



@server

iSeries

CL Commands Volume 2





@server

iSeries

CL Commands Volume 2

Contents

Command Descriptions	1
ADDPRDCRQA (Add Product Change Request Activity) Command Description	1
ADDPRDLICI (Add Product License Information) Command Description	8
ADDPGM (Add Program) Command Description	11
ADDPTFCRQA (Add Program Temporary Fix Change Request Activity) Command Description	12
ADDPCLTBLE (Add Protocol Table Entry) Command Description	20
ADDRDBDIRE (Add Relational Database Directory Entry) Command Description	21
ADDRMTDFN (Add Remote Definition) Command Description	27
ADDRMTJRN (Add Remote Journal) Command Description	30
ADDRPYLE (Add Reply List Entry) Command Description	34
ADDRSCCRQA (Add Resource CRQ Activity) Command Description	39
ADDREXBUF (Add REXX Buffer) Command Description	45
ADDRTGE (Add Routing Entry) Command Description	46
ADDSCHIDX (Add Search Index Entry) Command Description	49
ADDSVRAUTE (Add Server Authentication Entry) Command Description	51
ADDSRVTBLE (Add Service Table Entry) Command Description	52
ADDSOCE (Add Sphere of Control Entry) Command Description	53
ADDTAPCTG (Add Tape Cartridge) Command Description	54
ADDTCPHTE (Add TCP/IP Host Table Entry) Command Description	56
ADDTCPIFC (Add TCP/IP Interface) Command Description	60
ADDTCPPORT (Add TCP/IP Port Restriction) Command Description	67
ADDTCPRSI (Add TCP/IP Remote System Information) Command Description	69
ADDTCPRTE (Add TCP/IP Route) Command Description	73
ADDTRC (Add Trace) Command Description	78
ADDTRCFTR (Add Trace Filter) Command Description	82
ADDUSFCNNE (Add Ultimedia System Facilities Connection Entry) Command Description	85
ADDUSFDEVE (Add Ultimedia System Facilities Device Entry) Command Description	86
ADDUSFSVRE (Add Ultimedia System Facilities Server Entry) Command Description	89
ADDWSE (Add Work Station Entry) Command Description	90
ALCOBJ (Allocate Object) Command Description	93
ANZACGRP (Analyze Process Access Group) Command Description	99
ANZDBF (Analyze Database Files) Command Description	100
ANZDBFKEY (Analyze Database File Keys) Command Description	101
ANZDFTPWD (Analyze Default Passwords) Command Description	103
ANZJVAPGM (Analyze Java Program) Command Description	103
ANZJVM (Analyze Java Virtual Machine) Command Description	105
ANZLIBBRM (Analyze Libraries Using BRM) Command Description	107
ANZPFRDTA (Analyze Performance Data) Command Description	107
ANZPRB (Analyze Problem) Command Description	110
ANZPRFACT (Analyze Profile Activity) Command Description	112
ANZPGM (Analyze Programs) Command Description	113
ANZQRY (Analyze Query) Command Description	114
ANZUSROBJ (Analyze User Objects) Command Description	115
ANSLIN (Answer Line) Command Description	117
ANSQST (Answer Questions) Command Description	118
APYJRNCHG (Apply Journalized Changes) Command Description	118
APYPTF (Apply Program Temporary Fix) Command Description	130
APYRMTPTF (Apply Remote Program Temporary Fix) Command Description	135
ASKQST (Ask Question) Command Description	139
BCHJOB (Batch Job) Command Description	140
CALLPRC (Call Bound Procedure) Command Description	149
CALL (Call Program) Command Description	151
CHGACGCDE (Change Accounting Code) Command Description	154

CHGACTSCDE (Change Activation Schedule Entry) Command Description	156
CHGACTPRFL (Change Active Profile List) Command Description	157
CHGALRACNE (Change Alert Action Entry) Command Description	158
CHGALRD (Change Alert Description) Command Description	161
CHGALRSLTE (Change Alert Selection Entry) Command Description	165
CHGALRTBL (Change Alert Table) Command Description	170
CHGASPA (Change ASP Attribute) Command Description	171
CHGATR (Change Attribute) Command Description	173
CHGAUD (Change Auditing Value) Command Description	178
CHGAUT (Change Authority) Command Description	181
CHGAUTLE (Change Authorization List Entry) Command Description	186
CHGAJE (Change Autostart Job Entry) Command Description	188
CHGBCKUP (Change Backup Options) Command Description	190
CHGBPA (Change BOOTP Attributes) Command Description	194
CHGCCSA (Change Change Control Server Attributes) Command Description	194
CHGCRQA (Change Change Request Activity) Command Description	201
CHGCRQD (Change Change Request Description) Command Description	224
CHGCLS (Change Class) Command Description	226
CHGCOSD (Change Class-of-Service Description) Command Description	229
CHGCLNUP (Change Cleanup) Command Description	233
CHGCLUCFG (Change Cluster Configuration Tuning) Command Description	239
CHGCLUNODE (Change Cluster Node Entry) Command Description	240
CHGCLURCY (Change Cluster Recovery) Command Description	242

Command Descriptions



ADDPDCRQA (Add Product Change Request Activity) Command Description

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

ADDPDCRQA Command syntax diagram

Purpose

The Add Product Change Request Activity (ADDPDCRQA) command adds an activity to a change request description that performs a product distribution function.

The activity can be conditioned so that it 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. This command is shipped with public *EXCLUDE authority.
2. You must have *CHANGE authority to the change request description object and *EXECUTE authority to the library.
3. The product must have been previously packaged for distribution. You can use the Package Product for Distribution (PKGPRDDST) command to package the product.
4. If a NODL value is specified, the node list can only contain entries that have a value of *SNA for the address type.

The following notes provide information on how the command works.

Notes:

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 be performed.
3. The start times indicate when the activity can be started. Actual start times can be later due to network and system delays.
4. Action *DLTCLGE only deletes the save file containing the licensed program and deletes the distribution catalog entry that maps this save file. It does not delete the product.
5. If TGTRLS(*ONLY) is specified, the action parameter must be a value other than *INS.
6. When it is requested to send the license key of the product, a record with the license key information must exist in the central site license repository by the time this activity runs.
7. All the existing records in the central site license repository containing the license key information for the specified product will be sent to the specified managed system or systems, but only those that match the system serial number are added to the managed system license repository.

Required Parameters

CRQD Specifies the 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: 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.

ACTION

Specifies the product distribution function that is to be performed.

***SND:** Send the product to the specified managed systems.

***RTV:** Retrieve the product from the specified managed system.

***DLTCLGE:** Delete the catalog entry and its associated save file which contains the product packaged for distribution from the specified managed systems.

***INS:** Install the product on the specified managed systems.

***SNDINS:** Send and install the specified product on the specified managed systems.

PRDID

Specifies the 7-character identifier of the product for which the action is performed.

product-ID: Specify the 7-character product ID that is used in the activity.

Optional Parameters

RLS Specifies which version, release, and modification level of the product is used.

***ONLY:** The release level of the product that is installed on your system.

version-release-modification: Specify the release level in the format VxRxMy, where Vx is the version number, Rx is the release number, and My is the modification number. Valid values for x are the numbers 0 through 9. Valid values for y range from the numbers 0 through 9 and the letters A through Z.

OPTION

Specifies which of the optional parts of the product given in the PRDID parameter are used.

***BASE:** Only the base part of the product is used.

product-option-number: Specify the option number for the product load being used. Valid values range from 1 through 99.

LODTYPE

Specifies the product load objects being used.

***ALL:** Code and language objects specified on the LODID parameter are used.

***CODE:** The program objects associated with this product load are used.

***LNG:** The objects associated with the national language version (NLV) identified on the LODID parameter are used.

LODID

Specifies the load identifier used.

***ALL:** All languages for this product option are used.

***CODE:** The code load is used.

product-load-ID: Specify the load ID of the product when LODTYPE(*LNG) or LODTYPE(*ALL) is specified. The load ID must be one of the valid IBM national language versions and be specified in the form 29xx. The value of x can be 0 through 9.

TGTRLS

Specifies the release of the operating system on which you intend to use the product.

***ONLY:** The release is determined by the release of the existing product. This value is not valid if more than one release exists for the same product.

***CURRENT:** The product is to be used on the release of the operating system currently running on your system.

***PRV:** The product is to be used on the previous release with modification level 0 of the operating system.

release-level: Specify the release level in the format VxRxMx. The product can be 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 can change with each new release.

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 control point (CPNAME) parameter or ACTION(*RTV) 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 command 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. If ACTION(*RTV) is used, only one control point name can be 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.

KEEPCLGE

Specifies if the distribution catalog entry and its associated save file corresponding to the product are kept on the specified systems. This is only valid if ACTION(*INS) or ACTION(*SNDINS) is specified.

***NO:** The catalog entry and associated save file are not kept.

***YES:** The catalog entry and associated save file are kept.

SNDLICKEY

Specifies if the license key is to be sent with the product.

***YES:** The license key is sent with the product.

***NO:** The license key is not sent with the product.

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 which 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 was 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 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 not met

Element 4: Condition Mode

This element indicates which systems the conditioning activity must have completed on or 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 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. 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 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 after which 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

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

start-after-time: Specify the time after which this function can be started on the managed system. The time can be entered as 4 or 6 digits (hhmm) where hh = hours, mm = minutes. Seconds are optional. The time can be specified with or without a time separator. Seconds are optional. 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 can start on the managed system on any date on or after the date this activity starts on the central site system.

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

start-after-date: Specify the date after which the function can be started 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 time when the time zone value *MGDSYS is specified.
2. This parameter can only be specified if *INS or *SNDINS actions are specified.

TEXT Specifies an activity description.

***GEN:** A text description is generated based on the action chosen.

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 ADDPRDCRQA

Example 1: Adding an Activity to Retrieve

```
ADDPRDCRQA CRQD(MYLIB/CR1) ACTIVITY(ACT01) ACTION(*RTV)
PRDID(1ACCOUN) RLS(V5R2M0) OPTION(*BASE)
LODTYPE(*ALL) LODID(*ALL) CPNAME((*NETATR SYS1))
```

Add an activity to retrieve the base option for the 1ACCOUN product, release V5R2M0, for both the code and the language parts from the iSeries server SYS1.

Example 2: Adding an Activity to Send and Install

```
ADDPRDCRQA CRQD(MYLIB/CR2) ACTIVITY(ACT01)
ACTION(*SNDINS) PRDID(1CHECKS) RLS(V5R2M0)
OPTION(0010) LODTYPE(*ALL) LODID(*ALL)
CPNAME((*NETATR SYS2)) SNDLICKEY(*YES)
```

This command adds an activity to send and install option 10 of the 1CHECKS product, release V5R2M0, for both the code and the language parts from the iSeries server SYS2. The license key for this product is also sent and is added to the managed system license repository.

Example 3: Adding an Activity to Install Language Objects

```
ADDPRDCRQA CRQD(MYLIB/CR3) ACTIVITY(ACT02) ACTION(*SND)
PRDID(1ACCOUN) RLS(V5R2M0) OPTION(*BASE)
LODTYPE(*ALL) LODID(2924) NODL(NETLIB/ACCTSYS)
STRTIME((23:00:00 9/30/02))
```

This command adds an activity to send the language objects for the English version and the code objects of the base option of the 1ACCOUN product, to all the systems in the node list (NODL) ACCTSYS network at 11 p.m. on 30 September 2002. The license key for this product is also sent and is added to the managed system license repository.

Error messages for ADDPRDCRQA

*ESCAPE Messages

None <<

ADDPRDLICI (Add Product License Information) Command Description

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

ADDPRDLICI Command syntax diagram

Purpose

The Add Product License Information (ADDPRDLICI) command adds license information to the product or feature. This command can be used before or after the product or options belonging to a feature are packaged using the Package Product Option (PKGPRDOPT) command, but before the product or options belonging to a feature are installed. If the product has keyed (*KEYED) compliance, a product license information handle is included in the completion message received from this command.

The Add Product License Information (QLZADDLI) API is also available in the Application Program Interfaces (APIs) topic in the Information Center.

Restriction: This command is shipped with public *EXCLUDE authority.

Required Parameters

PRDID

Specifies the product identifier of the product loads to which the license information is being added.

Note: The identifier must be exactly 7 characters long.

RLS Specifies the version, the release, and the modification level of the product or feature to which the license information is being added.

version-release-modification: Specify the version, release, and modification level of the product or feature where the license information is being added. The release level must be in the format VxRyMz, where valid values for x and y range from 0 through 9 and valid values for z range from 0 through 9 and A through Z.

Optional Parameters

FEATURE

Specifies the feature for which the license information is being added. The license information is added to all product loads with the specified product ID and feature.

5001: License information for the 5001 feature of the product specified on the PRDID parameter is added.

load-ID: Specify that the license information for the feature on the PRDID parameter associated with this number is to be added. Valid values range from 5001 through 9999.

USGTYPE

Specifies the type of license usage.

***CONCURRENT:** The usage limit is for the number of unique jobs accessing the product or feature at one time.

***REGISTERED:** The usage limit is for the number of unique license users registered by the product or feature.

COMPLIANCE

Specifies the action that must be taken when the usage limit is exceeded.

***OPRACION:** The usage limit cannot be exceeded. If you attempt to use the product or feature after the usage limit has been reached, you are prevented from accessing the product or feature until the appropriate action is taken by the system administrator to increase the usage limit.

***WARNING:** If you attempt to use the product or feature after the usage limit has been reached, you are allowed access. A warning message indicating that the usage limit has been exceeded is sent to the licensing information owner for notification.

***KEYED:** A license key is required from the software provider to install or to increase the usage limit of the product or feature. The software provided can permit a grace period that allows the usage limit of a product or feature to be exceeded for n days before a new license key is required to allow additional users access to the product. If a new key with an increased usage limit is not installed prior to the end of the grace period, the usage limit of the product reverts back to the entitled usage limit.

DFTUSGLMT

Specifies the usage limit that is in effect the first time the product or feature is installed for a license term.

Note:

For *KEYED compliance, this is the number of users that are able to use the product or feature before the license key is installed using the Add License Key Information (ADDLICENSE) command or Add License Key Information (QLZAADDK) API. To prevent any access to the product or feature without a key, the default usage limit should be set to 0.

1: One user is allowed to access the product or feature.

***NOMAX:** Any number of users are allowed to access the product or feature.

default-usage-limit: Specify the default usage limit number. The valid values range from 0 through 999 999.

LICTRM

Specifies which levels of the product or feature share license information.

***VERSION:** All VxRx levels of the product, where Vx is the version level and Rx is the release level. When the version level of the product or feature changes, customers installing the new release must use the Change License Information (CHGLICINF) command for a compliance type of *WARNING or *OPRACION, or the Add License Key Information (ADDLICENSE) command for a compliance type of *KEYED to set the license information.

***RELEASE:** All VxRx levels of the product, where Vx is the version level and Rx is the release level, specified on the RLS parameter whose license term is also *RELEASE share the same license information. When the version level or the release level of the product or feature changes, customers installing the new release must use the Change License Information (CHGLICINF) command for a compliance type of *WARNING or *OPRACION or the Add License Key Information (ADDLICENSE) command for a compliance type of *KEYED to set the license information.

***MOD:** Each level of the product or feature is licensed. No license information is shared between levels. Customers installing a new level of the product or feature must use the Change License Information (CHGLICINF) command for compliance type of *WARNING or *OPRACION, or the Add License Key Information (ADDLICENSE) command for compliance type of *KEYED to set the license information.

MSGID

Specifies the message identifier of the message that describes the product feature. The message file for this message ID is specified in the product definition.

***BASEOPT:** There is no message describing this product feature. The message used for describing the base option of this product is used instead.

feature-message-id: Specify the message identifier of the message that describes this product feature.

ALWLICRLS

Specifies whether a license that is previously requested can be released using the WRKLICINF command. When the license is released, the usage count is decreased. This value can only be specified for a registered license type.

If it is necessary for the product to be notified that a license has been released, *NO should be specified and the product must provide its own mechanism for releasing a license.

***NO:** A license cannot be released using the WRKLICINF command.

***YES:** A license can be released using the WRKLICINF command.

VNDPWD

Specifies the software vendor's password. This password is encrypted and stored with the product. It is used in validating Add License Key requests and must be the same password used to generate keys for this product and feature on the Generate License Key (GENLICKEY) command.

***NONE:** There is no vendor password needed for the Add License Key Information (ADDLICKEY) command.

vendor-password: Specify the vendor password. It must begin with an alphabetic character such as A through Z, \$, # or @. It is then followed by no more than 9 alphanumeric characters such as A through Z, 0 through 9, \$, # @ or _.

GRACE

Specifies the number of days after a product first exceeds its usage limit that you have to obtain a new license key. If a new license key is not obtained from the software vendor by the time the grace period is expired, no users over the usage limit are allowed to access the product or feature. When the usage limit is first exceeded, the date the grace period expires is calculated by adding the number of days in the grace period to today's date.

Element 1: Number of Days

0: There is no grace period. The user is not allowed to exceed the usage limit.

grace-period: Specify the number of days in the grace period. Valid values range from 0 through 999.

Element 2: Allow for Default Usage Limit

***NO:** The grace period is not allowed for use with the default usage limit. This means that you are not allowed to exceed the default usage limit before the license key is installed using the Add License Key Information (ADDLICKEY) command or Add License Key Information (QLZAADDK) API.

***YES:** The grace period is allowed for use with the default usage limit. This means that you can exceed the default usage limit for the number of days in the grace period before the license key is installed.

Examples for ADDPRDLICI

Example 1: Adding Product License Information

```
ADDPRDLICI PRDID(1MYPROD) RLS(V5R2M0)
```


This command adds product license information to the product with product identifier 1MYPROD for release V5R2M0.

Example 2: Adding Product License Information for the V5R2M0 Release

```
ADDPRDLICI PRDID(2MYPROD) RLS(V5R2M0) COMPLIANCE(*KEYED)
          VNDOWD(PRODUCTPWD) GRACE(30 *YES)
```

This command adds product license information to the product with the product identifier 1MYPROD for release V5R2M0. This product supports keyed compliance. It can exceed its usage limit and default usage for 30 days.

Error messages for ADDPRDLICI

*ESCAPE Messages

CPF9E1A

License information conflict found.

ADDPGM (Add Program) Command Description

ADDPGM Command syntax diagram

Purpose

The Add Program (ADDPGM) command adds 1 to 20 programs to the group of programs currently being debugged. When included in debug mode, the specified programs can have breakpoints and traces added to them for controlling and tracing their processing. The values of the programs' variables can also be shown and changed.

When debugging one job from another job, debugging affects the running of the programs in the job being debugged, but not in the job doing the debugging. The user may run programs in a job doing the debugging, however, the programs will not be debugged.

Restrictions:

1. No more than 20 programs can be debugged at the same time.
2. Two or more programs with the same name cannot be debugged at the same time.
3. This command is valid only in debug mode. To start debug mode, see the Start Debug (STRDBG) command.
4. 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.
5. This command cannot be used to add bound programs.
6. The user must have either *CHANGE authority to the program, or *USE authority to the program and *SERVICE special authority.

Required Parameter

PGM Specifies the qualified name of one or more programs being debugged. The number of programs specified here depends on how many programs are already being debugged; 20 is the maximum at any time. The user cannot debug two programs that have the same name.

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 program to be added for debugging.

Optional Parameter

DFTPGM

Specifies the name of the program to use as the default program during debug mode. The program specified here is used as the default program for any of the other debug commands that specify *DFTPGM on their PGM parameter. That is, if a default program was previously specified, this parameter can change it.

***SAME:** The value does not change.

***NONE:** No program is specified as the default program; if a program was specified as a default program, it is no longer the default program. If the job has no default program, *DFTPGM cannot be specified on the PGM parameter of any other debug commands.

program-name: Specify the name of the program to use as the default program during debug mode. The same name must also be specified in the PGM parameter of this command or have been specified on the Start Debug (STRDBG) command or on a previous Add Program (ADDPGM) command.

Example for ADDPGM

```
ADDPGM PGM(QGPL/MYPROG)
```

This command adds the program MYPROG, located in the QGPL library, to the current debug mode. Breakpoints and traces can be put in MYPROG, and its variables can be displayed and changed by other debug commands. Because DFTPGM was not specified, the default program is not changed.

Error messages for ADDPGM

*ESCAPE Messages

CPF1999

Errors occurred on command.



ADDPTFCRQA (Add Program Temporary Fix Change Request Activity) Command Description

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

ADDPTFCRQA Command syntax diagram

Purpose

The Add Program Temporary Fix Change Request Activity (ADDPTFCRQA) command adds an activity to a change request description that performs a PTF distribution function.

Restrictions:

1. You must have *CHANGE authority to the change request description and *EXECUTE authority to the library.
2. The PTF must be for an OS/400 product package using the System Manager licensed program. The PTF must be supported using the Work with Supported Product (WRKSPTPRD) command. Software redesigns can be managed for other types of systems by using globally named objects on the Add Object Change Request Activity (ADDOBJCRQA) command and the Add Change Request Activity (ADDCRQA) command, or by using the Add Change Management Activity (QNSADDCM) API.
3. A PTF save file must exist and be released in order to be sent to another system.
4. A PTF can only be retrieved from a single managed system.
5. If a NODL value is specified, the node list can only contain entries that have a value of *SNA for the address type.
6. If the destination node does not have the Managed System Services licensed program installed, only the send action is available. The activity is considered successful when the PTF is sent, not when it arrives.
7. If you are distributing a PTF and the distribution queue is set to *SNADS in the Work with Service Requesters (WRKSRVRQS) command, you must have *USE authority to the Send PTF (SNDPTF) command.
8. If you are distributing a PTF and the distribution queue is set to *SVDS in the Work with Service Requesters (WRKSRVRQS) command, you must have *USE authority to the Copy PTF (CPYPTF) command.

The following notes provide information on how the command works.

Notes:

1. Authorization to the product specified on the activity is not verified until the activity runs.
2. All conditions must be met before the activity runs.
3. The start times indicate when the activity can be started. Actual start times can be later due to network and system delays.
4. PTFs that are not marked as delayed are applied immediately. PTFs marked as delayed are scheduled to be applied at the next IPL.
5. PTFs marked as delayed can only be applied permanently if they have been previously applied temporarily. Because they are delayed, these PTFs would be applied during the next scheduled IPL.
6. PTFs are always removed temporarily except Vertical Licensed Internal Code (VLIC) PTFs which are removed permanently.

Required Parameters

CRQD Specifies the 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: 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 identifier.

ACTION

Specifies the function to be performed on the PTF.

***SND:** Send the PTF to the specified managed system. The PTF is not loaded as part of this activity.

***RTV:** Retrieve the specified PTF from the specified managed system.

***DLT:** Deletes the PTF save file and cover letter on the specified managed system.

***APY:** Apply the PTF on the specified system. All requisite PTFs must have been applied. If the PTF is not loaded, then the PTF is loaded and applied. A PTF can be in loaded ("Not applied") status. If the PTF is not previously loaded, then the PTF must be loaded before it is applied. To apply all corequisite PTFs, set the status to "loaded" ("Not applied"). When you request to apply one PTF, all corresponding corequisites will be applied.

***SNDAPY:** Send, load, and apply the PTF on the specified system. All requisite PTFs must have already been applied. To apply the current PTF and corresponding corequisite PTFs, set the status to "loaded" ("Not applied"). When applying one PTF, that PTF and all corresponding corequisite PTFs are applied.

***RMV:** Temporarily remove the PTF on the specified system.

PTFID Specifies the PTF that is to be distributed.

Element 1: PTF Identifier

PTF-identifier: Specify a 7-character PTF identifier.

***ALL:** All PTFs for the specified product. It can only be specified when applying or removing PTFs. A product must be specified.

Element 2: Product Identifier

This is the product identifier of the product to which the PTF is associated.

***ONLY:** The PTF identifier specified is associated with only one supported product.

product-ID: Specify the product to which the PTF is associated. The product must be specified when PTF identifiers are not unique across products.

Element 3: Release Level of Product

This element shows the release level of the product.

release (VxRxMx): Specify the release level of the product to which the PTF is associated. The format is VxRxMx. The release must be specified when PTF identifiers are not unique across product releases.

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 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 where the PTF function 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 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.

PTFPART

Specifies whether the PTFs or cover letters should be sent, retrieved, or deleted.

***PTF:** Only the PTF should be handled.

***CVRLTR:** Only the PTF cover letter is handled. *CVRLTR can only be used with the *RTV and *SND actions.

If the delete action is specified, then both the PTF and the cover letter are always deleted.

CVRLTRLNG

Specifies the language of the cover letter to be sent with the PTF.

***SRVRQS:** The cover letter language specified in the service requester entry for each node determines which cover letter that particular node is sent. If a service requester entry is not present or a language is not specified, the default language 2924 is used. If the language specified is not found, then 2924 is sent except when 2926 is requested in which case 2950 is sent.

cover-letter-language-ID: Specify the 4-character language identifier of the cover letter to be sent or retrieved.

APY Specifies the extent of change when the PTF is applied.

***TEMP:** The PTF is applied temporarily.

***PERM:** The PTF is applied permanently.

***LODONLY:** The PTF is only loaded. This is useful for PTFs which are part of a corequisite group where all PTFs are applied, removed, or permanently applied as a group and the “loaded” (“Not applied”) status is the requisite to perform those functions.

DLYAPY

Specifies how PTFs are applied. Apply immediate PTFs one at a time while the activity runs on the specified system or later during the next IPL.

***NO:** Applies an immediate PTF at the time the activity runs. If the PTF is marked delayed, it is not applied until the next unattended IPL.

***YES:** Applies both immediate or delayed PTFs during the next unattended IPL.

DLYRMV

Specifies how a PTF is removed. Remove a PTF at the time the activity is specified on the system or later during the next IPL.

***NO:** Removes a PTF at the time an activity runs.

***YES:** Removes a PTF during the next unattended IPL.

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 an 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 is 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 codes used by the change request manager and recommended for applications are the following:

- 00** Activity completed successfully.
- 01-09** Activity completed with warning messages.
- 10-29** Activity did not complete successfully.
- 30-39** Activity was canceled by the 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 authorization reasons.
 - 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 not 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 activity specified in this condition element must have completed 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 can 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 when 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 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. This value cannot be specified if the start before date is *CURRENT.

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 was started on the central site 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 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 *APY, *RMV or *SNDAPY actions are specified.

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.

TEXT Specifies the activity description.

***GEN:** A description is generated based on the action selected.

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

Examples for ADDPTFCRQA

Example 1: Adding an Activity to Permanently Apply a PTF

```
ADDPTFCRQA CRQD(MYLIB/CR1) ACTIVITY(ACT01) ACTION(*APY)
  APY(*PERM) PTFID(SF12345) CPNAME((*NETATR SYS1))
```

This command adds an activity to permanently apply PTF SF12345 on system SYS1.

Example 2: Add an Activity to Send a PTF Cover Letter

```
ADDPTFCRQA CRQD(MYLIB/CR2) ACTIVITY(ACT03) ACTION(*SND)
  PTFID(SF89345) PTFPART(*CVRLTR) CVLTRLNG(2950)
  NODL(NETLIB/EUROPE SYS)
```

This command adds an activity to send a PTF cover letter to all systems in Europe that are identified in the EUROPE SYS node list.

Example 3: Adding an Activity to Temporarily Apply a PTF

```
ADDPTFCRQA CRQD(MYLIB/CR3) ACTIVITY(ACT01) ACTION(*SND)
  PTFID(SF99911) CPNAME((NET1 SYSX))
```

```
ADDPTFCRQA CRQD(MYLIB/CR3) ACTIVITY(ACT02) ACTION(*APY)
APY(*TEMP) PTFID(SF99911)
RMTSTRTIME(*MGDSYS ('02:00:00' '12/12/02'))
CPNAME((NET1 SYSX))
```

This command adds activities to send a PTF and apply it temporarily at 2:00 a.m. on December 12th, 2002, on a system encountering a problem.

