example follows.

Example 5: Assigning Entries by Hardware Problems

```

ADDPBRSLTE FILTER(PROBLIB/PROBFILTER)
  SELECT((*IF *RSCHDW *EQ 9404*) (*OR *RSCHDW *EQ 9406*)
(*OR *RSCHDW *EQ 9402*)) SEQNBR(*GEN) GROUP(AS400USER)

```

This command assigns all problems for iSeries 400 hardware (the hardware resource information *equals* machine type 9402, 9404 or 9406) to group AS400USER.

This is a better way to select on the sending hardware machine type. Only those machines with types of 9402, 9404, or 9406 will result in a match.

Example 6: Assigning Entries by Machine-detected Problems

```

ADDPBRSLTE FILTER(PROBLIB/PROBFILTER)
  SELECT((*IF *PROBTYPE *EQ 1)) SEQNBR(*GEN)
  GROUP(MACHDETECT)

```

This command assigns any problems that are machine-detected to group MACHDETECT.

Example 7: Assigning Entries by Product-specific Problems

```

ADDPBRSLTE FILTER(PROBLIB/PROBFILTER)
  SELECT((*IF *RSCSFW *EQ '5716SS1 03 06 00'))
  SEQNBR(15) GROUP(OS400V3R6)

```

This command assigns any problems that are specifically for OS/400 Version 3 Release 6 Modification 0 to group OS400V3R6. Notice that this entry is placed after entry number 10 in the filter, since 15 is specified as the sequence number.

Example 8: Assigning Entries by Matching Products

```

ADDPBRSLTE FILTER(PROBLIB/PROBFILTER)
  SELECT((*IF *RSCSFW *EQ '5716SS1*'))
  SEQNBR(25) GROUP(OS400)

```

This selection entry matches Version 3 Release 6 of the OS/400 licensed program.

Notes:

1. The order of selection entries within a filter is important. When the filter is applied to the problem log entry, the selection entries are examined from the *first* entry to the *last* entry in ascending order. The *first* selection entry that matches a problem is used. To ensure correct operation the most specific selection entries should be first, and the least specific selection entries last.
2. If the selection entries are not order specific (i.e. each selection entry matches one and only one problem) then the most likely or the most common should be placed first. This will ensure the best performance as fewer selection entries will need to be checked.
3. If no selection entries result in a match when a filter is applied, then the *LAST selection entry is used to assign a group. The *LAST selection entry is automatically added to the filter when it is created. The SELECT parameter for the *LAST selection entry is *ANY, which will always result in a match.
4. The *AND logical operator takes precedence over the *OR logical operator within a selection entry. Therefore, the following SELECT specification:

```

((*IF *PROBTYPE *EQ 1) (*AND *SEV *EQ 1)
(*OR *PROBTYPE *EQ 2) (*AND *SEV *EQ 1))

```

is equivalent to the following Boolean expression:

```

if ((*PROBTYPE = 1) and (*SEV = 1)) or
    ((*PROBTYPE = 2) and (*SEV = 1))

```

5. All attribute values are interpreted as character data, including numbers. When the problem filter is applied to a problem, the system converts all of the data in the filter to the type given in the problem template and compared. Message IDs are considered character data and are ordered as such.

Error messages for ADDPRBSLTE

***ESCAPE Messages**

CPF2150

Object information function failed.

CPF2151

Operation failed for &2 in &1 type *&3.

CPF7A82

Error occurred while applying the problem filter.

CPF812F

Filter damaged.

CPF91DA

Sequence number &4 already exists.

CPF91D9

Sequence number cannot be automatically created.

CPF91EA

*IF relationship not in correct position.

CPF91EB

Filter type &3 not correct for this operation.

CPF91EC

Internal processing error occurred.

CPF91E6

Generic values only allowed with *EQ or *NE.

CPF91E7

Character in position &4 not valid in value specified.

CPF91E8

Internal processing error occurred.

CPF9802

Not authorized to object &2 in &3.

CPF9803

Cannot allocate object &2 in library &3.

CPF9807

One or more libraries in library list deleted.

CPF9808

Cannot allocate one or more libraries on library list.



Printed in U.S.A.