Example 4: Adding an Activity to Load a PTF

```
ADDPTFCRQA CRQD(MYLIB/CR4) ACTIVITY(ACT01) ACTION(*APY)
PTFID(SF89555) CPNAME(*NETATR SYS4) APY(*LODONLY)
```

This command adds an activity to only load the PTF SF89555, which is part of a corequisite PTF group, on system SYS4.

Example 5: Adding an Activity to Send and Permanently apply a PTF in Delayed Mode

```
ADDPTFCRQA CRQD(MYLIB/CR5) ACTIVITY(ACT01) ACTION(*SNDAPY)
PTFID(SF91388) CPNAME(*NETATR SYS5) APY(*PERM) DLYAPY(*YES)
```

This command adds an activity to send and permanently apply PTF SF91388 during the next unattended IPL.

Example 6: Adding an Activity to Load and Apply an Individual PTF and the Corequisite Group

```
ADDPTFCRQA CRQD(MYLIB/CR6) ACTIVITY(ACT01) ACTION(*SNDAPY)
PTFID(SF97001) CPNAME(*NETATR SYS6) APY(*LODONLY)
```

```
ADDPTFCRQA CRQD(MYLIB/CR6) ACTIVITY(ACT02) ACTION(*SNDAPY)
PTFID(SF97002) CPNAME(*NETATR SYS6) APY(*LODONLY)
```

```
ADDPTFCRQA CRQD(MYLIB/CR6) ACTIVITY(ACT03) ACTION(*SNDAPY)
PTFID(SF97003) CPNAME(*NETATR SYS6) APY(*TEMP)
```

In this example, there are three PTFs: SF97001, SF97002, and SF97003. SF97001 has corequisite SF97002, and SF97002 has corequisite SF97003. To apply one of them, you must apply all of them as a group to have the product function correctly. Therefore, you need to load the PTFs first, then when you request to apply only one of them, that PTF and its associated corequisites will all be applied.

Error messages for ADDPTFCRQA

*ESCAPE Messages

None <<

ADDPCLTBLE (Add Protocol Table Entry) Command Description

ADDPCLTBLE Command syntax diagram

Purpose

The Add Protocol Table Entry (ADDPCLTBLE) command is used to add a protocol entry to the protocol table. You can use the protocol table to manage a list of protocols used in the Internet. The **Internet** is a collection of networks functioning as a single, cooperative, and virtual network using Transmission Control Protocol/Internet Protocol (TCP/IP) to support peer-to-peer connectivity.

The protocol table is shipped with a list of some valid protocols. Current protocol values are available to the Internet community in the assigned numbers **RFC** (Request for Comments) document, a formal specification of proposals and standards for a portion of TCP/IP.

Restriction: You must have system configuration (*IOSYSCFG) special authority to use this command.

Required Parameters

PROTOCOL

Specifies the name of the protocol to be added to the table. A protocol can be added to the table only once.

PCLNBR

Specifies the number that represents the protocol.

Optional Parameters

ALIAS Specifies the alternate name for the protocol. You can specify a maximum of 4 aliases. No checking is done to ensure that an alias is unique.

***NONE:** The protocol has no alternate name.

alias: Specify an alternate protocol name.

TEXT Specifies the text that briefly describes the protocol entry. More information is in Commonly used parameters.

***BLANK:** Text is not specified.

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

Example for ADDPCLTBLE

```
ADDPCLTBLE  PROTOCOL(TCP)  PCLNBR(6)
```

This command adds an entry for the TCP protocol to the protocol table. The protocol number for the TCP entry is 6.

Error messages for ADDPCLTBLE

*ESCAPE Messages

TCP290B

Protocol entry already exists in table. Entry was not added.

TCP2915

Protocol entry contains characters that are not valid. Entry was not added.

ADDRDBDIRE (Add Relational Database Directory Entry) Command Description

ADDRDBDIRE Command syntax diagram

Purpose

The Add Relational Database Directory Entry (ADDRDBDIRE) command adds a relational database entry to the relational database directory. >>

Each entry describes the method of accessing a relational database. One special entry must exist which names the local relational database located on the system auxiliary storage pool (ASP number 1) and configured basic user ASPs (ASP numbers 2-32). This entry has *LOCAL in the RMTLOCNAME parameter first element.

Entries for auxiliary storage pool groups (with ASP numbers > 32) will be added by the system the first time the ASP group is varied on, if not already present. The value of the RMTLOCNAME parameter first

element for the automatically added entries will be *LOOPBACK, which maps to an IP address associated with the host system. If some other IP address is required for an ASP group, it will have to be specified manually either before it is varied on, by means of the ADDRDBDIRE command, or afterward by means of the CHGRDBDIRE command. <<

One entry naming each remote relational database or application requester driver program must also be added.

Each relational database name must be unique within the relational database directory and within the distributed network. >>

For more information on the relational database directory, refer to the Distributed Database Programming topic in the Information Center. <<

Restriction: If adding an entry for an SQLCI application requester driver (ARD) program using the ARDPGM parameter, you must have execute authority to the application requester driver program and library to specify the ARD program on this command.

Required Parameters

RDB Specifies the name of the relational database being added. Each name entered into the directory must be unique. Because relational databases communicate with each other over a distributed network, each name must also be unique among other relational databases on the network. The system can determine whether a relational database name is unique in the directory, but you must determine whether it is unique among relational databases on the network. A maximum of 18 characters can be specified for the relational database name. If a DB2* for MVS relational database is identified, only 16 characters are allowed.

Note: Valid RDB names must begin with a letter and consist of uppercase A-Z, 0-9, and underscore.

RMTLOCNAME

Specifies the name or address of the system on which the relational database is located. >>

***LOCAL:** The relational database is located on the system auxiliary storage pool (ASP number 1) and configured basic user ASPs (ASP numbers 2-32) on this system. You can specify *LOCAL for only one entry in the relational database directory.

Note: If *LOCAL is specified, the DEV, LCLLOCNAME, RMTNETID, MODE, TNSPGM, and ARDPGM parameters will be ignored, and the value of the second element is forced to *IP.

***LOOPBACK:** The relational database is accessed using the internet protocol loopback or LOCAL HOST address. The use of *LOOPBACK allows a DRDA connection to a database on the local system.

Note: If *LOOPBACK is specified, the DEV, LCLLOCNAME, RMTNETID, MODE, TNSPGM, and ARDPGM parameters will be ignored, and the value of the second element is forced to *IP.



***ARDPGM:** The relational database is located by using the application requester driver program specified on the ARDPGM parameter. A remote location name is not used to locate the relational database.

Note:

If *ARDPGM is specified, the PORT, DEV, LCLLOCNAME, RMTNETID, MODE, and TNSPGM parameters will be ignored.

remote-location-name: The first element of this parameter can take several forms:

- SNA remote location name (LU name). Specify a maximum of 8 characters for the remote location name. If this form is used, the second element of this parameter must be *SNA (the default).
- SNA remote network identifier and remote location name separated by a period. Specify a maximum of 8 characters for the remote location name, and a maximum of 8 characters for the remote network identifier. If this form of the parameter is used, the second element of this parameter must be *SNA (the default), and any value specified for the RMTNETID parameter must agree. If the RMTNETID parameter is not specified, the RMTNETID value will be set to agree with the RMTLOCNAME parameter.
- IP address in dotted decimal form. Specify an internet protocol address in the form nnn.nnn.nnn.nnn where each nnn is a number in the range 0 through 255. If this form is used, the second element of this parameter must be specified as *IP.
- IP host domain name. Specify an internet host domain name of up to 254 characters in length. If this form is used, the second element of this parameter must be specified as *IP.

If *IP is specified for the second element, the DRDA application server at the remote location must support the use of TCP/IP, and the DEV, LCLLOCNAME, RMTNETID, MODE, and TNSPGM parameters will be ignored.

If *IP is not specified, the application server must support SNA connectivity. More information about SNA remote location names can be found in the APPC, APPN, and HPR topic in the Information Center.

Optional Parameters

TEXT Specifies text that briefly describes the relational database.

***BLANK:** No text is associated with the new relational database.

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

PORT Specifies the TCP/IP port that is used at the remote location to communicate with the system on which the relational database is located. This parameter will be ignored if *IP is not specified in the RMTLOCNAME parameter.

***DRDA:** The DRDA well-known port of 446 will be used.

port-number: Specify a number in the range 1-65535.

service-name: Specify a maximum of 14 characters for the service name. This name must be registered in the service database file.

RMTAUTMTH

Specifies the preferred remote authentication method on a DDM/DRDA TCP/IP connection request. The actual method used depends on the outcome of the negotiation process between client and server, which depends on the cryptographic support available and the server security configuration. The CHGDDMTCPA (Change DDM TCP/IP Attributes) command can be used to configure DDM/DRDA TCP/IP security on iSeries servers. This parameter will be ignored if *IP is not specified in the **Remote location** (RMTLOCNAME parameter).

Element 1: Preferred method

Specifies the initial authentication method proposed to the server. Based on the authentication methods supported by the server and the value specified for the **Allow lower authentication** element of this parameter, an authentication method is negotiated that is acceptable to both the Application Requester and Application Server systems.

Possible values are:

***ENCRYPTED** User ID and associated encrypted password is sent on a DDM connection request. Cryptographic support must be available on both systems for this authentication method to be used.

***USRID** User ID only is sent on a DDM connection request. This is the lowest authentication method.

***USRIDPWD** User ID and associated password is sent on a DDM connection request. Passwords are not encrypted if this authentication method is used.

***KERBEROS** Authentication occurs using Kerberos. The relational database name must map to a target principal name in the Enterprise Identity Mapping (EIM) environment. Kerberos needs to be configured on both systems for this authentication method to be used.

Element 2: Allow lower authentication Specifies whether an authentication method lower than what was specified for the **Preferred method** element of this parameter will be accepted during negotiation with the Application Server system. If the Application Server system is configured to require a higher authentication method than the value specified for the **Preferred method** element of this parameter and the Application Requester system can support a higher authentication method, the negotiated authentication method can always be higher than the **Preferred method**. From highest to lowest, the authentication methods are:

*KERBEROS

*ENCRYPTED

*USRIDPWD

*USRID

Possible values are:

***ALWLOWER** Allow negotiation of a lower authentication method than what was specified for the Preferred method element of this parameter.

***NOALWLOWER** Do not allow negotiation of a lower authentication method than what was specified for the Preferred method element of this parameter.

DEV Specifies the name of the advanced program-to-program communications (APPC) device description on the local system that is used to access this relational database. The device description does not need to exist when the relational database directory entry is added.

More information on device names is in the APPC, APPN, and HPR topic in the Information Center.

***LOC:** If APPC is being used, the system determines which device description is used. If the advanced peer-to-peer networking (APPN) function is being used, the system ignores this parameter.

device-description-name: Specify a maximum of 10 characters for the name of a device description.

LCLLOCNAME

Specifies the local location name by which the local system is identified to the system on which the relational database is located. The local location name cannot be the same as the remote location name.

More information on local location names is in the APPC, APPN, and HPR topic in the Information Center.

***LOC:** If APPC is being used, the system determines which local location name is used. If the APPN function is being used, the system uses the default local location name specified in the network attributes.

***NETATR:** The local location name defined in the network attributes is used.

local-location-name: Specify a maximum of 8 characters for the local location name.

RMTNETID

Specifies the remote network identifier of the system on which the relational database is located. If this parameter is specified, the RMTLOCNAME parameter must be consistent with this RMTNETID parameter. If the RMTLOCNAME parameter specified a network ID, this parameter must agree (otherwise, an error message will be issued). If the RMTLOCNAME parameter does not specify any network ID, there is no possibility of conflict with this parameter.

More information on remote network identifiers is in the APPC, APPN, and HPR topic in the Information Center.

***LOC:** If APPC is being used, the system determines which remote network identifier is used. If the APPN function is being used, the system uses the local network identifier defined in the system's network attributes for the remote network identifier.

***NETATR:** The local network identifier defined in the local system's network attributes is used for the remote network identifier.

***NONE:** No remote network identifier is used.

remote-network-identifier: Specify a maximum of 8 characters for the remote network identifier.

MODE Specifies the mode name that is used with the remote location name to communicate with the system on which the relational database is located.

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

mode-name: Specify a maximum of 8 characters for the mode name.

More information on mode names is in the APPC, APPN, and HPR topic in the Information Center.

TNSPGM

Specifies the name of the transaction program to use with the relational database entry.

***DRDA:** The Distributed Relational Database Architecture* (DRDA*) transaction program name, X'07F6C4C2', is used. DRDA is a means by which relational databases communicate with each other over a distributed network.

transaction-program-name: Specify the transaction program name in either of the following formats:

- A 4-byte hexadecimal name, which is entered by enclosing the 8 hexadecimal digits in apostrophes with a prefix of X. For example, X'07F6C4C2' is a 4-byte hexadecimal name.
- An 8-byte character name, which is entered by specifying the name in its 8-character form.

ARDPGM

Specifies the qualified name of the application requester driver (ARD) that is the program to be called to process SQL requests directed to the relational database. The program must exist and must be of the object type *PGM.

***DRDA:** The Distributed Relational Database Architecture (DRDA) application requester is used (that is, no ARD program is used).

The possible library values are:

***LIBL:** The library list when the directory entry is added is used to locate the program name.

***CURLIB:** The current library when the directory entry is added is used to locate the program name. 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 the program name is located.

program-name: Specify the name of the application requester driver program to be called to process the SQL requests.

Examples for ADDRDBDIRE

Example 1: Adding an Entry

```
ADDRDBDIRE RDB(MYRDB) RMTLOCNAME(*LOCAL)
```

This command adds an entry to the relational database directory. The entry identifies the local relational database. In an SQL program, this relational database name is used when referring to the local relational database.

Example 2: Adding an Entry

```
ADDRDBDIRE RDB(YOURRDB) RMTLOCNAME(NEWYORK)
```

This command adds an entry to the relational database directory. The entry identifies a remote location, NEW YORK.

Example 3: Adding an Entry for an Application Requester Driver Program

```
ADDRDBDIRE RDB(YOURRDB) RMTLOCNAME(*ARDPGM)
ARDPGM(MYLIB/MYPGM)
```

This command adds an entry to the relational database directory. The entry indicates that access to relational database YOURRDB will be done by an application requester driver program named MYPGM in the library MYLIB.

Example 4: Adding an Entry for TCP/IP usage

```
ADDRDBDIRE RDB(TCPRDB)
RMTLOCNAME(ROCHESTER.XYZ.COM *IP)
PORT(*DRDA)
```

This command adds an entry to the relational database directory. The entry specifies that the remote RDB associated with the RDB name of TCPRDB uses TCP/IP and is on the host with the domain name of ROCHESTER.XYZ.COM, and listens on the standard DRDA port of 446 (*DRDA is the default port so the PORT parameter is unnecessary in this case).

Example 5: Adding an Entry for TCP/IP using dotted decimal IP address and a numeric port number


```
ADDRDBDIRE RDB(DB2DSYS)
  RMTLOCNAME('9.5.36.17' *IP)
  PORT(5021)
```

This command adds an entry to the relational database directory. The entry specifies that the remote RDB associated with the RDB name of DB2DSYS uses TCP/IP and is on the host with an IP address of 9.5.36.17, and listens on port 5021. A System/390 MVS installation, for example, can have multiple DB2 subsystems, and TCP/IP can support only one server at each port number, so port numbers other than 446 are sometimes required.

Example 6: Adding an Entry for TCP/IP using a service name for the port identification

```
ADDRDBDIRE RDB(DB2ESYS)
  RMTLOCNAME(ROCHESTER.XYZ.COM *IP)
  PORT(DB2ESYS_PORT)
```

This command uses a service name to specify the port number when adding a new entry. OS/400 will attempt to resolve the name DB2ESYS_PORT to a port number by use of the TCP/IP Service Table. In order for the name to be properly resolved, an entry for DB2ESYS_PORT must exist in the TCP/IP Service Table. The WRKSRVTBLE or CFGTCP command can be used to update the service table.

Error messages for ADDRDBDIRE

***ESCAPE Messages**

CPF3EC0

Add relational database directory entry failed.

ADDRMTDFN (Add Remote Definition) Command Description

ADDRMTDFN Command syntax diagram

Purpose

The Add Remote Definition (ADDRMTDFN) command is used to define the attributes of a remote system and add them to the remote system definition table.

Restriction: You must have *ALLOBJ authority to use this command.

Required Parameter

SYSTEM

Specifies the system name and system group of the remote system being defined.

***ANY:** A default definition is added to be used for all remote systems whose attributes are not yet defined.

Element 1: System Name

system-name: Specify the name of the remote system being defined.

Element 2: System Group

system-group: Specify the group name of the remote system being defined. The system group name is blank if this value is not specified.

Optional Parameters

TEXT Specifies the text that briefly describes the remote system 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.

MTGNTCDOC

Specifies the type of meeting notice documents accepted by the remote system. If the system can accept Enterprise Meeting Notice Architecture documents (post-V2R1M1 AS/400 systems), you should specify *EMN for this parameter. If you are unsure, specify *FFTDCA for this parameter.

***FFTDCA:** Final-form text documents are accepted. The remote system does not accept enterprise meeting notice documents.

***EMN:** Enterprise meeting notice documents are accepted.

CALDTASTM

Specifies the type of calendar data stream that the local system uses when sending a request for calendar information to this remote system. Each type of calendar data stream represents a format in which remote calendar requests are made from the local system to this remote system.

***OV400:** The OfficeVision calendar data stream is used.

calendar-data-stream: Specify the name of the calendar data stream that is used. The name of the data stream can be a maximum of 10 characters.

Single Value

***NONE:** No calendar data stream is used.

RMTCALPWD

Specifies the password that is associated with user profile QRMTCAL on the remote system. This user profile is used to sign on to the remote system when processing a request for calendar information.

***NONE:** No password is used for user profile QRMTCAL.

calendar-password: Specify the password that is defined for QRMTCAL. If the password is numeric, it must begin with a Q (for example, specify Q1234 when 1234 is the password).

RMTUSRAUT

Specifies the object authority for calendar objects on the local system to be used for incoming requests for calendar information from remote system users. This parameter is used by OfficeVision calendar processing to determine authority to calendars.

***PRIVATE:** Private authority is used for requests from the remote system. If private authority does not exist, public authority is used.

***PUBLIC:** Public authority is used for requests from the remote system.

***MINIMUM:** The lesser of the private or the public authority is used for requests from the remote system.

***EXCLUDE:** Local system objects cannot be accessed by users on the remote system.

RMTLOCNAME

Specifies the remote location name of the remote system being added.

***SYSTEM:** The name specified on the SYSTEM parameter is used for the remote location name.

remote-location-name: Specify the full name of a remote location.

LCLLOCNAME

Specifies the location name that identifies the local system to the remote system being added.

***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 the name of the local location.

RMTNETID

Specifies the remote network identifier (ID) for the remote system being added.

***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 the remote network ID.

MODE Specifies the name of the mode that defines the device sessions used to request data from the remote system.

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

mode-name: Specify the name of the mode.

Examples for ADDRMTDFN

Example 1: Adding a Specific Remote Definition

```
ADDRMTDFN  SYSTEM(ABCXYZ)  TEXT('System XYZ')
           MTGNTCDOC(*EMN)
```

This command adds a definition for a remote system ABCXYZ and allows the system to accept enterprise meeting notice documents.

Example 2: Allowing Final Form Text Documents

```
ADDRMTDFN  SYSTEM(*ANY)  MTGNTCDOC(*FFTDCA)
```

This command allows all remote systems that do not have specific remote definitions to accept final form text meeting notices.

Example 3: Adding a Remote Definition with Password

```
ADDRMTDFN  SYSTEM(DALLAS1) TEXT('SYSTEM1')
           MTGNTCDOC(*EMN)  RMTCALPWD(CALPWD)
           LCLLOCNAME(*NETATR)
```

This command adds a definition for the remote system DALLAS1, which accepts enterprise meeting notice documents. The password to sign on the system is CALPWD. The remote system will identify the local system by the name specified in the system network attributes.

Error messages for ADDRMTDFN

*ESCAPE Messages

CPF6DCA

SYSTEM parameter cannot be local system.

CPF6DCB

Remote definition for system &1 &2 already exists.

CPF9899

Error occurred during processing of command.

ADDRMTJRN (Add Remote Journal) Command Description

ADDRMTJRN Command syntax diagram

Purpose

The Add Remote Journal (ADDRMTJRN) command associates a remote journal on the target system, as identified by the relational database directory entry, with the specified journal on the source system. The journal on the source system may be either a local journal or another remote journal. A maximum of 255 remote journals may be associated with a single journal on a source system.

When adding a remote journal to a source journal, the remote journal is created on the target system using a combination of the attributes from the source journal and the input parameters provided on this command. The library that the remote journal will be created in must already exist on the target system prior to this command being used on the source system. When created by this command, the remote journal will be created with a journal type of *REMOTE and the remote journal will not have an attached journal receiver.

Note: A receiver will be attached when the remote journal is activated using either the Change Remote Journal (CHGRMTJRN) command, or Change Journal State (QjoChangeJournalState) API.

When adding the remote journal, the remote journal can either be created into the same named library as that of the source journal or into a redirected library on the target system. A **redirected library** provides a means for remote journals and any of their associated journal receivers to reside in different named libraries on the target system from the corresponding local journal and journal receivers on the local system. When specified, all validation for the journal library on the target system will be performed using the redirected library name. Similarly, the journal receivers that will later be created and associated with this remote journal can either reside in the same library as the source journal receivers on the source system, or into a distinct redirected library name on the target system. The journal receiver library redirection, if desired, must be specified when the remote journal is added using this command.

When adding a remote journal on a target system, two remote journal types can be specified, *TYPE1 and *TYPE2. The remote journal type influences the redirection capabilities, journal receiver restore operations, and remote journal association characteristics. See the Journal management topic in the Information Center for detailed descriptions of the differences.

If the specified journal already exists on the target system, the journal can be associated with the source journal if all of the following are true:

- the journal is of type *REMOTE
- the remote journal type matches the specified remote journal type
- the remote journal was previously associated with this same source journal

Also, the journal may or may not have an attached journal receiver.

After the remote journal has been successfully added on the target system, the remote journal will have a journal state of *INACTIVE. A journal state of *INACTIVE for a remote journal means that the remote journal is currently not ready to receive journal entries from its source journal on the source system. The Change Remote Journal (CHGRMTJRN) command or Change Journal State (QjoChangeJournalState) API is used to activate a remote journal and start the replication of journal entries from the source journal to the remote journal.

Once a remote journal has been added to a journal, the receiver that was attached to the source journal at the time of running this command or any subsequently attached receivers, will be protected from deletion if all journal entries for a given journal receiver have not yet been replicated to the remote journal. This

protection ends when the remote journal is removed using the Remove Remote Journal (RMVRMTJRN) command or Remove Remote Journal (QjoRemoveRemoteJournal) API.

Restrictions:

- The Add Remote Journal (ADDRMTJRN) command may only be called from the source system.
- A user profile must exist on the target system by the same name as the user profile that is running the Add Remote Journal (ADDRMTJRN) command on the source system. This restriction is irrespective of the selected communications protocol.
- When adding a *TYPE1 remote journal to a source journal, the same journal and journal receiver library redirection must be specified that exists for any *TYPE1 remote journals which have already been added to the source journal. A remote journal will always use the redirected library, if any, that is specified for the local journal. The only way to change the value specified in the remote journal library field and the remote journal receiver library field is to do all of the following:
 1. Remove all of the associated *TYPE1 remote journals from the local journal.
 2. Delete the remote journal.
 3. Change the local journal to attach a new receiver.
 4. Add the remote journal specifying the new redirection.
- QTEMP cannot be specified for the remote journal library, remote journal receiver library, or remote message queue library.
- A remote journal whose name starts with a Q cannot specify a remote journal library that starts with a Q, unless the remote journal library is QGPL. This is required to prevent collisions between local and remote journals that are used for system functions.
- A *TYPE1 remote journal cannot be added to a *TYPE2 remote journal.
- The specified relational database directory entry (RDB) must meet the following rules:
 - The communications protocol must be one of the remote journal function supported protocols.
 - The remote location name in the RDB cannot refer to the *LOCAL database.
 - The RDB cannot use an application requester driver program (*ARDPGM) to locate the target system.
- >> The remote journal message queue on the remote system must be either in the same ASP group as the remote journal, or in the system ASP, or a basic user ASP.
- The remote receiver library and remote journal library on the remote system must both exist in either the system and basic user ASPs or in the same ASP group. They cannot be in two different ASP groups. <<

Required Parameters

RDB The name of the relational database directory entry that contains the remote location name of the target system. >> This name should match the name of the *LOCAL relational database directory entry on the target system. <<

relational-database-entry: Specify a maximum of 18 characters for the name of the relational database directory entry.

SRCJRN

The name of the journal on the source system to which the remote journal is being added, and the library where it resides. The journal on the source system may be either a local journal or another remote journal.

The name of the source journal 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.

source-journal-name: Specify the source journal to which the target journal is being added.

Optional Parameters

TGTJRN

Specifies the name of the remote journal on the target system.

***SRCJRN:** The target journal name is exactly the same as the source journal name.

library-name/target-journal-name: Specify the target journal that is being added as a remote journal to the source journal.

RMTRCVLIB

Specifies the name of the library for the remote journal receivers on the target system that will be associated with this remote journal.

***SRCRCVLIB:** The journal receivers are created on the target system in the same library as they exist on the source system.

remote-journal-receiver-library-name: Specify the name of the library for the remote journal receivers on the target system that will be associated with this remote journal.

RMTJRNTYPE

Specifies the type of remote journal on the target system. The remote journal type influences the redirection capabilities, journal receiver restore operations, and remote journal association characteristics. See the Journal management topic in the Information Center for detailed descriptions of the differences.

***TYPE1:** A *TYPE1 remote journal is added.

***TYPE2:** A *TYPE2 remote journal is added.

MSGQ

Specifies the name of the message queue associated with the remote journal. This value is only set for a journal that is created on the target system.

QSYS/QSYSOPR: The message is sent to the QSYSOPR message queue.

library-name/journal-message-queue: Specify the name of the journal message queue to which the journal messages are sent. If this message queue is not available when a message is to be sent, the message is sent to the QSYSOPR message queue.

DLTRCV

Specifies whether the system deletes the target journal receivers when they are no longer needed or keeps them on the target system for the user to delete after they have been detached by the target system. This value is only set for a journal that is created on the target system.

***NO:** The journal receivers are not deleted by the system.

***YES:** The journal receivers are deleted by the system.

» DLTRCVDLY

Specifies the time (in minutes) to be used to delay the next attempt to delete a target journal receiver associated with the remote journal. This value is only set for a journal that is created on the target system.

10: When the system cannot allocate an object needed to delete a journal receiver associated with the remote journal on the target system, it will wait 10 minutes before trying again.

delete-receiver-delay-time: When the system cannot allocate an object needed to delete a journal receiver associated with the remote journal on the target system, it will wait the specified number of minutes before trying again. Valid values range from 1 through 1440. <<

TEXT The text that briefly describes the remote journal on the target system. This value is only set for a journal that is created on the target system.

***BLANK:** Text is not specified.

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

Example for ADDRMTJRN

Example 1: Adding a *TYPE1 remote journal, specifying values for the DLTRCV and TEXT parameters.

```
ADDRMTJRN SRCJRN(LOCLIB/J) RDB(DETROI) TGTJRN(RMTLIB/J)
DLTRCV(*NO) TEXT('Remote Journal Created for Application Z')
```

This command adds remote journal J in library RMTLIB to journal J in library LOCAL, and the DLTRCV parameter on the remote journal will be *NO, irrespective of the attribute of journal J in library LOCLIB. If journal J in RMTLIB does not already exist, then it is created, otherwise, it is reassigned with journal J in LOCLIB, if it meets the appropriate criteria.

Error messages for ADDRMTJRN

*ESCAPE Messages

CPF69A4

Remote journal &1 in &2 not added.

CPF695A

Remote journal &1 in &2 not added.

CPF695B

Remote journal &1 in &2 not added.

CPF695C

Remote journal &1 in &2 not added.

CPF695D

Remote journal &1 in &2 not added.

CPF695E

Remote journal &1 in &2 not added.

CPF695F

Remote journal &1 in &2 not added.

CPF6973

Systems not compatible.

CPF6982

Relational database directory entry &1 not valid.

CPF6983

Remote journal &1 in &2 not added.

CPF6984

Remote journal &1 in &2 not added.

- CPF6985**
Remote journal &1 in &2 not added.
- CPF6988**
Remote journal &1 in &2 not added.
- CPF6989**
Remote journal &1 in &2 not added.
- CPF699B**
User profile &8 not found.
- CPF6991**
Remote journal &1 in &2 not added.
- CPF70DB**
Remote journal function failed.
- CPF70D6**
Remote journal ended, reason code &6.
- CPF701B**
Journal recovery of an interrupted operation failed.
- CPF7010**
Object &1 in &2 type *&3 already exists.
- CPF7011**
Not enough storage or resources.
- CPF9801**
Object &2 in library &3 not found.
- CPF9802**
Not authorized to object &2 in &3.
- CPF9803**
Cannot allocate object &2 in library &3.
- CPF9810**
Library &1 not found.
- CPF9820**
Not authorized to use library &1.
- CPF9830**
Cannot assign library &1.

ADDRPYLE (Add Reply List Entry) Command Description

ADDRPYLE Command syntax diagram

Purpose

The Add Reply List Entry (ADDRPYLE) command is used to add an entry to the system-wide automatic inquiry message reply list. The automatic message reply list is the source for default responses to inquiry messages. Each entry in the inquiry message list specifies both a message identifier and the reply that is sent when that message is sent as an inquiry message. The entry may also include comparison data, which further qualifies the message identifier. The message identifier may be specific or generic in scope. One of the following actions may be taken when one of the specific inquiry messages is issued:

- The default reply specified in the inquiry message file is sent to the message reply queue specified when the inquiry message was sent.

- A specific reply to the inquiry message is sent to the message reply queue specified when the inquiry message was sent.
- A manual reply to the inquiry message may be required from the operator.

The entry may also specify the dumping of information associated with the job that is sending the inquiry message.

The reply list is used only when an inquiry message is sent by a job that has the inquiry message reply attribute specified as INQMSGRPY(*SYSRPLY). The INQMSGRPY attribute can be changed by using the CHGJOB command.

Specific attributes of a reply list entry can be changed by using the Change Reply List Entry (CHGRPLY) command. Each reply list entry remains in the list until it is removed by the Remove Reply List Entry (RMVRPLY) command. The list can be shown by using the Work with Reply List Entry (WRKRPLY) command.

Restriction: This command is shipped with public *EXCLUDE authority and the QPGMR user profile has private authority to use the command.

Required Parameter

SEQNBR

Specifies the sequence number of the reply list entry being added to the reply list. The message identifier and message data of an inquiry message are matched against the reply list entry message identifiers and comparison data in ascending sequence number order. The search ends when a match occurs or when the last reply list entry is passed. Therefore, if more than one reply list entry matches the inquiry message identifier and comparison data, only the first entry that matches is used. If no reply list entry matches the inquiry message, the inquiry is sent, but no automatic reply is sent unless a default reply is sent, and the information associated with the job is not dumped. Default replies are sent when the delivery mode of the message queue is set to *DFT or when the inquiry message is sent to an external message queue in a batch job.

Sequence numbers range from 0001 through 9999 and duplicate sequence numbers are not allowed.

Optional Parameters

MSGID

Specifies the inquiry message identifiers for which automatic system action is taken. The message identifier can be either specific or generic in scope. Only predefined messages (messages known to the system by a message identifier) can be matched by reply list entries. Immediate messages cannot be used for comparison.

If no comparison data is specified, then only the message identifier is used to match the message to this reply list entry. If this is the first message reply list entry that matches the message, the action specified in this entry is taken.

***ANY:** This reply list entry matches all message identifiers. Unless comparison data is specified for this reply list entry, all reply list entries with a sequence number greater than this one are ignored.

message-identifier: Specify a reply message identifier that is compared with the inquiry message identifier. The message identifier must be seven characters in length and in the format, pppnnnn.

The first three characters (ppp) must be a code consisting of one alphabetic character followed by two alphanumeric (alphabetic or decimal) characters. The last four characters (nnnn) may consist of the decimal numbers 0 through 9 and the characters A through F.

To specify a generic message identifier, enter zeros in the rightmost two or four positions of the numeric field, such as pppnn00 or ppp0000. For example, CPA0000 would match any CPA inquiry message, while CPA4200 would match any CPA42xx inquiry message.

CMPDTA

Specifies the comparison data that is used to determine whether this entry matches an inquiry message. This parameter is made up of comparison data and a message data start value. If the identifier of the inquiry message matches the message identifier of this reply list entry, then the message data specified for the inquiry message is compared to this data. If a message data start value has not been specified, then the first part of the message data (up through the first 28 characters or less) must exactly match the comparison data specified here before the action requested for this reply list entry is taken. However, if a start value has been specified, then the part of the message data beginning with the character position specified in the start value must exactly match the comparison data before any requested action is taken. If the comparison data is longer than the message data, then no match occurs. If no comparison data is specified, then only the message identifier is used to match the message to this reply list entry. If, in the message reply list, this is the first entry that matches the message, then the action specified in this entry is taken.

Message data for an inquiry message may be specified in the MSGDTA parameter of the SNDUSRMSG or SNDPGMMSG commands for the inquiry message.

Element 1: Comparison Data

***NONE:** No comparison data is specified; if the inquiry message has the specified identifier, the action specified by this reply list entry is taken.

'comparison-data': Specify a character string of up to 28 characters (enclosed in apostrophes if blanks or other special characters are included). This string is compared with a string of the same length in the message data portion of the inquiry message, beginning with the first character (if no start value has been specified). If the comparison data string matches the inquiry message data string, the action specified by this reply list entry is taken.

Element 2: Message Data Start Position

***NONE:** No message data start value is specified. If a comparison value is specified and a message data start value is not specified, a default value of 1 is used.

message-data-start: Specify the character position in the message's replacement text (maximum value is 999) where the comparison of the comparison data and the replacement text starts. A start value is not valid without a specification of comparison data.

Coded Character Set Identifier (CCSID) Considerations

The text supplied on the CMPDTA parameter that corresponds to the *CCHAR type field 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 *CCHAR type field see the Add Message Description (ADDMSGD) command.

RPY Specifies how to reply to an inquiry message that matches this reply list entry. The reply specified (other than *RQD) in this reply list entry is automatically sent by the system without requiring user intervention. The inquiry message does not cause the job to be interrupted or notified when the message arrives at the message queue. The inquiry message is not shown before the reply message is sent and there is no opportunity for a manual reply to the inquiry message.

***DFT:** The default reply to the inquiry message is sent. If no default reply is specified in the message description of the inquiry message, the system default reply, *N, is used.

***RQD:** The inquiry message requires an explicit reply. If the message queue to which the inquiry is sent is in break mode, the message interrupts and is displayed. If the message queue is in the notify mode, the job to which it is allocated is notified. No reply is automatically sent. If the message queue is in the hold mode, the message remains on the queue until it is removed.

'message-reply': Specify a character string of up to 32 characters, enclosed in apostrophes if blanks or other special characters are included, that is sent as a reply to the inquiry message. If this reply is not valid for the inquiry message, the inquiry is sent as if RPY(*RQD) had been specified.

DUMP Specifies whether the job that sent the inquiry message is dumped. The dump is the same as the dump specified by DMPLST(*JOB) on the Add Message Description (ADDMSGD) command or by OUTPUT(*PRINT) on the Display Job (DSPJOB) command for the sending job. A job dump may be requested regardless of the value specified for the RPY parameter.

***NO:** The job is not dumped.

***YES:** The job is dumped before control returns to the program that is sending the message.

CCSID

Specifies the coded character set identification (CCSID) that the text specified on the CMPDTA parameter that corresponds to the *CCHAR type field is to be considered in.

When an inquiry message is sent in a job that is using the system reply list, the *CCHAR replacement data is converted to the CCSID of the CMPDTA that is stored in the system reply list before the comparison is made.

All other compare data is not converted before a comparison is made. For more information about the message handler and its use of CCSIDs, see the Globalization topic in the Information Center.

Note:

When specifying a CCSID other than *HEX, all CMPDTA specified is converted from that CCSID to the job CCSID when displayed on the Work with Reply List Entries panel. This occurs even when all CMPDTA does not correspond with *CCHAR data; therefore, when using a CCSID other than *HEX, specifying the length of the *CCHAR data or any other data field is not recommended.

***HEX:** The CMPDTA that corresponds to the *CCHAR data type field is assumed to be 65535. No conversion occurs before the replacement data is compared with the CMPDTA.

***JOB:** The CMPDTA that corresponds to the *CCHAR data type field is assumed to be in the CCSID of the job calling this command.

coded-character-set-identifier: Specify the CCSID you want the CMPDTA that corresponds to *CCHAR data type field to be considered in.

Examples for ADDRPLYE

Example 1: Reply Automatically Sent

```
ADDRPLYE SEQNBR(10) MSGID(RPG1241) RPY(G)
```

This command adds a reply list entry to the reply list for message identifier RPG1241 (database record not found). Whenever a RPG1241 inquiry message is sent by a job that is using the reply list, a reply of 'G' is automatically sent. The inquiry does not cause a job that has allocated the message queue to be interrupted or notified when the inquiry arrives, and no opportunity is given to reply to the message. The sending job does not have a job dump processed.

Example 2: Default Reply is sent; Job Dump Processed

```
ADDRPLYE SEQNBR(25) MSGID(RPG1200)  
RPY(*DFT) DUMP(*YES)
```

This command adds a generic reply list entry to the reply list for all RPG12xx messages. Whenever an RPG12xx inquiry message is sent by a job that is using the reply list, the equivalent to DSPJOB OUTPUT

(*PRINT) is automatically generated. The default reply will automatically be sent. This is either the default reply specified in the message description or (if none is specified in the message description) the system default reply. The inquiry does not cause a job that has allocated the message queue to be interrupted or notified when the inquiry arrives, and no opportunity is given to reply to the message. The sending job is dumped before control returns to the sending program. Note that because of the sequence numbers, the entry added by the previous example overrides this entry for message identifier RPG1241.

Example 3: Adding a Generic Reply List Entry

```
ADDRPYLE SEQNBR(30) MSGID(RPG0000)
RPY(D) DUMP(*YES)
```

This command adds a generic reply list entry to the reply list for all RPG messages. Whenever an RPG inquiry message is sent by a job that is using the reply list, a reply of 'D' is sent automatically. The inquiry does not cause a job that has allocated the message queue to be interrupted or notified when the inquiry arrives, and no opportunity is given to reply to the message. (If a value of D is not valid for a particular RPGxxx message, the user must reply as if *RQD were specified for the RPY parameter.) The sending job is dumped before control returns to the sending program. Note that the entries added by the previous two examples will override this entry for all RPG12xx messages.

Example 4: System Reply List for Spooled Output

```
ADDRPYLE SEQNBR(40) MSGID(CPA5316)
CMPDTA('QPSPLRT QSYS QSYSPRT') RPY(*RQD)
```

This command illustrates how to use the system reply list for spooled output for device QSYSPRT. The file and library name for spooled output is QSYS/QPSPLRT.

When compare value is specified, it is compared to the message data beginning with replacement variable &1. If the significant field appears in replacement variable &3, the compare value must include a value for replacement variables &1 and &2, or a message data start value may be entered to begin the comparison with replacement variable &3.

The message CPA5316 has a replacement data as follows:

&1	ODP file name	&CHAR	10
&2	ODP library name	&CHAR	10
&3	ODP device name	&CHAR	10

A compare for device name 'QSYSPRT' in replacement variable &3 must be preceded by values for &1 and &2 if a message data start value is not entered. Blanks are significant.

The message data of QSYSPRT is the DEVICE name as defined in the CPA5316 message. Whenever a CPA5316 inquiry message with comparison data of QSYSPRT is sent by a job that is using the reply list, the operator must make a manual reply to the inquiry. If the message queue to which the inquiry is sent is in break mode, the inquiry message interrupts. A reply is not sent (unless the queue is in the default mode or the message is sent to an external message queue in a batch job), and no job dump is taken.

Another reply list entry identical to the one listed above could be added, but with a different sequence number and with CMPDTA(WSPR01) specified. This would allow a unique response to a message based on the type of printer.

Example 5: Adding Reply List Entry For Any Message Identifier

```
ADDRPYLE SEQNBR(9999) MSGID(*ANY)
RPY(*RQD) DUMP(*YES)
```

This command adds a reply list entry to the reply list for any message identifier. This entry applies to any predefined inquiry message that is not matched by an entry with a lower sequence number. A manual reply to the inquiry message is required for any predefined inquiry message not matched by a previous

entry. If the message queue to which the inquiry message is sent is in break mode, the message interrupts. The job that sent the inquiry message is dumped (equivalent to DSPJOB OUTPUT(*PRINT)).

Example 6: Using Comparison Data

```
ADDRPYLE SEQNBR(5) MSGID(CPA5316) CMPDTA(QSYSPRT 21)
      RPY(I) DUMP(*NO)
```

Assume that the message CPA5316 is sent to QSYSOPR with the message replacement text of TESTEDFILESTLIBRARYQSYSPRT; because there is a match for MSGID, the message replacement text starting in position 21 (message data start) is tested by comparing it with the comparison data (for the length of the comparison data). This is a match because QSYSPRT = QSYSPRT, and therefore the reply of 'I' is sent.

Error messages for ADDRPYLE

*ESCAPE Messages

CPF2435

System reply list not found.

CPF2436

System Reply List entry not added or changed.

CPF247E

CCSID &1 is not valid.

CPF2499

Message identifier &1 not allowed.

CPF2555

Sequence number &1 already defined in system reply list.

CPF2557

System reply list damaged.

CPF2558

System reply list currently in use.



ADDRSCCRQA (Add Resource CRQ Activity) Command Description

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

ADDRSCCRQA Command syntax diagram

Purpose

The Add Resource Change Request Activity (ADDRSCCRQA) command adds an activity to a change request description that performs an IPL of a managed iSeries server or restarts a non-OS/400 resource such as a PS/2 with NetView DM/2 installed.

Restrictions:

1. You must have *CHANGE authority to the change request description and *EXECUTE to the library.
2. A resource other than *SYSx is not supported for iSeries managed systems.
3. If a value for NODL is specified, the node list can only contain entries that have a value of *SNA for the address type.

4. Keylock must be in NORMAL position or the request is rejected.

Notes:

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 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.

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.

ACTION

Specifies the functions to be performed on the resource.

***RESTART:** Use to restart the specified resource. If the resource specified is *SYS, *SYSA, or *SYSB, the managed system stops and restarts.

Optional Parameters

RSC Specifies the resource name.

***SYS:** The action is performed against the entire system. For the restart action, this means that the managed system is powered down and restarted. The system panel determines the IPL source.

***SYSA:** This is the same as *SYS except the IPL source is the A side.

***SYSB:** This is the same as *SYS except the IPL source is the B side.

resource-name: Specify a 16-character resource name on the managed system. This is not supported by iSeries managed systems.

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 performed. Control point names cannot be specified if the node list (NODL) parameter is specified.

***NONE:** The systems on which this activity is 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 performed.

control-point-name: Specify the APPN control point name of the managed system on which the activity is 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.

OPTION

Specifies how to end. It specifies whether the system allows the active subsystems to end processing of active jobs in a controlled manner, or whether the system ends the job immediately.

***CNTRLD:** The jobs are ended in a controlled manner. If the jobs cannot be ended in a controlled manner during the delay period, they are ended immediately.

***IMMED:** Jobs are ended immediately.

DELAY

Specifies the number of seconds the system waits for the jobs to be ended in a controlled manner.

3600: The system waits one hour before ending the jobs immediately.

delay-time: Specify the delay time in seconds.

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:** The possible single value is *NONE.

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 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 run before this activity.

Element 2: Relational Operator

This element is the relational operator to use when comparing the end code from an 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 is 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 the 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 the user before it completes.
 - 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

The 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 can run for that same node even though the conditioning activity specified cannot have completed for all other nodes. In the case where this activity can 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 when 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 is submitted on the date specified on the start before data element.

start-before-time: Specify the time before the activity must start. If the activity cannot be started before this time then 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 on any date after 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 when 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 when the functions start on the managed system. 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.

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.

TEXT Specifies the activity description.

***GEN:** A description is generated.

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

Examples for ADDRSCCRQA

Example 1: Adding an Activity

```
ADDRSCCRQA CRQD(MYLIB/CR1) ACTIVITY(ACT01)
ACTION(*RESTART) RSC(*SYS)
STRTIME((04:00:00 *NEXT)) NODL(NETLIB/STORES)
```

This command adds an activity to re-IPL the store systems at 4:00 a.m. tomorrow. These store systems are identified in the STORES node list.

Example 2: Performing an IPL

```
ADDRSCCRQA CRQD(MYLIB/CR1) ACTIVITY(ACT01)
ACTION(*RESTART) RSC(*SYSA)
CPNAME((*NETATR SYS1)) OPTION(*IMMED)
```

This command adds an activity to perform an IPL of the system SYS1 immediately to the A side.

Error messages for ADDRSCCRQA

*ESCAPE Messages

None <<

ADDREXBUF (Add REXX Buffer) Command Description

ADDREXBUF Command syntax diagram

Purpose

The Add REXX Buffer (ADDREXBUF) command allows the user to create a buffer in the REXX external data queue.

Optional Parameter

BUFFER

Specifies the name of the variable that receives the number of the new buffer. In a control language (CL) program, a decimal variable with a minimum length of 11 digits and no decimal position must be specified.

Example for ADDREXBUF

```
ADDREXBUF
```

This command creates a logical buffer within the REXX external data queue.

Error messages for ADDREXBUF

*ESCAPE Messages

CPF7CF7

REXX external data queue is damaged.

CPF7CF8

REXX external data queue is full.

ADDRTGE (Add Routing Entry) Command Description

ADDRTGE Command syntax diagram

Purpose

The Add Routing Entry (ADDRTGE) command adds a routing entry to the specified subsystem description. The associated subsystem can be active at the time. Each routing entry specifies the parameters used to start a routing step. For example, the routing entry specifies the name of the program to run when the routing data that matches the compare value in this routing entry is received.

Restriction: To use this command, the user must have object operational and object management authorities for the subsystem description.

Required Parameters

SBSD Specifies the qualified name and library of the subsystem description to which the routing 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 to which the routing entry is added.

SEQNBR

Specifies the sequence number of the routing entry that is added. Routing data is matched against the routing entry compare values in ascending sequence number order. Searching ends when a match occurs or the last routing entry is compared. Therefore, if more than one match possibility exists, only the first match is processed. Specify a unique sequence number (ranging from 1 through 9999) that identifies the routing entry.

CMPVAL

Specifies a value that is compared with the routing data to determine whether this is the routing entry used for starting a routing step for the job. If the routing data matches the routing entry compare value, that routing entry is used. Optionally, a starting position in the routing data character string can be specified for the comparison.

Element 1: Comparing Values with Routing Data

***ANY:** Any routing data is considered a match. To specify *ANY, the routing entry must have the highest SEQNBR value of any routing entry in the subsystem description.

compare-value: Specify a value (any character string not exceeding 80 characters in length) that is compared with routing data for a match. When a match occurs, this routing entry is used to start a

routing step. A starting position in the routing data character string can be specified for the comparison; if no position is specified, 1 is assumed.

Element 2: Starting Position

1: The comparison between the compare value and the routing data begins with the first position in the routing data character string.

starting-position: Specify a value, ranging from 1 through 80, that indicates which position in the routing data character string is the starting position for the comparison. The last character position compared must be less than or equal to the length of the routing data used in the comparison.

PGM Specifies the qualified name of the program called as the (first) program run in the routing step. No parameters can be passed to the specified program.

The program name can be either explicitly specified in the routing entry or extracted from the routing data. If a program name is specified in a routing entry, selection of that routing entry results in the routing entry program being called regardless of the program name passed in an EVOKE function. If the program specified in the EVOKE function is called, PGM(*RTGDTA) must be specified in the routing entry. If the program does not exist when the routing entry is added, a library qualifier must be specified because the qualified program name is retained in the subsystem description.

***RTGDTA:** The program name is taken from the routing data that was supplied and matched against this entry. A qualified program name is taken from the routing data in the following manner: the program name is taken from positions 37 through 46, and the library name is taken from positions 47 through 56. Care should be used to ensure that routing entries that specify *RTGDTA are selected only for EVOKE functions on jobs that have specified the program name in the correct position in the routing data.

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 program that is run from this routing entry.

Optional Parameters

CLS Specifies the qualified name of the class used for the routing steps started through this routing entry. The class defines the attributes of the running environment for processing the routing step associated with this routing entry. If the class does not exist when this routing entry is changed, a library qualifier must be specified because the qualified class name is retained in the subsystem description. More information is in Commonly used parameters.

***SBSD:** The class having the same qualified name as the subsystem description specified by the SBSD parameter is used for routing steps started through this entry.

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 that is used for routing steps started through this routing entry.

MAXACT

Specifies the maximum number of routing steps (jobs) that can be active at the same time through this routing entry. In a job, only one routing step is active at a time. When a subsystem is active and the maximum number of routing steps is reached, any subsequent attempts to start a routing step through this routing entry fails. If the routing data is entered interactively, an error message is sent to the user. Otherwise, the job is ended, and a message is sent by the subsystem to the job's log. More information on this parameter is in Commonly used parameters.

***NOMAX:** There is no maximum number of routing steps that can be active at the same time and be processed through this routing entry. This value is normally used when there is no reason to control the number of routing steps.

maximum-active-jobs: Specify the maximum number of routing steps that can be active at the same time and be processed through this routing entry. If a routing step being started would exceed this number, the job is implicitly ended.

POOLID

Specifies the pool identifier of the storage pool in which the program runs.

1: Storage pool 1 of this subsystem is the pool in which the program runs.

pool-identifier: Specify the identifier of the storage pool defined for this subsystem in which the program runs. Valid values range from 1 through 10.

Examples for ADDRTGE

Example 1: Adding to the Routing Portion of a Subsystem Description

```
ADDRTGE SBS(ORDLIB/PERT) SEQNBR(46)
      CMPVAL(WRKSTN2) PGM(ORDLIB/GRAPHIT)
      CLS(MYLIB/AZERO) MAXACT(*NOMAX) POOLID(2)
```

This command adds routing entry 46 to the routing portion of subsystem description PERT in the ORDLIB library. To use routing entry 46, the routing data must start with the character string WRKSTN2 starting in position 1. Any number of routing steps can be active through this entry at any one time. The program GRAPHIT in the library ORDLIB is to run in storage pool 2 by using class AZERO in library MYLIB.

Example 2: Adding to the Subsystem Description

```
ADDRTGE SBS(QGPL/ABLE) SEQNBR(5) CMPVAL(XYZ)
      PGM(QGPL/REORD) CLS(LIBX/MYCLASS) MAXACT(*NOMAX)
```

This command adds routing entry 5 to the subsystem description ABLE in the QGPL library. The program REORD in the general purpose library is started and uses the class MYCLASS in LIBX when a compare value of XYZ (starting in position 1) is matched in the routing data. The program runs in storage pool 1, and there is no maximum on the number of active routing steps allowed.

Error messages for ADDRTGE

*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.

ADDSCHIDX (Add Search Index Entry) Command Description

ADDSCHIDX Command syntax diagram

Purpose

The Add Search Index Entry (ADDSCHIDX) command is used to load help text topics into a search index.

A search index refers to help text from one or more panel groups. A panel group contains help text, which the user can access from display panels by pressing the Help key, or through the information search function using the Start Search Index (STRSCHIDX) command.

The sequence in which panel groups are loaded into a search index controls the sequence in which topic entries are presented when an information search is requested. The topics (ISCH tag entries) from the first-loaded panel group are presented first.

Restrictions:

1. The user must have *CHANGE authority for the search index that is being loaded and *USE authority for the panel group.
2. Only user-created panel groups can be added to user-created search indexes and only IBM-supplied panel groups can be added to IBM-supplied search indexes.
3. Panel group names must be unique within a search index.

Required Parameters

SCHIDX

Specifies the qualified name of the search index into which entries are loaded.

The name of the search index 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.

search-index-name: Specify the name of the search index.

PNLGRP

Specifies the qualified name of the panel group that contains the help text for which entries are to be loaded into the search index.

The search index object contains the name and library of the panel group. When help text is displayed for a search index, the name and library of the panel group that is contained in the search index object is used to find the panel group.

When *LIBL is used to qualify the panel group name, *LIBL is saved in the search index object. When the panel group name is qualified with either a library name or *CURLIB, the name of the library containing the panel group is saved in the search index object.

The names of panel groups added to the search index must be unique.

The name of the panel group 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.

panel-group-name: Specify the name of the panel group.

Example for ADDSCHIDX

```
ADDSCHIDX SCHIDX(ACCOUNTING) PNLGRP(PAYROLL)
```

This command adds panel group PAYROLL to search index ACCOUNTING. Both the panel group and the search index must exist in the library list.

Error messages for ADDSCHIDX

*ESCAPE Messages

CPF6E07

Panel group cannot be added to search index.

CPF6E08

Panel group cannot be added to search index.

CPF6E09

Panel group does not contain any synonyms or root words.

CPF6E12

Panel group not added to search index.

CPF6E47

Panel group &1 cannot be added to search index &3

CPF6E48

Panel group &1 cannot be added to search index &3

CPF6E49

Panel group &1 cannot be added to search index &3

CPF6E61

Panel group &1 cannot be added to search index &3.

CPF6E62

Panel group &1 already exists in search index &3.

CPF6E63

Error occurred while trying to recover from another error.

ADDSVRAUTE (Add Server Authentication Entry) Command

Description

ADDSVRAUTE Command syntax diagram

Purpose

The Add Server Authentication Entry (ADDSVRAUTE) command adds server authentication information for use by application requesters in connecting to application servers.

When using the ADDSVRAUTE command to add a server authorization entry for a Distributed Relational Database Architecture (DRDA) application that uses TCP/IP, make sure that the server name is entered in upper case.

Restriction: You must have *SECADM special authority, and *OBJMGT and *USE authorities to the user profile to which the server authentication entry is being added, or else be signed on under that user profile, to run this command.

Required Parameters

USRPRF

Specifies the user profile for which the server authentication entry will be added.

***CURRENT:** Specifies that the server authentication entry will be added for the current user.

user-profile-name: Specify the name of the user for which to add the server authentication entry.

SERVER

Specifies the name of the application server for which the entry is being added.

'server-name': Specify the name for the particular application server for which the entry is being added. Specify no more than 200 characters.

Optional Parameters

USRID

Specifies the user name for which requests will be made to the application server.

***USRPRF:** Specifies that the name specified in the user profile parameter will be the user ID specified on connection requests to the server.

'user-name': Specify the user ID to be used on connection requests. Specify no more than 1000 characters.

PASSWORD

Specifies the password to be used to authenticate the user when the client attempts to connect to the server.

Note:

If the retain server security data (QRETSVRSEC) system value is set to 0 (do not retain data), then the password will not be saved in the entry.

***NONE:** Specifies there is to be no password supplied on the connect request.

'password': Specify the password associated with the user ID. Specify no more than 696 characters.

Examples for ADDSVRAUTE

Example 1: Adding a default remote user ID and password for the current user

```
ADDSVRAUTE USRPRF(*CURRENT)
  SERVER(*ANY)
  USRID('JOHN') PASSWORD('XU53W4')
```

This command adds a server authentication entry for the currently signed on user specifying that for connection requests to any server for which there is no specific authentication entry, a remote user ID of JOHN and a password of XU53W4 is to be used.

Example 2: Adding an entry for another user for a specific server

```
ADDSVRAUTE USRPRF('SUSAN')
  SERVER('MPLS_RDB') USRID('SUSIE')
  PASSWORD('S23084')
```

This command adds an entry such that when a user is signed on to the local system under the user profile of SUSAN and attempts to connect to the server named MPLS_RDB, the user ID and password accompanying the connection request will be SUSIE and S23084.

Error messages for ADDSVRAUTE

*ESCAPE Messages

CPF2204

User profile &1 not found.

CPF2213

Not able to allocate user profile &1.

CPF2222

Storage limit is greater than specified for user profile &1.

CPF224F

Server authentication entry already exists.

CPF225F

Not all information stored.

CPF226C

Not authorized to perform function.

ADDSRVTBLE (Add Service Table Entry) Command Description

ADDSRVTBLE Command syntax diagram

Purpose

The Add Service Table Entry (ADDSRVTBLE) command is used to add a service entry to the service table. You can use the service table to manage the mapping of network services to ports and to record the protocols that the services use.

The service table is shipped with some standard port assignments. Values for common functions supported by Transmission Control Protocol/Internet Protocol (TCP/IP) are available to the Internet community in the assigned numbers **RFC** (Request for Comments) document, a formal specification of proposals and standards for a portion of TCP/IP.

Restriction: You must have system configuration (*IOSYSCFG) special authority to use this command.

Required Parameters

SERVICE

Specifies the name of the network service to be added to the table. A service can be added to the table more than once. Each service must be uniquely identified by a combination of the port number and the protocol name parameters.

PORT Specifies the port number assigned to the service. This value must be greater than zero.

PROTOCOL

Specifies the name of the protocol that the service uses. You can specify a maximum of 32 characters for the protocol name. No checking is done to ensure that the protocol exists.

Optional Parameters

ALIAS Specifies the alternate name for the network service. You can specify a maximum of four aliases. No checking is done to ensure that an alias is unique.

***NONE:** The service has no alternate name.

alias: Specify an alternate service name.

TEXT Specifies the text that briefly describes the network service entry. More information is in Commonly used parameters.

***BLANK:** Text is not specified.

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

Example for ADDSRVTBLE

```
ADDSRVTBLE SERVICE(FTP) PORT(21) PROTOCOL(TCP)
```

This command adds a service entry to the service table for the FTP network service. The service uses port 21 and the TCP protocol.

Error messages for ADDSRVTBLE

***ESCAPE Messages**

TCP290A

Service entry already exists in table. Entry was not added.

TCP2914

Service entry contains characters that are not valid. Entry was not added.

ADDSOCE (Add Sphere of Control Entry) Command Description

ADDSOCE Command syntax diagram

Purpose

The Add Sphere of Control Entry (ADDSOCE) command allows a CL user or program to add control points to the Alert Sphere of Control.

Required Parameter

ENTRY

Specifies the systems to add to the sphere of control. The systems are specified as a list of two elements that includes the network ID and control point name.

Element 1: Network ID of the System

***NETATR:** The NETID network attribute is used as the value of the network ID being added to the alert sphere of control.

network-ID: Specify the network ID of system being added to the alert sphere of control.

Element 2: Control Point Name of the System

control-point-name: Specify the control point name of the system being added to the alert sphere of control.

Example for ADDSOCE

```
ADDSOCE ENTRY((*NETATR RCHSTR1) (*NETATR RCHSTR2))
```

This command adds two systems (RCHSTR1 and RCHSTR2) to the alert sphere of control.

Error messages for ADDSOCE

*ESCAPE Messages

None

ADDTAPCTG (Add Tape Cartridge) Command Description

ADDTAPCTG Command syntax diagram

Purpose

The Add Tape Cartridge (ADDTAPCTG) command adds the specified cartridge identifiers to a usable category. Cartridges are placed in the insert category when they are placed in the library device and must be added to a usable category before they can be used by a tape device.

The cartridge identifier must be unique within a library device. If a duplicate cartridge identifier exists in a library device, both cartridges are unusable until one is physically removed from the library device.

Required Parameters

DEV Specifies the name of the library device to be used. The device name must have been created previously on the system using the Create Device Media Library (CRTDEVMLB) command.

CTG Specifies a maximum of 40 cartridge identifiers that are currently in the insert category and that are to be added to the category specified. Each cartridge identifier can be a maximum of 6 characters.

Note: The cartridge identifier should be the same as the external identifier if the library device has a bar code scanner to read external identifiers.

Optional Parameters

CGY Specifies the category to which the tape cartridge is added. The cartridge is moved to a slot in the library device, unless the cartridge is added to the convenience (CNV) category.

Element 1: Category Name

***NOSHARE:** The cartridge identifier specified cannot be shared with other systems that are attached to the same device.

***IPL:** The cartridge identifier specified can be used for an alternate initial program load (IPL) of a system. The management of the cartridges in this category must be done by the user.

Attention

When using this category, you must ensure that the cartridges are the proper ones to be used for the alternate IPL. Conflicts can arise if high-end and low-end systems are attached to the same library device.

***NL:** The cartridge is used as a non-labeled tape.

***CNV:** The cartridge identifier specified is added to the special convenience category. It is not moved to a slot in the library device. When the cartridge is unloaded from a device, it is removed (exported) to the convenience station.

category-name: Specify the name of a user-defined category. This category name must have been created previously with the Create Tape Category (CRTTAPCGY) command.

Element 2: Category System

This element identifies the system to which the category belongs. The system name is obtained from the pending system name field of a Display Network Attributes (DSPNETA) command. If there is no pending system name, the current system name attribute is used.

Attention

If a system name is changed, the tape cartridges in library devices that have the attribute of the system name before it was changed are no longer valid.

***CURRENT:** The category belongs to the system currently running the command.

system-name: Specify the name of the system to which the category belongs.

Single Value

***SHARE400:** The cartridge identifier specified can be shared with other iSeries 400 computers that are attached to the same device.

CHKVOL

Specifies whether the logical volume identifier is forced to be identical to the external identifier if the library device has a bar code scanner to read the bar code identifier.

***YES:** The tape cartridge is verified for the correct logical volume identifier by reading the volume label existing on the tape cartridge.

***NO:** The tape cartridge is not verified for the correct logical volume identifier. If a cartridge is added for which the external identifier does not match the logical volume identifier, the cartridge is valid for read-only operations. Output operations to the tape cartridge are not allowed unless the logical volume identifier is initialized to match the external identifier. If the tape is a non-labeled tape, this match is not enforced because there is no logical volume identifier.

Examples for ADDTAPCTG

Example 1: Adding a Single Cartridge to the *SHARE400 Category

```
ADDTAPCTG  DEV(LIB01)  CTG(VOL4)
           CGY(*SHARE400)  CHKVOL(*NO)
```

This command adds the cartridge identifier VOL4 to the usable category *SHARE400. The logical volume identifier in the volume labels of VOL4 are not verified.

Example 2: Adding Multiple Cartridges to the *NOSHARE Category

```
ADDTAPCTG  DEV(LIB01)  CTG(VOL1 VOL2 VOL3)
           CGY(*NOSHARE)  CHKVOL(*YES)
```

This command adds the cartridge identifiers VOL1, VOL2, and VOL3 to the usable category *NOSHARE. The logical volume identifiers on the tape cartridges are verified when the command is run.

Error messages for ADDTAPCTG

***ESCAPE Messages**

CPF67AB

&6 cartridges not added

CPF67A6

Category does not exist

CPF67D2

Cartridge does not exist

CPF67E4

Library device function not successful

CPF67EA

Function not successful

CPD67EB

Cartridge &2 not in *INSERT category

CPF67EC

Library device description &1 does not exist

CPF67ED

Library device &1 not available

CPF67F5

Duplicate cartridge ID found

CPF6708

Command ended due to error.

CPF6718

Cannot allocate device &1.

CPF6745

Device &1 not a media library device.

CPF9814

Device &1 not found.

CPF9825

Not authorized to device &1.

ADDTCPHTE (Add TCP/IP Host Table Entry) Command Description

ADDTCPHTE Command syntax diagram

Purpose

The Add TCP/IP Host Table Entry (ADDTCPHTE) command adds an internet address and its associated host names along with an optional text description field to the local host table. For each entry, the host table is defined to allow one internet address, up to 4 host names, and a text description field.

If an internet address already exists in the host table that matches the internet address specified in the command, an escape message is sent to the user and the duplicate internet address is not added.

If a remote name server is being used by your iSeries 400 for resolving a host name or an internet address, the choice to first search the remote name server or the local host table depends on how the searched-first value was configured on the configuration panel of the remote name server. To change the remote name server or the searched-first value, enter the Configure TCP/IP (CFGTCP) command and select option 13.

The TCP/IP host table is shipped with the loopback entry. This entry has an internet address of 127.0.0.1 and two host names; LOOPBACK and LOCALHOST. The LOOPBACK host name can only be associated with an internet address that has a first-byte value equal to 127.

Related APPC over TCP/IP Information:

APPC over TCP/IP (part of the AnyNet/400* function) uses the host name to map location names to internet addresses. The host name must be in the form:

```
location.netid.SNA.IBM.COM
```

Where *location* is the remote location the program is opening to, and *netid* is the network identifier for this connection. *SNA.IBM.COM* is the qualifier that designates this as the APPC over TCP/IP domain.

Location names support characters that cannot be present in host names (for example: \$ (dollar), @ (at sign), and # (number sign)). Therefore, the APPC application can open only to locations that fulfill the TCP/IP host name syntax. This limits location names used for APPC over TCP/IP to the characters A-Z (uppercase and lowercase) and 0-9.

Restriction: You must have *IOSYSCFG special authority to use this command.

Required Parameters

INTNETADR

Specifies the internet address that the host names and text descriptions are associated with. 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.

HOSTNAME

Specifies the host names corresponding to the internet address. The host name can be either the short form or the full domain version of the name. A common practice is to define one short name that is unique within your local network and to also define the full domain version of the host name that is unique within the internet. Specify from 1 to 4 different host names to be associated with the internet address.

A domain name or a host name can be a text string having 1 to 255 characters. Domain names consist of one or more labels separated by periods. Each label can contain up to 63 characters. The first character of each label must be an alphabetic character or a digit. The last character of each label must be an alphabetic character, a digit, or a period. The following characters are allowed in domain names:

- Alphabetical characters A through Z
- Digits 0 through 9
- Underscore (_)
- Minus sign (-)
- Period (.). Periods are allowed only when they separate labels of the domain style name or as the last character in the domain name. (Refer to RFC 1034.) A domain name cannot have two consecutive periods.

Note:

These characters are part of the Syntactic Character Set (character set number 640). This character set is also commonly referred to as invariant.

Other domain name and host name conventions include the following:

- Uppercase and lowercase characters are allowed, but no significance is attached to the case. The host name (HOSTNAME) may be converted to uppercase depending on the combination of characters and digits. If the HOSTNAME is enclosed in apostrophes ('), the case is maintained as entered.
- The host name returned when searching the host table for an internet address is the first host name associated with the internet address. For example, if the address 9.130.38.187 is defined in the host table with names ROCHESTER, JOHN, and RCHAS100, the name ROCHESTER would be returned. The other two host names would not be used in this type of search. However, these host names would be used when searching the host table to find the internet address associated with the names JOHN and RCHAS100.
- Try to limit your domain name labels to 12 characters. Shorter labels are easier to remember.
- It is a common practice to use hierarchical names that allow predictable extensions for change and growth. Domain names normally reflect the delegation of authority or hierarchy used to assign them.

For example, the name SYS1.MFG.ABC.COM can be broken down into the following:

COM All commercial networks.

ABC.COM

All systems in the ABC company's commercial network.

MFG.ABC.COM

All manufacturing systems in the ABC company's commercial network.

SYS1.MFG.ABC.COM

A host named SYS1 in the manufacturing area of the company's commercial network.

The COM designation is one of several domain names used by convention when connecting to the Internet. Some of the other domain names that follow this convention are:

COM Commercial organizations

EDU Educational institutions

GOV Government institutions

MIL Military groups

NET Major network support centers

ORG Organizations other than those listed previously

ARPA Temporary ARPANET domain

Country or region code

Countries other than USA

host-name: Specify a host name to be associated with the specified internet address. When running APPC over TCP/IP, *name* is in the form:

location.netid.SNA.IBM.COM

The default if a host name is not specified is blanks. At least one host name must be specified. An IP address cannot be a host name.

Optional Parameter

TEXT Specifies a comment associated with this host table entry.

Note:

If the host table will be copied to a system using a different code page than the system it was created on, it is suggested that you avoid using certain characters in a comment. Host table entry comments will be more portable if they are limited to characters in the Syntactic Character Set (invariant).

***BLANK:** The text-description field for this host table entry is to contain blanks.

'description': Specify a text-description field to be associated with the specified internet address. Comments can contain a maximum of 64 characters.

Examples for ADDTCPHTE

Example 1: Adding a Short Host Name

```
ADDTCPHTE  INTNETADR('132.28.71.5')
           HOSTNAME(AS400ETH)
           TEXT('iSeries 400 on Ethernet subnet')
```

This command associates the host name AS400ETH with the internet address of 132.28.71.5. The text 'iSeries 400 on Ethernet subnet' is saved as the descriptive comment for this host table entry.

Example 2: Adding Two Host Names

```
ADDTCPHTE  INTNETADR('9.5.42.6')
           HOSTNAME((AS400ETH.SALES.ABC.COM)
                   ('as400eth.sales.abc'))
           TEXT('Entry verified on 1 April 1994 by J. Jones')
```

This command associates the host names AS400ETH.SALES.ABC.COM and AS400ETH.SALES.ABC with the internet address of 9.5.42.6. Because no significance is attached to a case, a match is found on host name AS400ETH.SALES.ABC.COM or as400eth.sales.abc. The text 'Entry verified on 1 April 1994 by J. Jones' is saved as the descriptive comment for this host table entry.

Error messages for ADDTCPHTE

*ESCAPE Messages

TCP1901

Internet address &1 not valid.

TCP1902

Internet address &1 not valid.

TCP1903

Specified host name not valid.

TCP1904

Duplicate internet address &1 found in host table.

TCP1908

Internet address &1 not valid.

TCP1910

LOOPBACK internet address &1 not valid.

TCP1929

Host table not available.

TCP9999

Internal system error in program &1.

ADDTCPIFC (Add TCP/IP Interface) Command Description

ADDTCPIFC Command syntax diagram

Purpose

The Add TCP/IP Interface (ADDTCPIFC) command is used to define a new interface to the Transmission Control Protocol/Internet Protocol (TCP/IP) configuration. The interfaces defined by the ADDTCPIFC command are logical interfaces. They are not physical interfaces. Each interface is associated with a line description. The line description is the physical connection from the iSeries 400 to the TCP/IP network.

The iSeries 400 TCP/IP implementation supports *multihoming*. This allows you to specify either a single interface or multiple interfaces per line description. You can have your iSeries 400 appear as any one or combination of the following:

- A single host on a network over a communications line
- Multiple hosts on the same network over the same communications line
- Multiple hosts on different networks over the same communications line
- Multiple hosts on the same network over multiple communications lines
- Multiple hosts on different networks over multiple communications lines

Notes:

1. Up to 128 logical interfaces can be defined and started per communications line.
2. In SNMP, the interface is a physical interface. The physical interface relates directly to an input/output processor (IOP).
3. The interface table is shipped with a default interface of 127.0.0.1. The line description value associated with the 127.0.0.1 interface is *LOOPBACK. The host table is also shipped with an entry that has an internet address of 127.0.0.1 and host names of LOOPBACK and LOCALHOST.

Attention

Before attempting to start an X.25 interface, ensure that the remote system information (RSI) for non-DDN X.25 interfaces that use a permanent virtual circuit (PVC) is configured. Use the Add TCP/IP Remote System Information (ADDTCPRSI) command to do this. Incoming data from a remote system on the X.25 network is not processed unless an RSI entry for the PVC is configured on the X.25 interface before the interface is started.

Restriction: You must have *IOSYSCFG special authority to use this command.

Required Parameters

INTNETADR

Specifies an internet address that the local system responds to on this interface. An interface is associated with a line description. 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.

LIND Specifies the name of the line description associated with the new interface. The line description must be defined before the TCP/IP interface can be added. The following conditions are based on the interface type that the user defines:

Token-ring

The name must be previously defined on the Create Line Description (Token-Ring Network) (CRTLINTRN) command.

X.25 The name must be previously defined on the Create Line Description (X.25) (CRTLINX25) command.

Ethernet

The name must be previously defined on the Create Line Description (Ethernet) (CRTLINETH) command.

DDI The name must be previously defined on the Create Line Description (DDI Network) (CRTLINDDI) command.

Frame relay

The name must be previously defined on the Create Line Description (Frame Relay Network) (CRTLINFR) command.

Wireless

The name must be previously defined on the Create Line Description (Wireless Network) (CRTLINWLS) command.

Twinax (TDLC)

The name must be previously defined on the Create Line Description (CRTLINTDLC) command.

TCP/IP can also be used on certain line descriptions attached to these network interfaces (NWI):

- An ISDN NWI using an X.25 line description.
 - The ISDN NWI is created using the Create Network Interface ISDN (CRTNWIISDN) command.
 - The X.25 line is created using the Create Line X.25 (CRTLINX25) command and attached to the ISDN NWI by specifying the NWI, NWICHLTYPE, NWICHLNBR, and SWTNWILST parameters.
- A frame relay NWI using a frame relay, token ring, Ethernet, or DDI line description.
 - The frame relay NWI is created using the Create Network Interface Frame Relay Network (CRTNWIFR) command.
 - The line description is created using the appropriate Create Line command and attached to the frame relay NWI by specifying the NWI and NWIDLCL parameters.

***LOOPBACK:** This special value is used if and only if the first octet of the interface internet address is 127. This value indicates that the interface being added by this ADDTCPIFC command is the loopback or LOCALHOST interface. Because processing associated with loopback does not extend to a physical line, there is no line description associated with a loopback address.

***VIRTUALIP:** This special value is used if you are adding a 'circuitless' interface. This means that this interface has a real IP address but is NOT tied to any physical hardware. Interfaces of this type are useful to identify the address of the iSeries 400 to remote systems over point-to-point links (PPP, SLIP, Frame Relay).

***OPC:** This special value is used if you are adding an OptiConnect interface over TCP/IP. This interface is attached to the optical bus (OptiConnect).

line-description: Specify the line description to be used for this interface.

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.

***HOST:** The subnetmask value used will 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 subnet 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. 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 advised that the subnet bits be contiguous and located in the most significant bits of the host address.

Optional Parameters

LCLIFC

The local IP interface that the internet address defined in INTNETADR will be associated with.

The associated local interface is used to allow for transparent subnetting or unnumbered networks on the iSeries 400. Any local interface may be used for LCLIFC, except for interfaces defined for the X.25 or PPP linetypes.

***NONE:** No associated local interface is used.

local-interface: Specify an associated local interface for the interface being added. Note that the specified associated local interface must already exist.

TOS Specifies the type of service to be used. The type of service defines how the internet hosts and routers should make trade-offs between throughput, delay, reliability, and cost.

***NORMAL:** Normal service is used for delivery of data.

***MINDELAY:** Minimize delay means that prompt delivery is important for data on this connection.

***MAXTHRPUT:** Maximize throughput means that a high data rate is important for data on this connection.

***MAXRLB:** Maximize reliability means that a higher level of effort to ensure delivery is important for data on this connection.

***MINCOST:** Minimize monetary cost means that lower cost is important for data on this connection.

MTU Specifies the maximum size (in bytes) of IP datagrams that can be transmitted through this interface. A datagram is a basic unit of information passed over an internet network. The minimum size of any maximum transmission unit value is 576 bytes.

***LIND:** The MTU is determined by the information specified in the line description. If *LIND is specified, the MTU will be equal to the largest amount of data that can be transmitted on the line. If the LIND parameter specifies *LOOPBACK, *VIRTUALIP, or *OPC, then the MTU value will be:

***LOOPBACK**

576

***VIRTUALIP**

576

***OPC** 32768

maximum-transmission-unit: Specify a value for the maximum transmission unit in bytes. The maximum MTU that can be specified for this interface depends on the type of physical connection to the network. The following table lists the maximum MTU values that can be specified based on the line type:

X.25 4096

Token ring (4 meg)

4060

Token ring (16 meg)

16388

Ethernet 802.3

1492

Ethernet Version 2

1500

DDI 4352

Frame relay

8177

Wireless 802.3

1492

Wireless Version 2

1500

Twinax (TDLC)

4105

Notes:

1. The actual MTU value used for an interface is resolved during interface activation. This value is the minimum of either the specified MTU value for the interface or the largest amount of data that can be transmitted on the line.
2. It is suggested (not required) that the same MTU values be used for all interfaces on the same network. However, all interfaces must have an MTU that does not exceed the value used when *LIND is specified for the interface MTU.
3. To view the MTU value actually used for an interface, do the following:
 - a. Use the ADDTCPIFC command to add the interface.
 - b. Use the Start TCP/IP Interface (STRTCPIFC) command to activate the interface.

- c. Use the Work with TCP/IP Status (WRKTC PSTS or NETSTAT) command to view the actual MTU value of the interface in bytes.

AUTOSTART

Specifies whether the interface is automatically started when the TCP/IP stack is activated with the Start TCP/IP (STRTCP) command.

***YES:** The interface is automatically started when TCP/IP is started.

***NO:** The interface is not started when TCP/IP is started.

Note:

The Start TCP/IP Interface (STRTCP IFC) command can be used to start an interface any time after TCP/IP has been activated.

PVCLGLCHLI

Specifies the permanent virtual circuit (PVC) logical channel identifiers that can be established on an X.25 interface by the TCP/IP protocol stack. Up to 64 unique channel identifiers may be specified. These logical channel identifiers must be included in the X.25 line description that is specified with the LIND parameter on the ADDTCPIFC command.

With this parameter you can share the line with other communications software, such as Systems Network Architecture (SNA). It prevents the TCP/IP protocol stack from monopolizing the PVCs defined for the line.

Notes:

1. This parameter is valid only for an interface defined on a X.25 line description.
2. PVCs cannot be used in a DDN network.
3. When specifying PVCs for an X.25 interface, all interfaces on the same X.25 network should have the same set of PVC logical channel identifiers specified. This is especially important if one or more remote system information (RSI) entries will use a PVC to connect to the RSI entry's remote system on the X.25 network.
4. If the RSI entries are defined such that two or more remote internet addresses can be reached across the same PVC, that PVC is shared.
5. The sum of the maximum switched virtual circuits (MAXSVC) and the number of PVCs cannot exceed 64.

logical-channel-identifier: Specify the PVC logical channel identifier value. The value may be from 001 to FFF. Up to 64 PVC logical channel identifiers can be specified.

IDLVCTTIMO

Specifies the duration (in seconds) that TCP/IP waits before clearing an idle virtual circuit established on an X.25 interface. Clearing an idle virtual circuit frees resources on the network. TCP/IP automatically reestablishes virtual circuits when required to send or receive data. Virtual circuits are transparent to a TCP/IP client and have no noticeable effect on TCP connections.

Note:

This parameter is valid only for switched virtual circuits (SVCs) on an interface defined on an X.25 line description. It is not valid for permanent virtual circuits (PVCs).

60: The idle virtual circuit timeout is 60 seconds.

number-of-seconds: Specify the idle virtual circuit timeout. Valid values range from 1 through 600 seconds.

MAXSVC

Specifies the maximum number of concurrent switched virtual circuits (SVC) that can be established on an X.25 interface by the TCP/IP protocol stack.

With this parameter you can share the line with other communications software such as Systems Network Architecture (SNA). It prevents the TCP/IP protocol stack from monopolizing the SVCs defined for the line. This parameter is valid only for an interface defined on an X.25 line description.

Note: The sum of the maximum switched virtual circuits (MAXSVC) and the number of PVCs cannot exceed 64.

64: If 64 is specified, the number of SVCs that are configured is the sum of the number of *SVCIN, *SVCOUT and *SVCBOTH SVCs defined for the line description (LIND) that is used by this interface. This is the maximum number of SVCs that can be authorized for processing by the TCP/IP protocol stack.

X.25-maximum-virtual-circuits: Specify the number of SVCs that TCP/IP protocol stack can use simultaneously. The valid values range from 0 through 64.

DDN Specifies whether the X.25 interface is connected to the Defense Data Network (DDN). The DDN network is a special type of X.25 network used by TCP/IP customers with special security needs.

Note: This parameter is valid only for switched virtual circuits (SVCs) on an interface defined on an X.25 line description. It is not valid for permanent virtual circuits (PVCs).

Attention:

If you specify multiple interfaces to the same X.25 network, the DDN value should be equal for all of those interfaces. This is not enforced by the ADDTCPIFC or CHGTCPIFC commands.

If the X.25 network is on the DDN network, do not define the remote system information for any of the remote systems on the network. The remote system information for the DDN X.25 network is determined from the destination IP address.

***NO:** The X.25 interface is not connected to the Defense Data Network.

***YES:** The X.25 interface is connected to the Defense Data Network.

BITSEQ

Specifies the order, most or least significant bit first, in which the Address Resolution Protocol (ARP) places the bits in the hardware address. This parameter is valid only for a token-ring local area network (TRLAN) line.

Note: All interfaces defined to a single token-ring line must have the same BITSEQ value. This is checked by the ADDTCPIFC code to ensure consistent values.

***MSB:** The most significant bit is placed first.

***LSB:** The least significant bit is placed first.

Examples for ADDTCPIFC

Example 1: Adding a Non-AUTOSTART Interface

```
ADDTCPIFC  INTNETADR('130.14.3.5')
           LIND(COTTAGELAN) AUTOSTART(*NO)
           SUBNETMASK('255.255.255.0')
```

This command assumes that an Ethernet line has been created named COTTAGELAN using the CRTLINETH command. This command adds the interface 130.14.3.5 to the TCP/IP configuration. This interface uses the line description named COTTAGELAN. It is not automatically started when the STRTCP command is run. This interface must be started using the Start TCP/IP Interface (STRTCPIFC) command. The STRTCPIFC can be issued either directly from a command line or by using option 9 from either of the following lists:

- The Work with TCP/IP Interface Status list. Use menu option 1 from the menu displayed when the Work With TCP/IP Status (WRKTCPSTS) command is issued to display this list.
- The Work with TCP/IP Interfaces list. Use menu option 1 from the menu displayed when the Configure TCP/IP (CFGTCP) command is issued to display this list.

Example 2: Adding an AUTOSTART Interface

```
ADDTCPIFC  INTNETADR('8.77.0.21')
           LIND(COTTAGEX25) IDLVCTTIMO(45)
           MAXSVC(15) DDN(*YES)
           SUBNETMASK('255.255.255.0')
```

This command assumes that an X.25 line has been created named COTTAGEX25 using the CRTLINX25 command. This command adds interface 8.77.0.21 to the TCP/IP configuration. This interface uses the line description named COTTAGEX25. When TCP/IP is started using the Start TCP/IP (STRTCP) command, the interface is automatically started. The idle virtual circuit timeout is 45 seconds. The maximum number of concurrent SVCs allowed to be used by TCP/IP on this interface is 15. This interface is connected to the Defense Data Network. You do not need to define any remote system information (RSI) entries for this X.25 network because it is a DDN network.

Example 3: Adding an Interface for a twinax line that is using an associated local interface

```
ADDTCPIFC  INTNETADR('199.1.1.99')
           LIND(TDLCLINE) SUBNETMASK(255.255.255.0)
           LCLIFC('199.1.1.1')
```

This command will add a TCP/IP interface for the twinax line named TDLCLINE. This interface will be associated with local interface 199.1.1.1. This means that the devices attached to twinax line 199.1.1.99 can take advantage of 'appearing' to be on the same network as the local 199.1.1.1 interface (transparent subnetting). No special routing is required to ensure packets from the twinax connected hosts can travel to the local 199.1.1.0 network. Also, hosts on the 199.1.1.0 network can also reach the twinax hosts without any additional routing on the host systems.

Error messages for ADDTCPIFC

*ESCAPE Messages

TCP1D03

&1 member record length not correct.

TCP1D04

Error occurred processing member &1 of &2/&3.

TCP1901

Internet address &1 not valid.

TCP1902

Internet address &1 not valid.

TCP1908

Internet address &1 not valid.

TCP8050

*IOSYSCFG authority required to use &1.

TCP9999

Internal system error in program &1.

ADDTCPPORT (Add TCP/IP Port Restriction) Command Description

ADDTCPPORT Command syntax diagram

Purpose

The Add TCP/IP Port Restriction (ADDTCPPORT) command is used to restrict a port or range of ports in the TCP/IP configuration to a particular user profile. A port can be restricted for use by multiple user profiles. The addition of the user profile takes effect immediately. Any user profiles currently using a port that will not have access to that port after the use of this command are allowed to finish processing.

The default authorization for TCP/IP ports is to allow any user profile access to any port. If it is unnecessary to restrict a port to a user profile or a group of user profiles, the system administrator does not need to use this command.

Once an application running under a user profile has obtained the use of a restricted port, TCP/IP does not prohibit that application from passing its rights to another job that may be running under another user profile. The new user profile for the port is not checked against the list of user profiles having exclusive rights to that port. That is because the allocation of the port occurred under the user profile that had exclusive rights to that port.

The check for restricted use of the port occurs only on the BIND operation to the port. If other user profiles are currently using a port and an administrator wants to restrict a port or range of ports, the administrator may need to end all current TCP connections or user datagram protocol (UDP) sockets using that port. To do this, enter NETSTAT, select option 3, then select all of the connections or listening sockets that are using the port that you want to restrict. Enter an option 4 (ENDTCPCNN) for each.

There are two independent sets of ports. One set is for TCP processing and the other is for UDP processing. They are completely independent sets of ports and have no relationship to one another.

Restriction:

You must have *IOSYSCFG special authority to use this command.

Required Parameters

PORT Specifies the port number or range of port numbers identifying the port or ports that are being restricted. Valid values range from 1 through 65535. However, some of the ports in the range 1 through 1023 are used by system-supplied TCP/IP applications. If the user specifies one of these ports, it can affect the operation of those applications.

See the assigned numbers RFC for the definition of port numbers currently used by TCP/IP applications.

Element 1: Lower Port Value

lower-value: Specify the port value or the lower port value in a range that you want restricted.

Element 2: Upper Port Value

***ONLY:** The port value specified in the lower port value is the only port value that is restricted.

upper-value: Specify the upper port value in a range that you want restricted.

PROTOCOL

Specifies the transport protocol associated with the port or range of ports being restricted. Each transport protocol has its own distinct set of ports in the range of 1 to 65535.

***UDP:** The port is a User Datagram Protocol (UDP) transport protocol port.

***TCP:** The port is a Transmission Control Protocol (TCP) transport protocol port.

USRPRF

Specifies the name of the user profile to which the port or range of ports is being restricted. Only jobs running under this profile or group profile may use the port or range of ports specified.

A user profile that is used as a group profile may be specified in the user profile field of this command. If users have a group profile specified in their user profile and that group profile was specified for a particular port or range of ports, then these users are given access to the specified port or range of ports. However, adopted authorities are not used when deciding whether this port is restricted or not. Each user profile or group profile that wants to use a port or range of ports must be explicitly added.

When a socket application issues the bind() system call, or when a TCP/UDP PASCAL API application issues a TcpOpen, TcpWaitOpen, or UdpOpen, the user profile that the job is running under is checked against the list of user profiles that are associated with the specified port. If there is not a match on that user profile, then a check is made to determine if this user profile is part of a group and that the group profile is in the list of user profiles that are associated with the specified port.

For example, there are two user profiles, USER_1 and USER_2. USER_2 is specified as a member of a group associated with USER_1. If the TCP port 1015 has a user profile list consisting of USER_1, then a bind() by USER_2 will work because USER_2 is a part of the group profile USER_1.

user-profile-name: Specify the user profile that the port or range of ports is restricted to.

Examples for ADDTCPPORT

Example 1: Adding a Single User Profile

```
ADDTCPPORT PORT(7059) PROTOCOL(*UDP)
  USRPRF(TCPUSER)
```

This command adds the user profile TCPUSER to the set of user profiles that are allowed to bind UDP port 7059. User profiles that have not been added to this set or are not in a group profile that has been added will not be allowed to use UDP port 7059.

Example 2: Adding Multiple User Profiles

```
(1) ADDTCPPORT PORT(1590) PROTOCOL(*TCP)
  USRPRF(USER1)

(2) ADDTCPPORT PORT(1590) PROTOCOL(*TCP)
  USRPRF(USER2)
```

These commands show that a port can be restricted for use by multiple user profiles. User profiles USER1 and USER2 are the only users that are allowed to bind to TCP port 1590.

Example 3: Adding a Single User Profile to a Range of Ports

```
ADDTCPPORT PORT(1591 1600) PROTOCOL(*TCP) USRPRF(USER3)
```

This command adds the user profile USER3 to the set of user profiles that are allowed to bind TCP ports 1591 through 1600.

Error messages for ADDTCPPORT

***ESCAPE Messages**

TCP1D03

&1 member record length not correct.

TCP1D04

Error occurred processing member &1 of &2/&3.

TCP26E2

User profile &1 damaged.

TCP26E4

Port restriction action successful, but TCP/IP errors occurred.

TCP26FC

Upper port value must be *ONLY.

TCP26F1

Range of ports not valid.

TCP2677

Port restriction not added.

TCP2679

port entry was added successfully but errors occurred.

TCP2680

Duplicate port restriction found.

TCP8050

*IOSYSCFG authority required to use &1.

TCP9503

File &3 in library &2 not available.

TCP9509

Line &1 not found.

TCP9517

Duplicate port entry found.

TCP9526

User profile &1 not found.

TCP9999

Internal system error in program &1.

ADDTCPRSI (Add TCP/IP Remote System Information) Command Description

ADDTCPRSI Command syntax diagram

Purpose

The Add TCP/IP Remote System Information (ADDTCPRSI) command is used to associate an internet address with an X.25 network address or a local permanent virtual circuit (PVC) logical channel identifier in the TCP/IP configuration.

When the user works with an X.25 public or private data network, the internet address and the network address of each remote system or local (PVC) logical channel identifier needs to be specified.

Attention:

1. Do not specify the X.25 network address for systems on the X.25 Defense Data Network (DDN). The X.25 DDN has a built-in conversion algorithm that converts an IP address to the remote DTE address. If you specify an X.25 network address for remote systems on an X.25 DDN, the DDN conversion algorithm is bypassed. In this case it is possible that you will not be able to connect to the requested host.
2. Before attempting to start an X.25 interface, ensure that the remote system information (RSI) for non-DDN X.25 interfaces that use a permanent virtual circuit (PVC) is configured. Use the Add TCP/IP Remote System Information (ADDTCPRSI) command to do this. Incoming data from a remote system on the X.25 network is not processed unless an RSI entry for the PVC is configured on the X.25 interface before the interface is started.
3. Attempts to change or remove a route or interface that is required to reach an existing RSI entry will fail.

Restriction:

You must have *IOSYSCFG special authority to use this command.

Note:

If specific values are entered for DFTPFSIZE and DFTWDWSIZE and the interfaces or routes are changed, conflicts could result. If *LIND is used, these values are adjusted accordingly if changes occur at the interface and route level.

Required Parameters

INTNETADR

Specifies the internet address of the remote system. 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.

NETADR

Specifies the X.25 network address or the DTE that is to be associated with the X.25 internet address. The user can specify a decimal number that is 1 through 17 digits in length.

Note:

If you specify a value for this parameter, the PVCLGLCHLI parameter value cannot be specified.

network-address: Specify the X.25 network address of a remote X.25 system.

PVCLGLCHLI

Specifies the local permanent virtual circuit (PVC) logical channel identifier that is used to establish an X.25 PVC interface to the specified remote internet address. One unique channel identifier may be specified. This unique channel identifier must have been previously identified in the ADDTCPIFC or CHGTCPIFC command that defined a TCP/IP X.25 interface. The TCP/IP X.25 PVC logical channel identifier is used to establish the circuit between this iSeries 400 TCP/IP X.25 interface and the host defined by the remote internet address. The logical channel identifier must also exist in the X.25 line description used for the TCP/IP X.25 interface.

Notes:

1. If this parameter's value is specified, the NETADR parameter value cannot be specified.
2. When specifying a PVC, consider which interface or set of interfaces this RSI entry could use to connect to the remote system. Each of the interfaces that could be used to reach this RSI entry's remote system must have the specified PVC logical channel ID configured as part of the interface.

logical-channel-identifier: Specify the PVC logical channel identifier value. The value may be from 001 to FFF. Only 1 PVC logical channel identifier can be specified.

Optional Parameters

RVSCRG

Specifies whether reverse charges are accepted or requested on an X.25 remote system basis.

***NONE:** Reverse charges are not accepted or requested.

***REQUEST:** Reverse charges are requested on outgoing call request packets. Reverse charges are not accepted on incoming call request packets.

***ACCEPT:** Reverse charges are accepted on incoming call request packets. Reverse charges are not requested on outgoing call request packets.

***BOTH:** Reverse charges are requested for outgoing call request packets and are accepted on incoming call request packets.

DFTPKTSIZE

Specifies the default packet size used by the X.25 network for transmission and reception. The values specified here should match the default values used by the X.25 network.

Element 1: Transmit Packet Size

***LIND:** The value specified in the line description associated with the X.25 interface used to reach the remote system is used as the default packet size.

transmit-packet-size: Specify a default packet size for transmission. The valid values for the packet size are 64, 128, 256, 512, 1024, 2048, and 4096.

Element 2: Receive Packet Size

***LIND:** The value specified in the line description associated with the X.25 interface used to reach the remote system is used as the default packet size.

***TRANSMIT:** The value specified as the packet size for transmission is used as the default for reception.

receive-packet-size: Specify a default packet size for reception. The valid values for the packet size are 64, 128, 256, 512, 1024, 2048, and 4096.

DFTWDWSIZE

Specifies the default packet window size for transmission to and reception from remote systems attached to the X.25 line.

Element 1: Transmit Window Size

***LIND:** The value specified in the line description associated with the X.25 interface used to reach the remote system is used as the default window size.

transmit-window-size: Specify the appropriate default window size. Valid values range from 1 through 7 for networks that use modulus 8 packet numbering. Valid values range from 1 through 15 for networks that use 128 packet numbering. The modulus value is specified on the X.25 line description.

Element 2: Receive Window Size

***LIND:** The value specified in the line description associated with the X.25 interface used to reach the remote system is used as the default window size.

***TRANSMIT:** The value specified as the default window size for transmission is used as the default for reception.

receive-window-size: Specify the appropriate default window size. Valid values range from 1 through 7 for networks that use modulus 8 packet numbering. Valid values range from 1 through 15 for networks that use 128 packet numbering. The modulus is specified on the X.25 line description.

Examples for ADDTCPRSI

Example 1: Adding RSI with NETADR

```
ADDTCPRSI INTNETADR('8.76.0.12')
          NETADR(4005)
```

This command allows the TCP/IP protocol stack to associate the internet address of 8.76.0.12 with the X.25 network address of 4005. Defaults are used for the remaining parameters.

Example 2: Adding RSI with PVCLGLCHLI

```
ADDTCPRSI INTNETADR('145.9.43.188')
          PVCLGLCHLI(231)
```

This command allows the TCP/IP protocol stack to associate the internet address of 145.9.43.188 with the X.25 PVC local logical channel identifier 231. Defaults are used for the remaining parameters.

Example 3: Adding RSI with Additional Parameters

```
ADDTCPRSI INTNETADR('135.63.45.23')
          NETADR(6031546) RVSCRG(*BOTH)
          DFTPFSIZE(1024 *TRANSMIT)
          DFTWFSIZE(*LIND *TRANSMIT)
```

This command allows the TCP/IP protocol stack to associate the internet address of 135.63.45.23 with the X.25 network address of 6031546. The reverse charges are used for both outgoing and incoming call request packets. The default packet size is set to 1024, and the default window size is set to the value specified in the line description associated with the X.25 interface used to reach the remote system.

Error messages for ADDTCPRSI

*ESCAPE Messages

TCP1D03

&1 member record length not correct.

TCP1D04

Error occurred processing member &1 of &2/&3.

TCP1901

Internet address &1 not valid.

TCP1902

Internet address &1 not valid.

TCP1908

Internet address &1 not valid.

TCP26D5

Error occurred processing file.

TCP8050

*IOSYSCFG authority required to use &1.

TCP9999

Internal system error in program &1.

ADDTCPRTE (Add TCP/IP Route) Command Description

ADDTCPRTE Command syntax diagram

Purpose

The Add TCP/IP Route (ADDTCPRTE) command is used to identify a route to a remote network or a route to a remote destination system in the Transmission Control Protocol/Internet Protocol (TCP/IP) configuration.

Five parameter values uniquely define a route. These values are the route destination (RTEDEST), the subnet mask (SUBNETMASK), the type of service (TOS), the internet address of the next system on the route (NEXTHOP), and the preferred binding interface (BINDIFC). For default routes and default multicast routes (*DFTRROUTE and *DFTMCAST), the NEXTHOP, TOS, and BINDIFC values uniquely define the route because the SUBNETMASK is always *NONE.

Restrictions:

1. You must have *IOSYSCFG special authority to use this command.
2. 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 TCP/IP interface. An interface can be added using the ADDTCPIFC command.
3. A route destination value of 127.nnn.nnn.nnn (where nnn is any value from 0 to 255) is not allowed. It is a reserved value for *LOOPBACK.

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 subnetmask, type of service value, and next hop, the route destination uniquely identifies a route to a network or system.

***DFTRROUTE:** Specifies that a default route entry is being added. A default route entry is used by the system to route data that is being sent to a remote destination that does not have a specific route defined. The system allows a maximum of 8 default route entries. The default route entries are used based on the availability of the next hop gateway and the type of service (TOS). If the application requests a specific TOS, the TOS of the default route used must match the TOS requested. If no default route is found that matches the requested TOS, the first available default route with a TOS of *NORMAL is used.

***DFTMCAST:** Specifies that a default multicast route entry is being added. A default multicast route entry is used by the system to select a local interface when sending data to a multicast group. The default multicast entry is used when the application does not specifically name the local interface over which multicast packets should be sent. When RTEDEST(*DFTMCAST) is specified, SUBNETMASK(*NONE) must be specified and the NEXTHOP parameter must be the internet address of an interface that had previously been added with the Add TCP/IP Interface (ADDTCPIFC) command.

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.

Exceptions:

- The first byte (octet) must be greater than 0 and less than 255.
- The last byte (octet) may not equal 255.
- The last byte (octet) may not equal 0 if *HOST is specified for the SUBNETMASK value.
- Routes to a broadcast address are not allowed.

SUBNETMASK

Specifies a bit mask that identifies to TCP/IP 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.

***NONE**: There is no subnet mask. If RTEDEST(*DFTRROUTE) or RTEDEST(*DFTMCAST) is specified, SUBNETMASK(*NONE) must be specified. *NONE is valid only for the *DFTRROUTE and *DFTMCAST route destination values.

***HOST**: The internet address value specified in the route destination field is a host address. The subnetmask value is calculated to be 255.255.255.255.

subnet-mask: Specify the mask of the subnet field. The internet address is in the form *nnn.nnn.nnn.nnn* where *nnn* is a decimal number ranging from 0 through 255. 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 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.

TOS Specifies the type of service to be used. The type of service defines how the internet hosts and routers should make trade-offs between throughput, delay, reliability, and cost.

***NORMAL**: Normal service is used for delivery of data.

***MINDELAY**: Minimize delay means that prompt delivery is important for data on this connection.

***MAXTHRPUT**: Maximize throughput means that a high data rate is important for data on this connection.

***MAXRLB**: Maximize reliability means that a higher level of effort to ensure delivery is important for data on this connection.

***MINCOST**: Minimize monetary cost means that lower cost is important for data on this connection.

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 TCP/IP interface. An interface can be added by using the Add TCP/IP Interface (ADDTCPIFC) command.

*HOME is no longer supported for route specifications to define a direct route generated from a previously defined interface.

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.

Optional Parameters

MTU Specifies the maximum size (in bytes) of IP datagrams that can be transmitted through this route. A datagram is a basic unit of information passed over an internet network. The minimum size of any maximum transmission unit value is 576 bytes.

***IFC**: The maximum transmission unit (MTU) is the MTU of the interface that is associated with this route.

maximum-transmission-unit: Specify a value for the maximum transmission unit in bytes. The maximum MTU that can be specified for this route depends on the type of physical connection to the network. The following table lists the maximum MTU values that can be specified based on the line type:

X.25 4096

Token ring (4 meg)
4060

Token ring (16 meg)
16388

Ethernet 802.3
1492

Ethernet Version 2
1500

DDI 4352

Frame relay
8177

Wireless 802.3
1492

Wireless Version 2
1500

Twinax (TDLC)
4105

Notes:

1. TCP/IP uses the route MTU value to calculate the size of the datagrams it sends. If you are using path MTU discovery, specify MTU(*IFC). This will allow the TCP/IP support to calculate the most efficient MTU for this route. If you are not using path MTU discovery, and you do not know the smallest MTU used by host systems along the entire path of this route, use 576.
2. The MTU of a route cannot exceed the MTU of the interface on which the NEXTHop value is accessed. If the interface's MTU value was specified as *LIND, the interface's MTU value is derived from the line description. If the route's MTU value is specified as *IFC and the interface's MTU value is specified as *LIND, both values are derived from the line description.
3. The actual MTU value used for a route is resolved during interface activation. This value is the minimum of either the specified MTU value for the route or the MTU value determined from the associated interface used by the route.

METRIC

Specifies the 'cost' associated with the use of this route. A value of 1 is a route that is close (nearby) whereas a route with a value of 15 is relatively far away. A route with a metric of 16 is considered to be unreachable (infinitely far away).

1: The route metric is set to 1.

metric-value: Specify a metric value. Valid metric values range from 1 to 16.

REDST

Specifies whether this statically-defined route is to be redistributed (made available to other routers) in the future.

***YES:** Specifies that this route will be shown (redistributed) to other routers. This is the default.

***NO:** Specifies that this route will not be shown (redistributed) to other routers.

Note:

REDST(*YES) is analogous to the RIPv1 specification of STATIC. REDST(*NO) is analogous to the RIPv1 specification of PASSIVE.

DUPRTEPTY

The duplicate route priority value allows for ordering of duplicate routes within an internal route table. It allows specification of which particular duplicate route should be tried first in cases where a route cannot establish a connection.

5: Specifies a priority value of 5 for this route.

priority-value: Specify a priority value. Valid values are in the range of 1 to 10.

Note:

Higher priority values will allow a route to be chosen over a route that specifies a lower priority. The default value of 5 is essentially a "middle of the road" priority value allowing the user to lower or raise the value from this starting point.

BINDIFC

The local IP interface to bind this route to. The binding is preferred and absolute.

If the IP interface defined for BINDIFC is active then the route specified will be bound to that interface.

***NONE:** TCP/IP will not attempt to bind this route to a particular IP interface but will bind it to the first active IP interface on the network as defined by the NEXTHOP and SUBNETMASK parameters.

binding-interface: Specify an IP interface to bind this route to. The binding is preferred and absolute.

Examples for ADDTCP RTE

Example 1: Adding a Route

```
ADDTCP RTEDEST('132.65.8.0')
SUBNETMASK('255.255.255.0') TOS(*MINDELAY)
NEXTHOP('148.92.6.40') MTU(*IFC)
```

This command specifies the following for this route:

- A route destination of a class B network.
- Subnetting through the third octet.
- A minimum delay type of service for the interface.

- This route is connected to or can be reached by going through a gateway identified as 148.92.6.40.
- The maximum transmission unit (MTU) is to be calculated based on the interface associated with the next hop for this route.

Example 2: Adding a Route with a Specific MTU

```
ADDTCPRTE RTEDEST('9.10.45.0')
SUBNETMASK('255.255.255.0') TOS(*MAXRLB)
NEXTHOP('9.5.11.128') MTU(1994)
```

This command specifies the following for this route:

- A route destination of a class A network.
- Subnetting through the third octet.
- A maximum reliability type of service for the interface.
- This route is connected to or can be reached by going through a gateway identified as 9.5.11.128.
- A maximum transmission unit (MTU) of 1994.

Example 3: Adding a Default Route

```
ADDTCPRTE RTEDEST(*DFTRROUTE)
SUBNETMASK(*NONE) TOS(*MINCOST)
NEXTHOP('186.49.126.108')
MTU(*IFC)
```

```
ADDTCPRTE RTEDEST(*DFTRROUTE)
SUBNETMASK(*NONE) TOS(*NORMAL)
NEXTHOP('129.65.34.98')
MTU(576)
```

These commands specify that:

- Two default routes are used for this host.
- Data may be routed over either default route.
- Processing will use the first *DFTRROUTE specified that also has the same type of service requested by the application.
- Minimum cost (*MINCOST) type of service is used for the first route and normal (*NORMAL) type of service is used for the second route.
- A maximum transmission unit (MTU) of *IFC is used for the first route and 576 for the second route.

Note: You cannot specify a subnetmask on a default route entry. It must equal *NONE.

Error messages for ADDTCPRTE

*ESCAPE Messages

TCP1D03

&1 member record length not correct.

TCP1D04

Error occurred processing member &1 of &2/&3.

TCP1901

Internet address &1 not valid.

TCP1902

Internet address &1 not valid.

TCP1908

Internet address &1 not valid.

TCP261C

Process completed successfully.

TCP2665

&2 &1 not added successfully.

TCP2666

&2 &1 not added.

TCP8050

*IOSYSCFG authority required to use &1.

TCP9509

Line &1 not found.

TCP9999

Internal system error in program &1.

ADDTRC (Add Trace) Command Description

ADDTRC Command syntax diagram

Purpose

The Add Trace (ADDTRC) command specifies which program statements in a program to trace in debug mode. Up to five ranges of high-level language (HLL) statements or machine instructions can be traced during the processing of a program through one or more ADDTRC commands, and up to 10 program variables can be recorded or monitored for change in each specified statement range. A separate ADDTRC command is required for each unique variable associated with a statement range. When the specified program being traced is run, the system records the sequence in which the traced statements are processed and optionally records the value of the variables associated with the trace each time a traced statement is processed. After a trace has been completed, the user can display this information using the Display Trace Data (DSPTRCDTA) command.

All of the trace ranges specified in a program are active at the same time. If both a HLL statement identifier and a machine instruction number are used to specify a given trace range, the trace range is treated as an HLL trace range. That is, in addition to tracing the machine instruction number specified, the system traces the HLL statement identifiers between that machine instruction number and the specified HLL statement identifier. More information on testing and debugging at the machine interface level is in the

CL Programming  book.

Restrictions:

1. This command is valid only in debug mode. To start debug mode, refer to 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 trace bound programs.

Optional Parameters

STMT Specifies which program statements (or machine instructions) to trace in one or more statement ranges in the program being traced.

***ALL:** All statements in the specified high-level language program are traced.

***ALLINST:** All machine instructions in the specified program are traced.

start-statement-identifier [stop-statement-identifier]: Specify the HLL statement identifiers (or machine instruction numbers) at which tracing starts and, optionally, the identifier at which tracing

stops. Up to five trace ranges can be defined at the same time for any program in debug mode. Each trace range begins with the specified starting statement, and all following statements are traced until the ending statement is reached. If only a starting statement identifier is specified for a range, the single statement specified is the only statement traced for that range. If machine instruction numbers are 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'&nbsp;' /43') ('/62'&nbsp;' /98')
```

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

PGMVAR

Specifies the names of the variables whose values are recorded when a trace statement in a program is processed. The OUTVAR parameter determines whether the values can be recorded for every trace statement processed or only when a variable changes value. The program variables can be specified either by their HLL names or by their machine-interface object-definition-table-vector (MI ODV) numbers.

Element 1: Program Variables

***NONE:** No program variables have their values recorded during tracing.

***CHAR:** This value is used 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 one to ten program variables whose values are recorded during tracing. If a variable name contains special characters, it must be enclosed in apostrophes. 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. 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 may 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, MI ODV number, asterisk (single-dimensional cross-section), or a numeric variable name can be specified for a subscript. For more information on the program-variable value, refer to parameter values used for testing and debugging.

Some examples are:

```
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 displayed. In some languages, the program variable may be based on a pointer variable. This set of values allows the user to specify the basing-pointers for the variable to be recorded. 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 qualifiers, subscripts, embedded blanks, parentheses, and commas. It does not include the enclosing apostrophes when special characters are used. An integer, MI ODV number, or a

numeric variable name can be specified for a subscript. For more information on the basing-pointer value, refer to . 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 recorded during tracing. 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.

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 being recorded in the program variable.

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 zero. 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 being recorded during the trace, starting at the position specified in 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 recorded; for a character string, the value specifies the number of characters recorded.

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

displayed-length: Specify the length of the data recorded. 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.

OUTFMT

Specifies the format in which the objects are shown.

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

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

OUTVAR

Specifies whether the values of the program variables are recorded only when their values change, or if they are recorded regardless of any of their values being changed. This parameter is ignored if PGMVAR(*NONE) is specified or assumed.

Note:

Within each range, the values of all the traced variables are always recorded the first time a statement in the range is processed. The OUTVAR parameter determines when the variables are recorded for all following statements in the range.

***CHG:** The system records the values of all the program variables when one or more of the values have changed since the last trace point. A variable is considered changed not only when its value is changed, but also when any of the displayed attributes change (such as length, lower/upper bounds, and subscript values). For example, if an array is specified and the upper bound changes for the array, the array is considered to have changed.

Note:

The value may not appear to have changed if it contains characters that cannot be shown on the display (a value less than 40 hex). The variable is still recorded even though the user cannot see the change by what is shown. If `OUTFMT(*HEX)` is specified, the changes can be observed in the traced data.

***ALWAYS:** The system should record the values of the specified variables every time any of the specified trace statements are processed, whether or not any variable had its value changed.

TRCPGM

Specifies the qualified name of the user-supplied, trace-handling program that is called when a statement being traced is reached in the program specified on the `PGM` parameter. The program with the traced statement passes informational parameters to the trace-handling program when it is called. These parameters identify the program name, the recursion level, the high-level language statement identifier, the machine instruction number at which the traced statement occurred, and a changed variable indicator. The parameters have the following formats:

1. Program name (10 bytes). Specifies the name of the program in which the traced statement was reached.
2. Recursion level (5 bytes). Specifies the recursion level number of the program in which the traced statement was reached. This value is a 1- to 5-digit number padded on the right with blanks.
3. Statement Identifier (10 bytes). Specifies the high-level language program statement identifier that was reached. If the traced statement does not correspond to a statement identifier, the parameter contains a slash (/) followed by a 4-digit hexadecimal machine instruction number.
4. Instruction number (5 bytes). Specifies the machine instruction number that corresponds to the high-level language statement at which the traced statement was reached. No slash appears in front of the machine instruction number. The value consists of 1 to 4 hexadecimal characters representing the MI instruction number, followed by one or more blanks. If the program passes a machine instruction number on the third parameter, the values on the third and fourth parameters will be the same.

All the parameter values are left-justified and padded on the right with blanks. When control returns to the program with the traced statement, processing continues.

When a trace-handling program is specified and `OUTVAR(*CHG)` is specified, the trace-handling program is called only if a program variable specified on the `PGMVAR` parameter changes. No trace data is recorded.

***NONE:** No trace-handling program is called when a traced point specified on this command is reached in a 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, trace-handling program to be called when a traced statement is reached during debugging in a batch environment. The program specified must not be the same as the program specified on the PGM parameter. If the same program is specified on both the TRCPGM and PGM parameters, results can be unpredictable. After the program runs, control is returned to the interrupted program and processing continues.

PGM Specifies the name of the program that contains the specified statement identifiers or the machine instruction numbers that are traced.

***DFTPGM**: The program currently specified as the default program contains the statements to trace.

program-name: Specify the name of the program that contains the statements to trace. The specified program must already be in debug mode.

Example for ADDTRC

```
ADDTRC STMT((100 120) (150 200))
        PGMVAR('&CTR' '&BRCTR' '&SAM')
```

This command traces program statements in the default program between the ranges of statements 100 through 120 and 150 through 200. Also, whenever the values of any of the program variables &CTR, &BRCTR, and &SAM are changed by one of the traced statements within those ranges, the values of all three are recorded before the traced statement is processed. When all of the traced statements have been processed, or when a breakpoint is reached, the Display Trace Data (DSPTRCDTA) command can be used to show the trace data collected.

Error messages for ADDTRC

*ESCAPE Messages

CPF1999

Errors occurred on command.



ADDTRCFTR (Add Trace Filter) Command Description

ADDTRCFTR Command syntax diagram

Purpose

The Add Trace Filter (ADDTRCFTR) command adds a new trace filter to the system. A trace filter identifies the trace flow (call/return) data that is to be collected during a trace session, and is meant to limit the amount of data collected by specifying a compare value. If the data in the trace record matches the compare value, then the data will be collected. If not, the data is discarded. The filter is specified on the STRTRC (Start Trace) 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.

Required Parameter

FTR Specifies the name of the trace filter to be added. If the specified filter already exists, an error

condition will occur. The user can either change the filter name or remove the existing filter using the Remove Trace Filter (RMVTRCFTR) command, and try this command again.

filter-name: Specify the name of the new trace filter.

Optional Parameters

PGMTRG

If a procedure is called that matches this trigger specification, then STRTRC will begin collecting flow trace records for this STRTRC session. The flow trace records will be collected only for the thread where the trigger occurs.

When the procedure returns and it matches the trigger specification, STRTRC will stop collecting the flow trace records.

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 STRTRC 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 flow trace records at procedure entry time. At procedure exit, the collection of flow trace records is disabled.

***ENTRY**: The specified trigger procedure enables the collection of flow trace records at procedure entry time. The collection of flow trace records continues for the duration of the trace session.

PGMFTR

Specifies the program comparisons to use for this filter.

Element 1: Relational Operator

***EQ:** Flow trace records having program data that matches the specified program are included in the data collected.

***NE:** Flow trace records having program data that matches the specified program are excluded from the data collected. These trace records will not show up.

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 STRTRC 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.

JVAFTTR

Specifies the Java package, class, and methods to be used as compare values for the Java filter.

Element 1: Relational Operator

***EQ:** Flow trace records having Java data that match the specified packages, classes, and methods are included in the data collected.

***NE:** Flow trace records having Java data that matches the specified packages, classes, and methods are excluded from the collection and will not show up.

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.

Examples for ADDTRCFTR

Example 1: Adding a trace filter for a program trigger

```
ADDTRCFTR FTR(PGMFTR) PGMTRG(MYLIB/MYPGM MYMODL *PEP *PGM *ENTRY)
```

This command adds a new trace filter named PGMFTR. If this filter is used in the Start Trace (STRTRC command), the collection of data will begin when the program entry of MYMODL module of MYPGM is called.

Error messages for ADDTRCFTR

*ESCAPE Messages

CPFAF22

Filter already exists.



ADDUSFCNNE (Add Ultimedia System Facilities Connection Entry) Command Description

ADDUSFCNNE Command syntax diagram

Purpose

The Add Ultimedia System Facilities Connection Entry (ADDUSFCNNE) command can be used to add an entry to the multimedia connector table. The entry describes the connection of a multimedia device to a switch.

Required Parameters

MMDEV

Specifies the multimedia device for which an entry is to be added. The multimedia device is specified by the multimedia device name, the remote location name, and the remote network identifier (ID).

Element 1: Device Name

device-name: Specify the name of the multimedia device.

Element 2: Remote Location Name

***NONE:** The multimedia device does not have a remote location name.

remote-location-name: Specify the remote location name of the programmable work station (PWS) that is the multimedia device.

Element 3: Remote Network ID

***NONE:** The multimedia device does not have a remote network ID.

remote-network-identifier: Specify the network identifier of the remote system of the PWS that is the multimedia device.

DEVCNN

Specifies the connector on the multimedia device that is attached to the switch.

Element 1: Device Connector

device-connector: Specify the identifying number of the multimedia device connector.

Element 2: Connector Direction

***INPUT**: The connector is used for input.

***OUTPUT**: The connector is used for output.

SWITCH

Specifies the name of the switch that is used for the multimedia device connection.

SWTCNN

Specifies the identifying number of the connector on the switch that is used for the multimedia device connection.

Optional Parameter

DFTDEVCONN

Specifies whether the multimedia device connector is the default device connector.

Note:

Only one default device connector can be added for a multimedia device.

***YES**: The multimedia device connector is the default device connector.

***NO**: The multimedia device connector is not the default device connector.

Example for ADDUSFCNNE

```
ADDUSFCNNE MMDEV(VIDEO2) DEVCNN(1 *OUTPUT)
           SWITCH(EASYON) SWTCNN(2)
```

This command adds an entry for the multimedia device VIDEO2. The connection uses the output device connector 1, the switch named EASYON, and the switch connector 2.

Error messages for ADDUSFCNNE

None

ADDUSFDEVE (Add Ultimedia System Facilities Device Entry) Command Description

ADDUSFDEVE Command syntax diagram

Purpose

The Add Ultimedia System Facilities Device Entry (ADDUSFDEVE) command can be used to add an entry to the multimedia device table. The entry describes the multimedia device.

Required Parameters

MMDEV

Specifies the multimedia device for which an entry is to be added. The multimedia device is specified by the multimedia device name, the remote location name, and the remote network identifier (ID).

Element 1: Device Name

device-name: Specify the name of the multimedia device.

Element 2: Remote Location Name

***NONE**: The multimedia device does not have a remote location name.

remote-location-name: Specify the remote location name of the programmable work station (PWS) that is the multimedia device.

Element 3: Remote Network ID

***NONE**: The multimedia device does not have a remote network ID.

remote-network-identifier: Specify the network identifier of the remote system of the PWS that is the multimedia device.

DEVTYPE

Specifies the type of multimedia device to be added.

***VIDEOTAPE**: The multimedia device is a video cassette recorder device.

***VIDEODISC**: The multimedia device is a videodisc recorder device.

***SWITCH**: The multimedia device is a video switch or a combined video and audio switch.

***TUNER**: The multimedia device is a Personal System/2* television screen.

***CODEC**: The multimedia device is a videoconferencing coder and decoder device, which is used to compress and decompress video images at the end of the phone line.

device-type: Specify the numeric value for the type of multimedia device to be added.

EXITPGM

Specifies the name of the exit program used by the multimedia device.

***ANCOR**: The exit program used is for an audio or video switch made by Ancor Communications, Inc.

***AUTOPATCH**: The exit program used is for an AutoPatch audio or video switch made by XN Technologies, Inc.

***JVC**: The exit program used is for a JVC** video cassette recorder made by Victor Company of Japan, Limited.

***NEC**: The exit program used is for a video cassette recorder made by NEC Technologies, Inc. or NEC Canada, Inc.

***NONCTBL**: The exit program used is for a multimedia device that is not controllable.

***PICTEL**: The exit program used is for a PictureTel** videoconferencing coder and decoder made by the PictureTel Corporation.

***P2200**: The exit program used is for a LaserDisc** videodisc recorder, model LD-V2200, made by the Pioneer Electronic Corporation.

***P8000**: The exit program used is for a LaserDisc videodisc recorder, model LD-V8000, made by the Pioneer Electronic Corporation.

***SIGMA**: The exit program used is for an audio or video switch made by Sigma Electronics Inc.

The name of the exit program can be qualified by a library value:

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

exit-program: Specify the name of the exit program.

Optional Parameters

DEVCTBL

Specifies whether the multimedia device can be controlled by the programmable work station (PWS). If the device is controllable, you can send instructions (such as play, stop, or rewind) over a communications line to control the multimedia device.

***YES:** The multimedia device can be controlled.

***NO:** The multimedia device cannot be controlled.

SERVER

Specifies the name of the server that controls the multimedia device.

***NONE:** The multimedia device is not controlled by a server.

server: Specify the name of the server.

SVRPORT

Specifies the identifying number of the server port that is used to communicate with the multimedia device.

***NONE:** No server port is specified. The multimedia device is not controlled by a server.

server-port-number: Specify the port number.

LINESPEED

Specifies the line speed in bits per second (bps).

9600: The speed is 9600 bps.

1200: The speed is 1200 bps.

2400: The speed is 2400 bps.

4800: The speed is 4800 bps.

PARITY

Specifies the parity state that is used when communicating with the multimedia device.

***NONE:** A parity state is not specified.

***EVEN:** The parity state is even-numbered.

***ODD:** The parity state is odd-numbered.

BITSCHAR

Specifies the number of data bits per character used when communicating with the multimedia device.

8: The bits per character is 8 bits.

7: The bits per character is 7 bits.

STOPBITS

Specifies the number of stop bits that make up the stop signal used when communicating with the multimedia device.

1: The number of stop bits is 2.

2: The number of stop bits is 1.

CAL Specifies the name of the calendar that is associated with the multimedia device.

***NONE:** A calendar is not specified.

calendar-name: Specify the name of the calendar.

TEXT Specifies the text that briefly describes the multimedia device. 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 ADDUSFDEVE

```
ADDUSFDEVE MMDEV(EASYON) DEVTYPE(*SWITCH)
EXITPGM(MYLIB/VIDEOEND)
SERVER(LOCALSWITCH) SVRPORT(8)
```

This command adds a multimedia device entry for the switching device EASYON to the multimedia device table. The switch uses the user-defined exit program VIDEOEND in the library MYLIB. The multimedia device can be controlled by the server LOCALSWITCH using the server port 8.

Error messages for ADDUSFDEVE

None

ADDUSFSVRE (Add Ultimedia System Facilities Server Entry) Command Description

ADDUSFSVRE Command syntax diagram

Purpose

The Add Ultimedia System Facilities Server Entry (ADDUSFSVRE) command can be used to add an entry to the multimedia server device table. In addition, this command creates the associated asynchronous device description, line description, and controller description iSeries 400 objects needed for the system to communicate with the server. The program device name is added to an intersystem communications function file.

Required Parameters

SERVER

Specifies the name of the multimedia server device to be added.

Note:

The name specified cannot be the same as the name of an existing device description, line description, or controller description.

RSRCNAME

Specifies the resource name of the line that is used to communicate with the multimedia server.

Optional Parameter

TEXT Specifies the text that briefly describes the multimedia server device. 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 ADDUSFSVRE

```
ADDUSFSVRE SERVER(LOCALSWITCH) RSRNAME(EASYONLINE)
TEXT('LOCAL SWITCH TO EASYON CONTROLLABLE DEVICE')
```

This command adds an entry for the multimedia server device LOCALSWITCH. The server is attached to the line with the name EASYONLINE. The text “LOCAL SWITCH TO EASYON CONTROLLABLE DEVICE” is the description for the server entry.

Error messages for ADDUSFSVRE

None

ADDWSE (Add Work Station Entry) Command Description

ADDWSE Command syntax diagram

Purpose

The Add Work Station Entry (ADDWSE) command adds a work station job entry to the specified subsystem description. Each entry describes one or more work stations that are controlled by the subsystem. The work stations identified in the work station entries are allowed to sign on the subsystem, or enter the subsystem and run jobs.

Restriction: To use this command, the user must have object operational and object management authorities for the subsystem description.

Required Parameters

SBSD Specifies the qualified name of the subsystem description to which the work station job entry is added.

Note:

The following IBM-supplied objects are not valid on this parameter:

- QLPINSTALL
- QSYSSBSD

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 in which the work station job entry is added.

WRKSTN

Specifies the name of the work station used by the subsystem. The device description name specified in the Create Device Description (Display) (CRTDEV DSP) command associated with the work station is the name that is used.

The WRKSTN parameter and the WRKSTNTYPE parameter are mutually exclusive.

work-station-name: Specify the name of the work station for which a work station entry is added.

generic-work-station-name*: Specify the generic name of the work station. 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.

WRKSTNTYPE

Specifies the type of work station associated with the entry being added. This entry applies to all work stations of this type that do not have specific entries for an individual work station.

The WRKSTN parameter and the WRKSTNTYPE parameter are mutually exclusive.

The following type codes are valid:

Type Code	Device
3179	3179 Display Station
3180	3180 Display Station
3196	3196 Display Station
3197	3197 Display Station
3277	3277 Display Station
3278	3278 Display Station
3279	3279 Display Station
3476	3476 Display Station
3477	3477 Display Station
3486	3486 Display Station
3487	3487 Display Station
5251	5251 Display Station
5291	5291 Display Station
5292	5292 Color Display Station
5555	5555 Display Station (on systems supporting DBCS (double-byte character set))

***ALL:** The work station entry for all valid work station types is added.

***NONASCII:** The work station entry for all valid work stations that use 5250 data streams is added.

***ASCII:** The work station entries for all work stations that use ASCII data streams are added.

***CONS:** System console display. This entry overrides a device type entry that specifies the same device type as the device being used as the console.

work-station-type: Specify the name of the work station device type for which work station entry is added.

Optional Parameters

JOB Specifies the qualified name of the job description used for jobs that are created and processed through this work station 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.

***USRPRF:** The job description named in the user profile of the user that signs on at this work station (or at this type of work station) is used for jobs that are started through this entry.

***SBSD:** The job description having the same name as the subsystem description specified by the SBSDB parameter is used for jobs created through this 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 name of the job description used for jobs that are created through this entry.

MAXACT

Specifies, for work stations that use this work station job entry, the maximum number of work station jobs that can be active at the same time. More information on this parameter 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 work entry.

AT Specifies when the work stations associated with this job entry are allocated. For more information on how work stations are allocated to subsystems, see the Start Subsystem (STRSBS) command description.

Note:

The following should be considered if two or more work station entries specify AT(*SIGNON), if they apply to the same work station and if they are in more than one subsystem description. If the work station is varied on while more than one of the subsystems are active, it cannot be predicted to which subsystem the work station is assigned.

***SIGNON:** The work stations are allocated when the subsystem is started if the work station is not already in use (signed on) in another subsystem. A sign-on prompt is shown at each work station associated with this work entry. If a work station becomes allocated to a different subsystem, interactive jobs associated with the work station are allowed to enter this subsystem through the Transfer Job (TFRJOB) command.

***ENTER:** The work stations associated with this work entry are not allocated when the subsystem is started. However, the interactive jobs associated with the work stations are allowed to enter this subsystem through the Transfer Job (TFRJOB) command.

Examples for ADDWSE

Example 1: Adding a Work Station Job Entry

```
ADDWSE SBSDB(LIB7/ORDER) WRKSTNTYPE(5251)
      JOBD(QCTL) AT(*SIGNON)
```

This command adds a work station job entry to a subsystem description named ORDER in library LIB7. All type 5251 work stations are allocated to this subsystem when the subsystem is started, unless they are already active in a previously started subsystem. The work stations are signed on when requested by the user. After sign-on is complete, the IBM-supplied job description QCTL is used to start the routing step.

Example 2: Adding a Work Station Job Entry

```
ADDWSE SBS(D/LIB7/ORDER) WRKSTN(A12)
      JOBD(LIB7/ORDER) AT(*SIGNON)
```

This command adds a work station job entry to a subsystem description named ORDER in library LIB7. Work station A12 is signed on when requested by a user.

Error messages for ADDWSE

*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.

ALCOBJ (Allocate Object) Command Description

ALCOBJ Command syntax diagram

Purpose

The Allocate Object (ALCOBJ) command is used to reserve an object or list of objects for use later in the job or thread. If an object that is needed is not specified in an ALCOBJ command, an allocation is made automatically when the object is used.

Objects can be deallocated with the Deallocate Object (DLCOBJ) command. Allocated job-scoped locks are automatically released when the job ends. Allocated thread-scoped locks are automatically released when the thread ends. If a thread received a job-scoped lock, the job will continue to hold that lock after the requesting thread ends. ➤ Lock-space-scoped locks are not automatically released. ⏪

The DLCOBJ command should not be issued for an object that was not explicitly allocated by the ALCOBJ command. If the DLCOBJ command is used this way, internal locks on the object are released, making the object capable of being deleted.

Notes:

1. When allocating database files, use the DLCOBJ command before deleting the file if the file being allocated is a logical file.
2. If a file is being allocated that is affected by a file override, the ALCOBJ command ignores the override and attempts to allocate the file named in the OBJ parameter.
3. When allocating distributed data management (DDM) files and distributed files, additional time is required for the command to complete because of the time required for communication and for allocating files on remote systems.
4. Work station message queues cannot be allocated. A work station message queue is associated with a work station device description of the same name. Therefore, to do an operation on a work station message queue that must be allocated, the user must allocate the associated device description. When the device description is allocated, the work station message queue is implicitly allocated.

5. When ALCOBJ is executed to get an EXCL lock on a program (*PGM), only the program object description is locked. The program code is not locked exclusively. Therefore, the program may still be run by another user. Changes are not allowed for the program object description while the actual program can still be used.
6. The system does not lock programs when calling them.
7. When ALCOBJ is executed to get an EXCL lock on a logical file member (*FILE), the lock occurs on both the logical file member and the associated physical file members. No other user can use the physical file members (not even through some other logical file member).

Restrictions:

1. This command cannot be used to allocate a device description, *DEV D, for an Advanced Program-to-Program Communications (APPC) device or for an intrasystem (INTRA) device.
2. This command can be used to allocate only the following database *FILE types:
 - Physical files
 - Logical files
 - Distributed files
This allocates the piece of the file on each node in the node group.
 - DDM files
This allocates both the DDM file on the local system and the file on the remote system that is identified in the DDM file.
3. The object must exist on the system.
4. The user issuing the command must have object operational authority for the object.
5. If the allocation cannot be completed, none of the locks are granted, and a message is sent to the thread that issued the command. If the command is issued from a program, the Monitor Message (MONMSG) command can be used to determine whether the allocation was successful.
6. In multithreaded jobs, this command is not threadsafe for distributed files. This command is also not threadsafe and fails for Distributed Data Management (DDM) files of type *SNA.

Required Parameter

OBJ Specifies the qualified names of one or more system objects that are to be allocated to the job, thread, >> or lock space, << the type of each object specified, the lock state of each object, and the member name (if the object is a database file or DDM file).

Element 1: Name of the Object to be Allocated

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

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

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

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

object-name: Specify the name of the object to be allocated.

Element 2: Type of Object to be Allocated


object-type: Specify the type of the object to be allocated. Refer to Table 1 (96), for the list of object types that are valid for this command.

Element 3: Lock State for the Object to be Allocated

Specify the lock state for the object. Valid lock states include the following:

Value	Lock State Meaning
*SHRRD	Shared for read
*SHRUPD	Shared for update
*SHRNUP	Shared, no update
*EXCLRD	Exclusive, allow read
*EXCL	Exclusive, no read

You can specify all five lock states (*EXCL, *EXCLRD, *SHRUPD, *SHRNUP, and *SHRRD) for most, but not all, object types. Refer to Table 1 (96), for the list of object types and the valid lock states allowed for each object type.

Note: Additional details about lock states can be found in the CL Programming  book.

Multiple locks can be specified for the same object in the same job with duplicate or different lock states. Each lock is held separately. For example, if an *EXCL lock is already held for an object, and a second *EXCL lock request occurs, the second lock is acquired. Both locks must be released in the job (deallocated with the DLCOBJ command) before another job can access the same object.

Element 4: Member of the Database File to be Allocated

Note: The following values can only be specified if the object type is a database file.

***FIRST:** The first member of the database file is allocated to the job.

member-name: Specify the name of the member to be allocated to the job or thread. If the specified file is a logical file, the physical file members associated with the members of the logical file are also allocated to the job or thread.

Optional Parameters

WAIT Specifies the number of seconds that the program waits for the object to be allocated. If the object cannot be allocated in the specified wait time, a message, which can be detected by a Monitor Message (MONMSG) command, is sent to the program. If one or more device descriptions are in the list of objects to be allocated, the system may wait more than the specified amount of time to attempt the allocation.

When allocating DDM files and distributed files, additional time is required for communications and for allocating files on remote systems. A separate WAIT time is used for each remote system. [»](#)
When allocating objects with a lock-space-scope, the lock space may override the wait time specified. [«](#)

***CLS:** The job default wait time is used as the wait time for the resources being allocated.

SCOPE

Specify the scope for this lock request.

***JOB:** The lock is scoped to the job. [»](#)

***LCKSPC:**The lock is scoped to the lock space attached to the current thread. If no lock space is attached, the lock is scoped to the job. <<

***THREAD:** The lock is scoped to the thread.

All object types supported by the OBJ parameter support job-scoped locks. >> All object types supported by the OBJ parameter support lock-space-scoped locks. When allocating DDM objects with a lock-space-scope, the lock on the remote system is scoped to the job. << To determine if an object type supports thread-scoped locks refer to Table 1.

Locks scoped to a thread can never conflict with a lock scoped to its containing job, but may conflict with a lock scoped to a different job or any other thread (depending on the lock states involved). >> Locks scoped to a lock space can conflict with a lock scoped to the thread or job associated with the lock space. <<

CONFLICT

Specify the action to be taken if a lock conflict exists. This parameter is only supported for certain database *FILE objects. This parameter is ignored for all other objects. The supported database *FILE objects are:

- Physical file
- Logical file
- Distributed file

***NORQSRLS:** No requests are sent to the jobs or threads which are holding conflicting locks.

***RQSRLS:** A request is sent to the system code running in each job and thread that is holding a conflicting lock for the specified object. Notification of lock contention is not visible to user applications which hold conflicting locks. Only locks which are acquired implicitly by system code are eligible to be released. Locks acquired explicitly by user application code are not eligible to be released. If *RQSRLS specified for a distributed file, the request to release the lock is sent to each node in the node group that holds a conflicting lock.

Table 1. Object Information

Object Type	Object Type Definition	Lock States ¹					Thread Scope ²
		*EXCL	*EXCLRD	*SHRUPD	*SHRNUP	*SHRRD	
*AUTL	Authorization List	x	x	x	x	x	
*BNDDIR	Binding directory	x	x			x	
*CLD	C Locale description	x	x	x	x	x	
*CRQD	Change request description	x	x	x	x	x	
*CSI	Communications side information	x	x	x	x	x	
*DEVD	Device Description		x	x			x
*DTAARA	Data area	x	x	x	x	x	x
*DTADCT	Data dictionary	x	x	x	x	x	x
*DTAQ	Data queue	x	x	x	x	x	x
*FCT	Forms control table	x	x	x	x	x	
*FILE	File	x	x	x	x	x	x

*FNTRSC	Font resource	x	x	x	x	x	
*FNTTBL	Font mapping table	x	x	x	x	x	
*FORMDF	Form definition	x	x	x	x	x	
*IMGCLG	Image catalog	x	x	x	x	x	x
*IPXD	Internet packet exchange description	x	x	x	x	x	x
*LIB	Library		x	x	x	x	x
*LOCALE	Locale space object	x	x	x	x	x	x
*MEDDFN	Media definition	x	x	x	x	x	
*MENU	Menu	x	x	x	x	x	
*MGTCOL	Management collection	x	x	x	x	x	x
*MODULE	Module	x	x			x	
*MSGQ	Message queue	x				x	x
*NODL	Node list	x	x	x	x	x	
*NTBD	NetBIOS description	x	x	x	x	x	x
*NWSD	Network server description	x	x	x	x	x	x
*OVL	Overlay	x	x	x	x	x	
*PAGDFN	Page definition	x	x	x	x	x	
*PAGSEG	Page segment	x	x	x	x	x	
*PDG	Print descriptor group	x	x	x	x	x	
*PGM	Program	x	x			x	x
*PNLGRP	Panel group	x	x	x	x	x	
*PSFCFG	Print service facility configuration object	x	x	x	x	x	
*QMFORM	Query management form	x	x	x	x	x	
*QMQR	Query management query	x	x	x	x	x	
*QRYDFN	Query definition	x	x	x	x	x	
*S36	S/36 machine description	x	x	x	x	x	
*SBSD	Subsystem description	x					x
*SCHIDX	Search index	x	x	x	x	x	

*SQLPKG	Structured Query Language package	x	x	x	x	x	
*SRVPGM	Service program	x	x	x	x	x	x
*SSND	Session description	x	x	x	x	x	
*USRIDX	User index	x	x	x	x	x	x
*USRQ	User queue	x	x	x	x	x	x
*USRSPC	User space	x	x	x	x	x	x
*VLDL	Validation list object	x	x	x	x	x	x
*WSCST	Workstation customizing object	x	x	x	x	x	

¹ x indicates this lock state is allowed for this object type.

² x indicates a thread-scoped lock is allowed for this object type.

Examples for ALCOBJ

Example 1: Allocate File for Job

```
ALCOBJ OBJ((LIBB/FILEA *FILE *EXCL MEMBERA)) SCOPE(*JOB)
```

This command exclusively allocates MEMBERA of FILEA in LIBB to the job in which the ALCOBJ command is used. If MEMBERA is unavailable, the number of seconds to wait for it to become available is the job default wait time.

Example 2: Allocate Data Area for a Thread

```
ALCOBJ OBJ((LIBY/DATAAREAX *DTAARA *EXCL ))
SCOPE(*THREAD)
```

This command exclusively allocates DATAAREAX in LIBY to the requesting thread in which the ALCOBJ command is used. If DATAAREAX is unavailable, the number of seconds to wait for it to become available is the job default wait time. >>

Example 3: Allocate File for Lock Space

```
ALCOBJ OBJ((LIBB/FILEA *FILE *EXCL MEMBERA)) SCOPE(*LCKSPC)
```

This command exclusively allocates MEMBERA of FILEA in LIBB to the lock space attached to the current thread. If no lock space is attached, the lock is scoped to the job. <<

Error messages for ALCOBJ

*ESCAPE Messages

CPF1002

Cannot allocate object &1.

CPF1040

Maximum number of objects allocated on system.

CPF1085

Objects not allocated.

ANZACCGRP (Analyze Process Access Group) Command Description

Notes:

- This command is part of the 5722-PT1 (Performance Tools for iSeries) licensed program.
- You should not use this command because the Licensed Internal Code no longer uses process access groups for caching data used by a job.

ANZACCGRP Command syntax diagram

Purpose

The Analyze Process Access Group (ANZACCGRP) command produces a report that summarizes the process access group (PAG) data collected earlier with the Display Access Group (DSPACCGRP) command. The report is useful in examining the process access group data from a large number of jobs.

The ANZACCGRP command produces a three-part report that includes the environment section, job section, and file section. The environment section shows information summarized by job types, the average number of files, duplicate files, display files, I/O counts, and process-access group size for the different job types. The job section shows information for each selected job. The file section shows information for each open file, the file name and type, count, and the average open data path size. The report output is written to the printer file QSYSPT.

Optional Parameters

MBR Specifies the member in the file QAPTPAGD in which the process access group data is saved by the Display Access Group (DSPACCGRP) command.

QAPAGDTA: The standard member name, QAPAGDTA, is used.

member-name: Specify the name of the member in which the data is saved.

LIB Specifies the library where the process access group data is saved by the DSPACCGRP command.

QPFRDATA: The process access group data is saved in the IBM-supplied performance data library, QPFRDATA.

library-name: Specify the name of the library where the process access group data is saved.

TITLE Specifies a short title that is printed at the top of each page of the report.

***BLANK:** No title is used.

'report-title': Specify a title of up to 40 characters, enclosed in apostrophes.

JOB Specifies the job name used if the job is submitted for batch processing.

Note: If *NONE is specified on the JOBD parameter, this parameter is ignored; job processing is performed interactively.

ANZACCGRP: The command name is used for the job name.

***MBR:** The name selected for the performance data member on the MBR parameter is used.

job-name: Specify the name to be used for any and all batch jobs.

JOBD Specifies the job description used to submit jobs for batch processing.

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.

QPFRJOB: The IBM-supplied job description, QPFRJOB, is used.

job-description-name: Specify the name of an alternative job description.

Other Single Values

***NONE:** A batch job is not submitted; instead, processing continues interactively while the user waits. The user's workstation is not available for other use during this time, which could be significant for long jobs.

Example for ANZACCGRP

ANZACCGRP

This command produces a report from the process access group data previously stored in the default location, member QAPAGDTA of QPFRDATA/QAPTPAGD, by the Display Access Group (DSPACCGRP) command.

Error messages for ANZACCGRP

*ESCAPE Messages

PFR5431

Cannot access process access group data file.

PFR9802

Unexpected message monitored.

ANZDBF (Analyze Database Files) Command Description

Note: To use this command, you must have the 5722-PT1 (Performance Tools for iSeries) licensed program installed.

ANZDBF Command syntax diagram

Purpose

The Analyze Database Files (ANZDBF) command produces two reports that show the physical and logical files in a set of libraries and the relationships between the files. It saves the information in a database file for further analysis by the Analyze Database File Keys (ANZDBFKEY) command. Both reports (physical to logical file relationships and logical to physical file relationships) are written to the printer file QPPTANZD. Two printer files with the same name are produced. One printer file contains summary data; the other contains detail data. The data is saved in member QAPTAZDR of the database file QPFRDATA/QAPTAZDR.

Required Parameter

LIBL Specifies the libraries that contain the database files on which to report.

Optional Parameters

JOB Specifies the job name used if the job is submitted for batch processing.

Note: If *NONE is specified on the JOBD parameter, this parameter is ignored; job processing is performed interactively.

ANZDBF: The command name is used for the job name.

job-name: Specify the name to be used for any and all batch jobs.

JOBD Specifies the job description used to submit jobs for batch processing.

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.

QPFRJOB: The IBM-supplied job description, QPFRJOB, is used.

job-description-name: Specify the name of an alternative job description.

Other Single Values

***NONE:** A batch job is not submitted; instead, processing continues interactively while the user waits. The user's work station is not available for other use during this time, which could be significant for long jobs.

Example for ANZDBF

```
ANZDBF  LIBL(APDTA ARDTA)
```

This command produces reports showing the relationships for all files in the Accounts Payable (APDTA) and Accounts Receivable (ARDTA) data libraries.

Error messages for ANZDBF

*ESCAPE Messages

CPF9801

Object &2 in library &3 not found.

CPF9802

Not authorized to object &2 in &3.

PFR9802

Unexpected message monitored.

ANZDBFKEY (Analyze Database File Keys) Command Description

Note: To use this command, you must have the 5722-PT1 (Performance Tools for iSeries) licensed program installed.

ANZDBFKEY Command syntax diagram

Purpose

The Analyze Database File Keys (ANZDBFKEY) command produces, from the data created by the ANZDBF command, two reports showing the key structure of the database files.

One report is written to the printer file QPPTANZK. The other report is written to the printer file QPPTANKM. QPPTANZK contains a listing of the access paths (logical files only) and selection criteria for each key field or selection rule. QPPTANKM contains a matrix of the key fields for all logical files based on the physical file.

Optional Parameters

FILE Specifies which physical files to select from the list processed earlier by the Analyze Database File (ANZDBF) command. The report includes all logical files associated with each selected physical file.

***NUMLF:** Specifies all physical files that have at least a minimum number of associated logical files. The specific minimum is defined by the NUMLF parameter.

file-name: Specify the name of a particular physical file.

NUMLF

Specifies, if FILE(*NUMLF) is specified, the minimum number of logical files associated with a physical file before that physical file is selected.

5: At least 5 logical files must be associated with a physical file.

file-count: Specify the minimum number of logical files required.

JOB Specifies the job name of a job submitted for batch processing.

Note: If *NONE is specified on the JOB parameter, this parameter is ignored; job processing is performed interactively.

ANZDBFKEY: The command name is used for the job name.

job-name: Specify the name to be used for any and all batch jobs.

JOB Specifies the job description used to submit jobs for batch processing.

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.

QPFRJOB: The IBM-supplied job description, QPFRJOB, is used.

job-description-name: Specify the name of an alternative job description.

Other Single Values

***NONE:** A batch job is not submitted; instead, processing continues interactively while the user waits. The user's workstation is not available for other use during this time, which could be significant for long jobs.

Example for ANZDBFKEY

```
ANZDBFKEY FILE(*NUMLF) NUMLF(2)
```

This command produces reports on the keys for all files that refer to physical files with at least two associated logical files.

Error messages for ANZDBFKEY

*ESCAPE Messages

PFR5251

Cannot access data to analyze database file.

PFR9802

Unexpected message monitored.

ANZDFTPWD (Analyze Default Passwords) Command Description

ANZDFTPWD Command syntax diagram

Purpose

The Analyze Default Passwords (ANZDFTPWD) command allows you to print a report of all the user profiles on the system that have a default password and to take an action against the profiles. A profile has a default password when the profile's password matches the user profile name.

When the system is operating at password level 2 or 3, both the uppercase and lowercase values of the user profile name are checked. However, mixed case values of the user profile name will not be checked. For example, if the user profile JAMES has a password of 'JAMES' or 'james' it will be detected as having a default password; but passwords of 'JaMeS' or 'James' will not be detected as default passwords.

Restriction: You must have *ALLOBJ and *SECADM special authorities to use this command.

Optional Parameter

ACTION

Specifies the action to be taken against the user profiles that have a default password. Multiple actions may be specified, except for *NONE.

***NONE:** No action is taken against profiles with a default password.

***DISABLE:** The 'Status' field in the user profile is set to *DISABLED.

***PWDEXP:** The 'Set password to expired' field in the user profile is set to *YES.

Example for ANZDFTPWD

```
ANZDFTPWD ACTION(*DISABLE *PWDEXP)
```

Any user profiles on the system that have a default password will be disabled and their passwords will be set to expired.

Error messages for ANZDFTPWD

*ESCAPE Messages

CPFB301

Cannot open file &2 in library &3.

CPFB302

Not authorized to check for default passwords.

ANZJVAPGM (Analyze Java Program) Command Description

ANZJVAPGM Command syntax diagram

Purpose

The Analyze Java Program (ANZJVAPGM) analyzes an iSeries 400 Java program, lists its classes and shows the current status of each class.

Restrictions: The file must be in one of the following file systems: QOpenSys, "root", or a user-defined file system.

Required Parameter

CLSF Specifies the Java class or Java Archive (JAR) file name that identifies the Java program to analyze. The file name may be qualified by one or more directory names.

Optional Parameters

CLASSPATH

Specifies a list of directories or JAR files to be searched for classes referenced by the Java program being analyzed.

***PGM:** The class path to use is the same one used to create the Java program being analyzed.

***ENVVAR:** The class path is to be taken from the environment variable CLASSPATH.

class-path: Path used to locate classes. An example class path is '/directory1/directory2:/QIBM/ProdData/Java400'.

JDKVER

Specifies the Java Development Kit (JDK) version to use when analyzing the Java program.

***PGM:** The JDK version used to create the Java program being analyzed is to be used.

***CURRENT:** The JDK version currently installed as the system default is to be used.

Java-Development-Kit-version: The JDK version to be used. An example JDK version is '1.2.2'.

DETAIL

Specifies the amount of detail to show.

***NONCURRENT:** Show only those classes which are not current with respect to the given classpath.

***FULL:** Show all classes included in the Java stand-alone program. A status of **current** or **not current** will be shown for each class.

OUTPUT

Specifies whether the output from the command is shown at the requesting workstation or printed with the job's spooled output. More information on this parameter is in Commonly used parameters.

*****: Output requested by an interactive job is shown on the display. Output requested by a batch job is printed with the job's spooled output.

***PRINT:** The output is printed with the job's spooled output.

Examples for ANZJVAPGM

Example 1: Analyze Java Program Using Command Defaults

```
ANZJVAPGM FILE('/projectA/team2/myJavaclassname.jar')
```

This command will analyze the Java program associated with file */projectA/team2/myJavaclassname.jar*. The classpath used will be taken from the Java program being analyzed. The Java Development Kit version used will be the version used to create the Java program being analyzed. Only those classes

which are not current with respect to the classpath will be shown. If the command is being run interactively, the results will be shown on the display. If the command is run in a batch job, the results are written to a spooled file.

Example 2: Analyze Java Program

```
ANZJVAPGM FILE('/projectB/myJavaclassfile.jar')
          CLASSPATH(*ENVVAR) JDKVER('1.1.8') OUTPUT(*PRINT)
```

This command will analyze the Java program associated with file */projectB/myJavaclassfile.jar*. The classpath used will be taken from environment variable CLASSPATH. The Java Development Kit version used will be version 1.1.8. Only those classes which are not current with respect to the classpath will be shown. The results are written to a spooled file.

Error messages for ANZJVAPGM

None >>

ANZJVM (Analyze Java Virtual Machine) Command Description

ANZJVM Command syntax diagram

Purpose

The Analyze Java Virtual Machine (ANZJVM) command collects information about the Java Virtual Machine (JVM) for a specified job. A set of JVM information is collected immediately when the command is run. This collected JVM data is called a snapshot. A second snapshot is done a specified amount of time later. By taking a snapshot of the JVM and comparing the data with a snapshot taken at a later time, the data can be analyzed to help find object leaks. The information is dumped using printer device file QSYSPRT. The user data for the QSYSPRT file is 'ANZJVM'. The dump includes formatted information about the JVM heap. Details include names of classes, number of active objects per class, and the class loader used to load each class.

Restrictions:

- This command uses the Start Service Job (STRSRVJOB) and Start Debug (STRDBG) commands. The user of this command must be authorized to those commands.
- This command is shipped with public *EXCLUDE authority and the QPGMR, QSYSOPR, QSRV, and QSRVBAS user profiles have private authorities to use the command.
- This command must be run under a user profile that is the same as the job user identity of the JVM job, or that has use (*USE) authority to the job user identity of the JVM job.
- This command is not allowed if the remote service operation has been started for another job and that job is not the same job specified on this command.
- This command is not allowed if the JVM job is held, suspended, or ending.

Optional Parameters

JOB Specifies the name of the job where the Java Virtual Machine (JVM) is running. If no job number is given, all of the jobs currently in the system are searched for the simple job name. The job name entered must be a job in which a JVM is currently running.

***SRVJOB:** Information about the JVM in the job currently being serviced will be dumped. If no job is currently being serviced, then a job identifier is required.

A job identifier is a qualified name with up to three elements. For example:

```
job-name
user-name/job-name
job-number/user-name/job-name
```

job-name: Specify the name of the JVM job.

user-name: Specify the name of the user of the JVM job.

job-number: Specify the number of the JVM job.

INTERVAL

Specifies the time interval in seconds between the snapshots of the JVM to be analyzed.

60: Sixty seconds will pass between the time data is collected.

number-of-seconds: The maximum number (0-3600) of seconds that will pass between JVM snapshots of data.

FRCGC

Specifies if a garbage collection cycle should be forced to take place.

***YES**: A garbage collection cycle will take place before each snapshot of data is collected.

***NO**: No garbage collection cycle will be forced to take place while collecting the data.

SORT Specifies the order in which the information is sorted.

***NUMOBJCHG**: Information is sorted by the number of objects changed.

***NUMOBJ**: Information is sorted by number of objects in the first snapshot.

***SIZECHG**: Information is sorted by the change in the amount of space used by the object.

***SIZE**: Information is sorted by the amount of space used by the object.

***NAME**: Information is sorted by the class name.

DUPJOB OPT

Specifies the action taken when duplicate jobs are found by this command.

***SELECT**: The selection display is shown when duplicate jobs are found during an interactive session. Otherwise, an escape message is issued.

***MSG**: An escape message is issued when duplicate jobs are found.

Example for ANZJVM

```
ANZJVM JOB(099246/FRED/QJVACMDSRV) INTERVAL(60)
```

This command will collect two snapshots of the Java Virtual Machine (JVM), 60 seconds apart, for the job with job name QJVACMDSRC, user name FRED, and job number 099246. The analyzed data from the snapshots is written to a spooled file. The spooled file name will be QSYSPRT and the spooled file user data text will be ANZJVM.

Error messages for ANZJVM

***ESCAPE Messages**

JVAB602

Job parameter required.

JVAB603

Unable to open print file.

JVAB605

ANZJVM failed with reason code &1

JVAB60A

Job not found

CPF1938

Command is not allowed while serviced job is not active.

CPF3524

More than one job with specified name found.

CPF3536

Job completed and cannot be serviced.

CPF3938

Already servicing another job.

CPF9824

Not authorized to command &2 in library &3.



ANZLIBBRM (Analyze Libraries Using BRM) Command Description

Note: To use this command, you must have the 5722-BR1 (Backup Recovery and Media Services for iSeries) licensed program installed.

ANZLIBBRM Command syntax diagram

Purpose

The Analyze Libraries using BRM (ANZLIBBRM) command prints an analysis of libraries that you have backed up as well as those that you did not back up. The size of the library and the number of objects is listed for each library. Before you can run the ANZLIBBRM command, you must have first run the RTVDSKINF command. The RTVDSKINF command creates the file that ANZLIBBRM uses to analyze your libraries.

The report that is produced is the Library Backup Analysis report. The report, if printed, is written to the printer file QP1ALA.

There are no parameters for this command.

Error messages for ANZLIBBRM

None

ANZPFRDTA (Analyze Performance Data) Command Description

Note: To use this command, you must have the 5722-PT1 (Performance Tools for iSeries) licensed program installed.

ANZPFRDTA Command syntax diagram

Purpose

The Analyze Performance Data (ANZPFRDTA) command produces recommendations to improve the performance of the user's system. In the interactive mode, you can request that the system make the

recommended changes. In the batch mode, the recommended changes are printed, and you must then enter the individual commands to make the recommended changes.

Optional Parameters

MBR Specifies the member that contains the performance data collected by Collection Services.

***SELECT:** An interval selection display is shown from which you can select one or more intervals to include. This value is valid in the interactive mode only.

member-name: Specify the name of the member that contains the performance data.

LIB Specifies the library where the performance data is located.

QPFRDATA: The data is located in the IBM-supplied performance data library, QPFRDATA.

library-name: Specify the name of the library where the performance database files are located.

PERIOD

Specifies the period of time on which to report. The parameter consists of four elements: a starting time and date, and an ending time and date. Data that is collected prior to the starting time on the starting date and after the ending time on the ending date is not included in the report.

The symbol *N is used to designate the default value for any of the four elements.

Element 1: Starting Time

Specify the starting time of the reporting period. Data collected prior to this time is not included in the report.

***FIRST:** Data records that start from the beginning of the first day (00:00:00) of the collection period are included.

***SELECT:** An interval selection display from which time intervals can be selected for inclusion is shown. This value is valid in the interactive environment only. If this value is used, the remaining values of this parameter (starting time and date and ending time and date) are ignored.

start-time: Specify the time of the first data record to include in the report. 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.

Element 2: Starting Date

Specify the starting date of the reporting period. Data collected prior to the starting time on this date is not included in the report.

***FIRST:** Records starting from the beginning of the first day (00:00:00) of the collection period are included in the report.

start-date: Specify the date of the first data record to include in the report. The date must be entered in the format specified by the system value QDATFMT and, if separators are used, as specified by the system value QDATSEP.

Element 3: Ending Time

Specify the ending time of the reporting period. Data collected after this time on the ending date is not included in the report.

***LAST:** Data records through the end of the day (23:59:59) are included in the report.

end-time: Specify the time of the last data record to include in the report. See the *start-time* value description in this parameter for details on how the time must be specified.

Element 4: Ending Date

Specify the ending date of the reporting period. Data collected after the ending time on this date is not included in the report.

***LAST:** Data records through the last day of the collection period are included in the report.

end-date: Specify the date of the last record to include in the report. The date must be entered in the format specified by the system value QDATFMT and, if separators are used, as specified by the system value QDATSEP.

OUTPUT

Specifies whether the output from the command is shown at the requesting workstation or printed with the job's spooled output. More information on this parameter is in Commonly used parameters.

***:** Output requested by an interactive job is shown on the display. Output requested by a batch job is printed with the job's spooled output.

***PRINT:** The output is printed with the job's spooled output.

DATATYPE

Specifies the type of data that is analyzed.

***ALL:** All data (sample data and trace data) is analyzed.

***SAMPLE:** Only sample data is analyzed.

Example for ANZPFRDTA

ANZPFRDTA

This command provides recommendations for improving the performance of the system.

Error messages for ANZPFRDTA

*ESCAPE Messages

PFR1010

Cannot process request because of missing data.

PFR5501

Performance data file(s) are not upward compatible.

PFR5502

Cannot process request because of missing data.

PFR5501

Performance data file(s) are not downward compatible.

PFR7003

Cannot show performance data.

PFR7004

Cannot show performance data.

ANZPRB (Analyze Problem) Command Description

ANZPRB Command syntax diagram

Purpose

The Analyze Problem (ANZPRB) command allows you to analyze, create problem records for, or report problems that are not detected by the system. For example, you can analyze or report:

- Job or programming problems.
- Equipment or communications problems.
- Problems that made it necessary to do an initial program load (IPL) of the system again.
- Problems on a device or system not attached to your system.

Restriction: This command is shipped with public *EXCLUDE authority and the QPGMR, QSYSOPR, QSRV, and QSRVBAS user profiles have private authorities to use the command.

Optional Parameters

ANZTYPE

Specifies the type of analysis to do. The default value, *MENU, allows you to do a local or remote analysis on an iSeries 400 or to do a remote analysis on another system that is not an iSeries 400. If you want to do an analysis on a local or remote iSeries 400, then you can bypass the initial menu by specifying *LOCAL or *REMOTE.

Note:

The user cannot do remote analysis unless the System Manager for iSeries 400 licensed program is installed.

***MENU:** The *Select Type of System* menu is shown. You can analyze problems on:

- The iSeries 400.
- Another iSeries 400 which is enrolled as a service requester.
- Another iSeries 400 which is not enrolled as a service requester.
- Another system which is not an iSeries 400.

***LOCAL:** Local analysis is selected. Problem analysis is done on the iSeries 400.

***REMOTE:** Remote analysis is selected. Problem analysis is done for another iSeries 400 that is enrolled as a service requester.

RCPNAME

Specifies the remote control point name for the service requester system where the remote analysis is done.

NETID Specifies the network identifier (ID) for the service requester system where the remote analysis is done.

***NETATR:** The network ID of the service provider is used.

network-identifier: Specify the network ID.

USERID

Specifies the user identifier (ID) used to access the remote system.

PASSWORD

Specifies the password used to access the remote system.

Note:

This field may be left blank if the service requester system is an unsecured system.

***NONE:** No password is needed to access the remote system because the remote system has a security level of 10.

password: Specify the password used to access the remote system.

Examples for ANZPRB**Example 1: Displaying the Menu**

```
ANZPRB
```

This command shows the Analyze Problem menu.

Example 2: Starting Remote Analysis

```
ANZPRB ANZTYPE(*REMOTE)
```

This command shows the display which prompts for the remaining values of the command. After you specify the appropriate values, remote analysis begins.

Example 3: Accessing Remote System with User ID and Password

```
ANZPRB ANZTYPE(*REMOTE) RCPNAME(RCH38377)  
USERID(JON) PASSWORD
```

This command shows the display which prompts for the remaining values of the command. After you specify the appropriate values beyond the ones specified on the command example, remote analysis begins.

Example 4: Remote Analysis has Security Level of 10

```
ANZPRB ANZTYPE(*REMOTE) RCPNAME(RCH38377)  
USERID(JON)
```

This command is slightly different than the preceding example. The same display prompt appears; however, if **PASSWORD** is not specified, the system assumes that the remote system has a security level of 10, that is, it does not use passwords. After you specify the appropriate values beyond the ones specified on the command example, remote analysis begins.

Example 5: Displaying the Menu

```
ANZPRB ANZTYPE(*MENU)
```

This command shows a menu prompting you for the type of analysis to be done. The remaining parameters do not appear on the display.

Example 6: Starting Local Analysis

```
ANZPRB ANZTYPE(*LOCAL)
```

This command begins analysis on the local device. The remaining parameters do not appear on the display.

Error messages for ANZPRB

*ESCAPE Messages

CPF2B01

Problem analysis cannot continue because of an error in the program.

CPF2B3C

Licensed program &1 not installed.

CPF9308

Unable to complete problem analysis. Reason code &1.

ANZPRFACT (Analyze Profile Activity) Command Description

ANZPRFACT Command syntax diagram

Purpose

The Analyze Profile Activity (ANZPRFACT) command will determine if user profiles have been inactive for the specified number of days. If a user profile has been inactive for the specified number of days it will be disabled. The last used date on the user profile is used to determine the number of days a profile has been inactive. If the last used date is blank, the restore date is used. If the restore date is blank, the creation date is used.

User profiles can be excluded from this processing by using the Change Active Profile List (CHGACTPRFL) command to add them to the list of profiles that will always be considered active.

It is recommended that you add to this list any profiles that have been created to own application objects and are not used to sign on. You will also want to add any other IBM ("Q") profiles to this list that you do not want disabled. It is not necessary to add any of the profiles in the following list since they will not be considered inactive.

The following user profiles will never be considered inactive:

QAUTPROF	QMSF
QDBSHR	QSECOFR
QDFTOWN	QSNADS
QDIRSRV	QSPL
QDOC	QSPLJOB
QDSNX	QSRV
QFNC	QSRVBAS
QGATE	QSYS
QLPAUTO	QTCP
QLPINSTALL	QTMHHTTP
QNETSPLF	QTMHHTTP1
QNFSANON	QTSTRQS

Restriction: You must have *ALLOBJ, *SECADM, and *JOBCTL special authorities to use this command.

Required Parameters

INACDAYS

Specifies the number of days a user profile can be inactive before it is deleted or disabled. This can be from 1 to 366.

number-of-inactive-days: Specify the number of days from 1 to 366.

Example for ANZPRFACT

ANZPRFACT INACDAYS(30)

User profiles that have been inactive for 30 days or more will be disabled.

Error messages for ANZPRFACT

*ESCAPE Messages

CPFB304

User does not have required special authorities.

ANZPGM (Analyze Programs) Command Description

Note: To use this command, you must have the 5722-PT1 (Performance Tools for iSeries) licensed program installed.

ANZPGM Command syntax diagram

Purpose

The Analyze Programs (ANZPGM) command produces a report that shows the programs and files in a set of libraries and the relationships between them. Both reports are written to the printer file QPPTANZP. Two printer files are produced with the same name. One printer file contains program-to-file cross reference information; the other contains file-to-program cross reference information.

Required Parameter

LIBL Specifies the libraries that contain the programs on which to report.

library-name: Specify up to 10 libraries that contain programs to include in the Analyze Program report.

Optional Parameters

JOB Specifies the job name of a job submitted for batch processing.

Note: If *NONE is specified on the JOBD parameter, this parameter is ignored; job processing is performed interactively.

ANZPGM: The command name is used for the job name.

job-name: Specify the name to be used for any and all batch jobs.

JOBD Specifies the job description used to submit jobs for batch processing.

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.

QEFRJOB: The IBM-supplied job description, QEFRJOB, is used.

job-description-name: Specify the name of an alternative job description.

Other Single Values

***NONE:** A batch job is not submitted; instead, processing continues interactively while the user waits. The user's work station is not available for other use during this time, which could be significant for long jobs.

Example for ANZPGM

```
ANZPGM  LIBL(APPGM ARPGM)
```

This command produces reports showing the program and file relationships for all programs in the Accounts Payable (APPGM) and Accounts Receivable (ARPGM) program libraries.

Error messages for ANZPGM

*ESCAPE Messages

CPF9801

Object &2 in library &3 not found.

CPF9802

Not authorized to object &2 in &3.

PFR9802

Unexpected message monitored.

ANZQRY (Analyze Query) Command Description

ANZQRY Command syntax diagram

Purpose

The Analyze Query (ANZQRY) command allows the user to analyze a query definition (QRYDFN) object for query management conversion problems. Output from this command includes diagnostic messages about potential differences between Query/400 and query management use of query and form information derived from the analyzed QRYDFN object. A completion message shows the highest severity of potential problems that are found.

Required Parameter

QRY Specifies the name of the query definition (QRYDFN) to be analyzed.

The name of the QRYDFN 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.

query-name: Specify the name of the QRYDFN to be analyzed.

Optional Parameter

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.

0: All diagnostic messages about differences are logged.

severity-code-filter: Specify a severity code filter. Valid values range from 0 through 99.

Examples for ANZQRY

Example 1: Displaying All Messages

```
ANZQRY QRY2
```

This command analyzes the first QRYDFN named QRY2 in the user's library list. Messages about conversion problems, for example, text that is too long, are sent to the job log. The messages are displayed when the analysis has completed.

Example 2: Displaying Specific Messages

```
ANZQRY QRY2 99
```

This command analyzes the first query named QRY2 in the user's library list. Only the completion message and messages diagnosing conditions which need to be investigated before a run is attempted are shown and logged.

Error messages for ANZQRY

None

ANZUSROBJ (Analyze User Objects) Command Description

ANZUSROBJ Command syntax diagram

Purpose

The Analyze User Objects (ANZUSROBJ) command collects or reports information for user-created objects on the system. It can be used to determine whether source exists which is likely to have been used in the creation of user objects.

The user must have *ALLOBJ authority.

Required Parameter

OPTION

Specifies whether data collection, data reporting, or both should be done.

The possible option values are:

***COLLECT:** User object data in all user libraries is searched and collected for later analysis.

***REPORT:** User object data is analyzed and reported for the libraries specified (LIB parameter).

Optional Parameters

RPTTYPE

Specifies the types of user object reports that are to be generated.

***ALL:** The system summary, library summary, and library detail user object analysis reports are generated.

***SYSSUM:** The system summary report is generated.

***LIBSUM:** The library summary report is generated.

***LIBDTL:** The library detail report is generated.

LIB Specifies the libraries to be analyzed when generating reports.

» ***ALLUSR:** All user libraries in the auxiliary storage pools (ASPs) defined by the ASPDEV parameter are searched. User libraries are all libraries with names that do not begin with the letter Q except for the following:«

#CGULIB	#DSULIB	#SEULIB
#COBLIB	#RPGLIB	
#DFULIB	#SDALIB	

» Although the following libraries with names that begin with the letter Q are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are also considered user libraries:«

QDSNX	» QSYS2xxxx«	QUSROND
QGPL	QS36F	QUSRPOSGS
QGPL38	QUSER38	QUSRPOSSA
QMPGDATA	QUSRADSM	QUSRPYMSVR
QMOMDATA	QUSRBRM	QUSRDRARS
QMOMPROC	QUSRDIRCL	QUSRSYS
QPFRDATA	QUSRDIRDB	QUSRVI
QRCL	QUSRIJS	QUSRVxRxMx
» QRCLxxxx«	QUSRINFSKR	
» QSYS2«	QUSRNOTES	

Notes:

1. » 'xxxx' is the number of a primary auxiliary storage pool.«
2. A different library name, of the form QUSRVxRxMx, can be created by the user for each release that IBM supports. VxRxMx is the version, release, and modification level of the library.

generic-library-name:* Specify the generic name of the library or group of libraries to be analyzed. To specify a generic library name, add an asterisk (*) at the end of the character string common to the names of all the libraries to analyze. See generic names for additional information.

library-name: Specify the full name of the library to analyze. »

ASPDEV

Specifies the name of the auxiliary storage pool (ASP) device to be analyzed when generating reports.

***ALL:** All ASP devices should be analyzed. This includes the system ASP (ASP 1), basic ASPs (ASPs 2-32), and all independent ASP devices for which data was collected.

***SYSBAS:** The system ASP (ASP 1) and all configured basic ASPs (ASPs 2-32) should be analyzed.

auxiliary-storage-pool-device-name: Specify the device name of the independent ASP that should be analyzed. «

Examples for ANZUSROBJ

Example 1: Collecting Object Information

```
ANZUSROBJ OPTION(*COLLECT)
```

Information for user objects is all user libraries is collected for later analysis.

Example 2: Generate Object Information Reports

```
ANZUSROBJ  OPTION(*REPORT) LIB(MYLIB*)
```

Information previously collected by running ANZUSROBJ with OPTION(*COLLECT) will be analyzed for all libraries with names that begin with 'MYLIB'. System summary, library summary, and library detail user object analysis reports will be generated.

Error messages for ANZUSROBJ

None

ANSLIN (Answer Line) Command Description

ANSLIN Command syntax diagram

Purpose

The Answer Line (ANSLIN) command identifies a communications line that is manually answered by the system operator. This command indicates that the operator manually answered an incoming call and validated the requirements of the caller. When this command is entered, the manual answer sequence is run for the line and, when completed, instructs the operator to select data mode on the modem.

Required Parameter

LINE Specifies the name of the communications line that is being answered.

Example for ANSLIN

```
ANSLIN  LINE(LINE01)
```

This command answers an incoming call on a line named LINE01.

Error messages for ANSLIN

***ESCAPE Messages**

CPF2704

Line description &1 not found.

CPF5914

Answer Line (ANSLIN) command for line &1 failed.

CPF5915

Line &23 not in a valid state for answering.

CPF5917

Not authorized to line description &1.

CPF5919

Line &1 not available.

CPF5935

Error occurred during command processing.

CPF5938

Another job using line &1.

CPF5939

Another job using line &1.

ANSQST (Answer Questions) Command Description

ANSQST Command syntax diagram

Purpose

The question-and-answer (Q & A*) database coordinator uses the Answer Questions (ANSQST) command to display and answer questions asked by users of a Q & A database. More information is available in the Basic System Operations topic in the Information Center.

Restrictions:

1. This command is shipped with public *EXCLUDE authority.
2. A user must have authority to the command and be a Q & A coordinator for any Q & A database referred to by the command.
3. This command is interactive only.

Optional Parameters

QSTDB

Specifies the Q & A database with which to show and answer questions.

***SELECT:** The user is asked to specify a Q & A* database. If only one Q & A database exists on the system, it is the default.

question-database: Specify the name of the Q & A database with which to display and answer questions.

LIB

Specifies the name of the library that contains the Q & A database.

***QSTLIB:** The library containing the specified Q & A* database is searched. If *SELECT is specified on the QSTDB parameter, any Q & A database in any library for which the user is authorized can be selected.

library-name: Specify the name of the library to be searched. If *SELECT is specified on the QSTDB parameter, any database in the library for which the user is authorized can be selected.

Example for ANSQST

ANSQST

This command shows the Select Question Status display. If more than one Q & A database is available for selection, the Select Q and A Database display is shown first.

Error messages for ANSQST

None

APYJRNCHG (Apply Journalized Changes) Command Description

APYJRNCHG Command syntax diagram

Purpose

The Apply Journalized Changes (APYJRNCHG) command applies the changes that have been journalized for a particular journalized object to a saved version of the object to recover it after an operational error or some form of damage. The journalized changes are applied from the specified starting point, either the point

at which an object was last saved or a particular entry on the journal, until the specified ending point has been reached. The ending point can be the point at which the object has had all changes applied, the object was last restored, a specified entry has been reached, a specified time has been reached, or the object was opened or closed by a job (the CMTBDY parameter is used for handling changes that are still pending).

Note: The Display Journal (DSPJRN) command can be used to help determine the desired starting and/or ending points.

A list of journaled objects can be specified. The journaled changes are applied in the order that the journal entries are found on the journal, which is the same order in which the changes were made to the objects.

If an error is found at any point while the journal entries are being applied, the operation ends and the objects might be only partially updated from the journal entries.

Additionally, the command can end when journal entries list operations which cannot be replayed by the command. For example, the command ends when a journal entry is found that indicates one of the following has occurred:

- A physical database file member was reorganized
- A physical database file member was restored
- The system had already applied or removed the changes through the APYJRNCHG command or the Remove Journaled Changes (RMVJRNCHG) command.

See the [Journal management](#) article in the Information Center for a complete listing of the various entries and how they are handled by this command including those entries which can stop the command.

The command also ends on illogical conditions, such as attempts to do the following:

- To add a record to an existing relative record number
- To add a record beyond the next record position after the end of the file
- To add a record that has a duplicate key
- To delete a deleted record
- To update a nonexistent record.

Most illogical conditions are caused by starting the apply journaled changes operation at the wrong place in the journal with respect to the current contents of the objects.



Note: If applying journaled changes ends for one of the objects specified, it ends for all of the objects specified.

If the command ends due to illogical conditions and it is logically possible to restart the apply operation, you can issue the command again specifying a new starting sequence number.

It is possible to apply changes even if the sequence numbers have been reset. The system sends an informational message and continues to apply the changes.

Restrictions:

1. This command is shipped with public *EXCLUDE authority and the QPGMR and QSRV user profiles have private authorities to use the command.
2. The objects specified on this command must currently have their changes journaled and they must have been journaled to the specified journal throughout the period indicated on the command.
3. If a restore operation occurs before the apply operation, the object being restored must have been journaled at the time of the save operation.

4. The objects indicated on the command are allocated exclusively while the changes are being applied. If an object cannot be allocated, the command ends and no journaled changes are applied.
5. If there is no journal entry that corresponds to the period indicated on the command, the command ends and no journaled changes are applied.
6. If the journal sequence numbers have been reset in the range of the receivers specified, and a sequence number is specified on the FROMENT or TOENT parameter, the first occurrence of the sequence number specified on either parameter is used.
7. The TOJOB0 and TOJOB C parameters cannot be used to specify when the apply journaled changes operation is to end if one or more journal receivers in the specified receiver range was attached to a journal  that had a RCVSIZOPT or FIXLENTA option specified that omitted the collection of that data. .
8. The maximum number of objects that can have changes applied with this command is 65,535. If more than 65,535 objects are included in the specifications, an error message is sent and no changes are applied. You can change the values specified on this parameter so that the limit is not exceeded. This limit will include any objects which are created as a result of applying the journaled changes to another object.

Required Parameters

JRN Specifies the qualified name of the journal associated with the journal entries that are applied. The name of the journal 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.

journal-name: Specify the name of the journal associated with the journal entries being applied.

FILE Specifies a maximum of 300 qualified names of physical database files to which journal entries are being applied.

Either the FILE parameter must be specified or the Object Information parameters (OBJ or OBJPATH) must be specified, but not both.

Element 1: File Name

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.

***ALL:** All physical files in the specified library whose changes are journaled to the specified journal have their journal entries applied. The library name must be specified. If *ALL is specified and the user does not have the required authority for all the files in the library, a message is sent and the applying of journal entries ends.

file-name: Specify the name of the physical database file that is to have its journal entries applied.

Element 2: Member Name

The FILE parameter also specifies the name of the member in the file that has its journal entries applied.

***FIRST:** The first member in the file has its journal entries applied.

***ALL:** All members in the file have their journal entries applied.

member-name: Specify the name of the member in the file that has its journal entries applied.

If *ALL is specified for the first part of this parameter, the value specified for the member name is used for all applicable files in the library. For example, if *FIRST is specified, the first member of all applicable files in the library has the changes applied.

OBJ Specifies a maximum of 300 qualified object names to which journal entries are being applied.

Either the FILE parameter must be specified or the Object Information parameters (OBJ or OBJPATH) must be specified, but not both.

Element 1: Object

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.

***ALL:** All objects in the specified library of the specified type whose changes are journaled to the specified journal have their journal entries applied. The library name must be specified. If *ALL is specified and the user does not have the required authority for all objects in the library, a message is sent and the applying of journal entries ends.

object-name: Specify the name of the object that is to have its journal entries applied.

Element 2: Object Type

***FILE:** Entries for database file members are applied.

***DTAARA:** Entries for data areas are applied.

Element 3: Member Name

***FIRST:** The first member in the file has its journal entries applied.

***ALL:** All members in the file have their journal entries applied.

member-name: Specify the name of the member in the file that has its journal entries applied.

If ***ALL** is specified for the first part of this parameter, the value specified for the member name is used for all applicable files in the library. For example, if ***FIRST** is specified, the first member of all applicable files in the library has the changes applied.

Note: If the specified object-type was not ***FILE**, the member name value is ignored.

OBJPATH

Specifies a maximum of 300 object path names to which journal entries are being applied. Only objects whose path name identifies an object of type ***STMF**, ***DIR** or ***SYMLNK** that is in the Root ('/'), QOpensys, and User-defined file systems are supported.

Either the **FILE** parameter must be specified or the Object Information parameters (**OBJ** or **OBJPATH**) must be specified, but not both.

Element 1: Path Name

path-name: Specify the name of the object that is to have its journal entries applied.

In the last component of the path name, an asterisk (*) or a question mark (?) can be used to search for patterns of names. The * tells the system to search for names that have any number of characters in the position of the * character. The ? tells the system to search for names that have a single character in the position of the ? character. Symbolic links within the path name will not be followed. If the path name begins with the tilde (~) character, then the path is assumed to be relative to the appropriate home directory. For more information on specifying path names, refer to path names.

Additional information about path name patterns is in the Integrated file system topic in the File systems and management category of the Information Center.

Element 2: Include or Omit

The second element specifies whether names that match the path name or a pattern should be included or omitted from the operation. Note that in determining whether a name matches a pattern, relative name patterns are always treated as relative to the current working directory.

Note: The **SUBTREE** parameter specifies whether the subtrees are included or omitted.

***INCLUDE:** The objects that match the object name pattern are to be included in determining what journal entries are being applied, unless overridden by an ***OMIT** specification.

***OMIT:** The objects that match the object name pattern are not to be included in determining what journal entries are being applied. This overrides an ***INCLUDE** specification and is intended to be used to omit a subset of a previously selected pattern.

Optional Parameters

SUBTREE

Specifies whether the directory subtrees are included in determining the objects for which journal entries are being applied.

Note: This parameter is only valid if one or more path names were specified on the OBJPATH parameter.

***NONE:** Only the objects that match the selection criteria are processed. The objects within selected directories are not implicitly processed.

***ALL:** All objects that meet the selection criteria are processed in addition to the entire subtree of each directory that matches the selection criteria. The subtree includes all subdirectories and the objects within those subdirectories.

PATTERN

Specifies a maximum of 20 patterns to be used to include or omit objects for which journal entries are being applied.

Note: This parameter is only valid if one or more path names were specified on the OBJPATH parameter.

Element 1: Name Pattern

'*': All objects that match the input OBJPATH parameter are to be included.

name-pattern: Specify the pattern to be used to include or omit objects for which journal entries are being applied. Only the last part of the path name will be considered for the name pattern match. Path name delimiters are not allowed in the name pattern.

If the Name Pattern parameter is not specified the default will be to match all patterns.

For more information on specifying path names, refer to path names.

Additional information about path name patterns is in the Integrated file system topic in the File systems and management category of the Information Center.

Element 2: Include or Omit

The second element specifies whether names that match the pattern should be included or omitted from the operation. Note that in determining whether a name matches a pattern, relative name patterns are always treated as relative to the current working directory.

***INCLUDE:** The objects that match the object name pattern are included in the operation, unless overridden by an *OMIT specification.

***OMIT:** The objects that match the object name pattern are not to be included in the operation. This overrides an *INCLUDE specification and is intended to be used to omit a subset of a previously selected pattern.

RCVRNG

Specifies the starting and ending journal receivers used in applying the journal entries. The system begins by applying the journal entries in the first journal receiver in this journal receiver range and proceeds through the receivers until it applies the journal entries in the last journal receiver in this journal receiver range.

Note:

The maximum number of receivers that can be included in a range of receivers is 256. If more than 256 receivers are included in the range specified, an error message is sent and no changes are applied. You can change the values specified on this parameter so that the limit is not exceeded.

***LASTSAVE:** The range of journal receivers used is determined by the system, as a result of save information for the objects that have their journaled changes applied. This parameter value is only valid if FROMENT(*LASTSAVE) is also specified.

***CURRENT:** The journal receiver that is currently attached when starting to apply journal entries is used.

Element 1: Starting Journal Receiver

The name of the journal receiver 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.

starting-journal-receiver: Specify the name of the journal receiver used as the first (oldest) receiver.

Element 2: Ending Journal Receiver

***CURRENT:** The journal receiver that is currently attached when starting to apply journal entries is used.

The name of the journal receiver 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.

ending-journal-receiver: Specify the name of the journal receiver used as the last (newest) receiver with journal entries to be applied. If the end of the receiver chain is reached before finding this receiver, no entries are applied, and an escape message is sent.

FROMENT

Specifies the entry that is used as the starting point for applying changes that have been journaled.

***LASTSAVE:** The journal entries are applied beginning with the first journal entry after the object was last saved. The system determines the actual starting position for each of the objects specified on the command. The parameter value implies that the object was just restored on the system.

The system verifies information for each object specified, such as if the date and time of the restore is after the date and time of the last save. The system also verifies that the date and time of the saved version of the object that is restored on the system is the same as the date and time that the object was last saved, as indicated on the journal.

If the dates and times do not match, no entries are applied and an inquiry message is sent to the user or system operator requesting a cancel or ignore response. If an ignore response is given to the message, the operation is attempted. A cancel response causes the operation to end, and no journal entries are applied.

If the object was last saved with the save-while-active function, the saved copy of each object includes all changes in the journal entries up to the corresponding start-of-save journal entry. In this case, the system applies changes beginning with the first journal entry following the start-of-save entry.

If the object was last saved when it was not in use (normal save), the saved copy of each object includes all changes in the journal entries up to the corresponding object saved journal entry. In this case, the system applies changes beginning with the first journal entry following the object saved entry.

***FIRST:** The journal entries are applied beginning with the first journal entry in the first receiver supplied in this command.

starting-sequence-number: Specifies the sequence number of the first journal entry that is applied from the journal entries supplied.

TOENT

Specifies the entry used as the ending point for applying changes that are journaled.

***LASTRST:** The journal entries are applied ending with the entry before the object was last restored. The system determines the actual ending position for each of the objects specified on the command. The system verifies that the date and time of the restored version of the object on the system is the same as the date and time that the object was last restored, as indicated on the journal. If the dates and times do not match, no entries are applied and an inquiry message is sent to the user or system operator, requesting a cancel or ignore response. If an ignore response is given to the message, the operation is attempted. A cancel response causes the operation to end, and no journal entries are applied.

This parameter value is valid only if FROMENT(*LASTSAVE) is also specified. If no other TO parameter (TOTIME, TOJOB0, TOJOB1) is specified, TOENT(*LASTRST) is assumed.

***LAST:** Journal entries are applied through the last entry.

ending-sequence-number: Specify the sequence number of the last entry that is applied to the file member.

TOTIME

Specifies the time and date of the last journal entry that was applied to the file member. The first entry with that or the next earlier time is the ending point for applying journal entries. The format of the date must be defined by the job attributes DATFMT and, if separators are used, DATSEP. The time can be entered as 4 or 6 digits (hhmm or hhmmss), where hh = hours, mm = minutes and ss = seconds. If colons are used to separate the time values, the string must be enclosed in apostrophes ('hh:mm:ss').

Element 1: Ending Date

ending-date: Specify the ending date.

Element 2: Ending Time

ending-time: Specify the ending time.

TOJOB0

Specifies the job identifier of the job that, when it opens an object that is specified, ends the applying of journal entries by this command. The first job open entry found for any of the specified objects, is the ending point for all the objects specified.

Only objects of type *FILE, *DIR or *STMF have journal entries related to job opens.

The job identifier has a maximum of three elements.

job-name
user-name/job-name
job-number/user-name/job-name

The job name must be specified. The null value (*N) may be used in place of the job number or the user profile name to maintain the position in the sequence. For example, 123456/*N/job-name specifies the job number, 123456, and the job name, without specifying the user profile under which the job is run.

TOJOB C

Specifies the job identifier of the job that ends the applying of journal entries by this command. The first job close entry found for any of the specified objects, is the ending point for all objects specified. The applying of journal entries is ended when either of the following occurs:

- The specified job closes an object that is specified.
- The specified job is ended.

Only objects of type *FILE, *DIR or *STMF have journal entries related to job closes.

The job identifier has a maximum of three elements.

job-name
user-name/job-name
job-number/user-name/job-name

The job name must be specified. The null value (*N) may be used in place of the job number or the user profile name to maintain the position in the sequence. For example, 123456/*N/job-name specifies the job number, 123456, and the job name, without specifying the user profile under which the job is run.

CMTBDY

Specifies whether commitment boundaries are honored when the journal entries to which journaled changes are to be applied are part of a commitment control logical unit of work (LUW). More information on the use of commitment control is in the Commitment control topic in the Information Center.

Note:

For purposes of this parameter description, the TO option is used to describe either the TOENT, the TOTIME, the TOJOB0, or the TOJOB C parameter, whichever is specified.

***NO:** The journal entries are applied from the entry specified on the FROMENT parameter to the entry indicated on the TO option, regardless of commitment boundaries. Even if a journal entry within this range is a participant of the LUW, the operation is attempted.

***YES:** The journal entries are applied from the entry specified on the FROMENT parameter to the entry indicated on the TO option, honoring commitment boundaries.

- If the journal entry specified on the FROMENT parameter is in the middle of the LUW of which it is a participant, an error message is sent and the operation is not attempted.
- If the journal entry indicated on the TO option is in the middle of the LUW of which it is a participant, the operation stops at the commitment boundary before that journal entry. A diagnostic message is sent at the end of the operation.

Note:

If a journal entry is encountered that causes the operation to end before the entry indicated on the TO option, commitment boundaries might not be honored.

Examples for APYJRNCHG

Example 1: Applying Changes to First Member

```
APYJRNCHG JRN(FIN/JRNACT) FILE(FIN/RCVABLE)
```

This command causes the system to apply to the first member of file RCVABLE in library FIN all changes journaled to JRNACT in library FIN since the file was last saved. The receiver range is determined by the system. The changes are applied beginning with the first journaled change on the receiver chain after the file was last saved and continue through all applicable journal entries to the point at which the file was last restored.

Example 2: Applying Changes to a Specific Member

```
APYJRNCHG JRN(JRNA) FILE((LIB2/PAYROLL JAN))
RCVRNG(RCV22 RCV25) FROMENT(*FIRST) TOENT(*LAST)
```

This command causes the system to apply all changes journaled to JRNA to member JAN of the file PAYROLL in library LIB2. The journal receivers containing the journaled changes are contained in the receiver chain starting with receiver RCV22 and ending with receiver RCV25. Applying the changes starts with the first change journaled on this receiver chain and ends with the last change journaled on this receiver chain. The library search list (*LIBL) is used to find the journal JRNA and the journal receivers RCV22 and RCV25.

Example 3: Applying Changes to IFS Objects

```
APYJRNCHG JRN(JRNS/JRNA) OBJPATH(('HRinfo/payroll/Jan*'
('HRinfo/payroll/JanSummary' *OMIT))
SUBTREE(*ALL) PATTERN(('*.data') ('Temp*.data' *OMIT))
FROMENT(20) TOENT(400)
```

This command causes the system to apply changes to IFS objects. The changes will be applied from starting sequence number 20 to ending sequence number 400 in journal JRNS/JRNA.

1. All objects in the IFS subtree '/HRinfo/payroll' that start with the characters 'Jan', but omitting the object named '/HRinfo/payroll/JanSummary'.
2. All objects in the subtree of any directories that matched number 1, whose names end with '.data', but omitting names ending in '.data' that begin with the characters 'Temp'.

Error messages for APYJRNCHG

*ESCAPE Messages

CPFA0D4

File system error occurred.

CPF70CC

Cannot perform operation beyond journal entry &7.

CPF70CD
Cannot perform operation beyond journal entry &7.

CPF70CE
Cannot perform operation beyond journal entry &7.

CPF70EB
Referential constraint error on member &3.

CPF70EC
Referential constraint error. Reason code &9.

CPF70EE
Maximum encoded vector access paths for member &3.

CPF7002
File &1 in library &2 not a physical file.

CPF7003
Entry not journaled to journal &1. Reason code &3.

CPF7006
Member &3 not found in file &1 in &2.

CPF7007
Cannot allocate member &3 file &1 in &2.

CPF701B
Journal recovery of an interrupted operation failed.

CPF704A
Record length incorrect for member &3.

CPF704F
TOJOB0 or TOJOB C parameter not valid for receiver range.

CPF7041
Entry for job &3/&2/&1 not found.

CPF7042
Object not journaled or journaled to different journal.

CPF7044
Apply or remove of journaled entries failed, reason code &7.

CPF7045
Journal receiver &1 in &2 partially damaged.

CPF7046
Duplicate key not allowed for member &3.

CPF7047
Member &3 file &1 in &2 full.

CPF7048
Cannot perform journaled change to member &3.

CPF7049
Cannot perform operation beyond journal entry &6.

CPF705A
Operation failed due to remote journal.

CPF7050
LASTSAVE date not same as restored version of *&4 object.

CPF7051
Save entry for *&6 object not found in RCVRNG.

CPF7052
Select/omit failure in logical file over member &3.

CPF7053
Values for RCVRNG parameter not correct; reason code &1.

CPF7054
FROM and TO values not valid.

CPF7055
Maximum number of objects exceeded.

CPF7057
*LIBL not allowed with FILE(*ALL) or OBJ(*ALL).

CPF7058
Apply or remove journaled entries operation failed.

CPF7059
Entry for &1 not found in RCVRNG.

CPF7067
FROMENT option not valid. Commit boundary violation.

CPF7068
Entry needed for apply or remove operation not found.

CPF7069
No entries applied or removed using journal &1.

CPF7075
Restore date of *&4 object not same as in journal.

CPF7076
Restore entry for *&6 object not found in RCVRNG.

CPF7077
Key mapping error on member &3.

CPF7078
Cannot apply or remove changes to member &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.

CPF9809
Library &1 cannot be accessed.

CPF9810
Library &1 not found.

CPF9812
File &1 in library &2 not found.

CPF9820
Not authorized to use library &1.

APYPTF (Apply Program Temporary Fix) Command Description

APYPTF Command syntax diagram

Purpose

The Apply Program Temporary Fix (APYPTF) command applies Program Temporary Fixes (PTFs) to a specified product. Before a PTF can be applied, it must be loaded by the Load Program Temporary Fix (LODPTF) command.

When a PTF is applied, it completely replaces the affected objects in the product. PTFs can be applied temporarily or permanently. If they are applied temporarily, the replaced objects are saved by the system and can later be restored to the program by the Remove Program Temporary Fix (RMVPTF) command. If PTFs are applied permanently, the replaced objects are deleted from the system.

The APYPTF command is used to apply immediate PTFs at the time the command is run, or to request PTFs to be applied during the next unattended initial program load (IPL).

During an attended IPL, the Work with PTFs display is used to apply PTFs at the time the system is started. Some IPLs may take longer than others when PTFs are being applied.

Restriction: This command is shipped with public *EXCLUDE authority and the QSRV user profile has private authority to use the command.

Required Parameter

LICPGM

Specifies the 7-character identifier of the product to which the PTFs are applied.

Note:

LICPGM(*ALL) is valid only if SELECT(*ALL) is specified, and OMIT is not specified on this command.

***ALL:** PTFs are applied to all products installed on the system.

licensed-program: Specify the 7-character product identifier to which PTFs are applied.

Optional Parameters

SELECT

Specifies which of the previously loaded PTFs are being applied to specified products. The OMIT parameter cannot be specified if single PTF numbers are specified in the SELECT parameter.

***ALL:** All the PTFs that were loaded are being applied to the program. If all PTFs cannot be applied, messages are sent indicating the PTFs that were not applied and the reasons (for example, required PTFs were not yet applied).

PTF-number: Specify the PTF identification number of each programming fix being applied. Up to 300 PTF numbers can be specified.

OMIT Specifies the PTF numbers that are not applied. Up to 300 PTF numbers can be specified. The OMIT parameter cannot be specified if single PTF numbers are specified in the SELECT parameter.

APY Specifies whether the PTFs are applied on a temporary or permanent basis. Permanently applied fixes cannot be removed; temporarily applied fixes can be removed by the Remove Temporary Fix (RMVPTF) command.

***TEMP:** The PTFs are applied as temporary PTFs.

***PERM:** The PTFs are applied permanently.

DELAYED

Specifies whether immediate PTFs are applied at the time the command is processed or whether immediate or delayed PTFs are applied during the next unattended IPL.

***NO:** Immediate PTFs that are identified are applied at the time the command is processed. Delayed PTFs and immediate PTFs with delayed prerequisite or corequisite PTFs are ignored during the APYPTF request and are not applied. Immediate PTFs which have preconditions that are not satisfied are not applied. A message is sent for each PTF that is not applied.

***YES:** Both delayed and immediate PTFs that are identified are applied during the next unattended IPL. The IPLAPY parameter determines whether the PTFs are applied during the next unattended IPL, or whether any request to apply PTFs during the next unattended IPL is canceled.

***IMMDLY:** All the immediate PTFs are applied and the delayed PTFs or PTFs with delayed prerequisites or corequisites are set to be applied at the next unattended IPL. Any immediate PTFs with preconditions that are not satisfied are set to be applied at the next unattended IPL.

IPLAPY

Specifies the action that is done for delayed or immediate PTFs at the next unattended IPL.

Element 1: PTFs Applied at IPL

This element is valid only if DELAYED(*YES) is also specified.

***YES:** The identified PTFs are applied at the next unattended IPL. The APY parameter determines whether the apply operation is temporary or permanent.

***NO:** Previous requests to apply the identified PTFs at the next unattended IPL are canceled.

Element 2: Licensed Internal Code Fixes Applied

The Licensed Internal Code prerequisites are applied immediately or at the next IPL depending on the value specified on the DELAYED parameter.

***APYPERM:** If LICPGM(*ALL) is specified or APYREQ(*YES) is specified and a product's PTFs have prerequisite Licensed Internal Code fixes, then the required Licensed Internal Code fixes are also identified to be permanently applied.

***NOAPY:** No Licensed Internal Code prerequisite PTFs are identified to be applied.

RLS Specifies the release level of the PTFs being applied. If multiple releases are installed, the release is required.

***ONLY:** This value is only valid when *one* release of the product's base option is installed on the system. PTFs for all installed options of the product are applied regardless of the release-level of the option.

release-level: Specify the release level in VxRyMz format, where Vx is the version number, Ry is the release number, and Mz is the modification level. The variables x and y can be a number from 0 through 9, and the variable z can be a number from 0 through 9 or a letter from A through Z.

APYREQ

Specifies whether the corequisite PTFs of the PTFs specified on the SELECT parameter and their prerequisites within the same product and option are applied with the PTFs specified on the SELECT parameter list.

***NO:** The corequisite and prerequisite PTFs are not applied with the SELECT parameter list. No PTFs are applied if any PTF specified in the list has requisite PTFs not also in the list or already applied. Messages identify the missing requisite PTFs and the PTFs that require them.

***YES:** The PTFs are applied with the SELECT parameter list.

Examples for APYPTF

Example 1: Applying PTFs Temporarily

```
APYPTF LICPGM(5722SS1) DELAYED(*YES)
```

This command applies all the programming fixes that affect the Operating System/400* product (5722SS1). The fixes are temporarily applied at the next IPL.

Example 2: Applying PTFs Permanently

```
APYPTF LICPGM(5722SS1) SELECT(SI00003 SI00008 SI00012)
APY(*PERM) DELAYED(*YES)
```

This command permanently applies PTFs SI00003, SI00008, and SI00012 to the Operating System/400 product in library QSYS at the next IPL.

Example 3: Applying All Loaded PTFs

```
APYPTF LICPGM(*ALL) DELAYED(*IMMDLY)
```

This command permanently applies all PTFs that can be applied immediately and sets the rest to be applied at the next IPL.

Example 4: Applying Immediate PTFs and their Immediate Corequisites and Prerequisites at the Time the Command is Run

```
APYPTF LICPGM(5722SS1) SELECT(SI00003 SI00008 SI00012)
APYREQ(*YES)
```

This command applies the identified PTFs and their corequisites and prerequisites at the time the command is run if the PTFs and their corequisites and prerequisites are defined as immediate. PTFs defined as delayed or defined with corequisites or prerequisites defined as delayed are ignored along with the delayed requisites.

Example 5: Applying PTFs and their corequisites and prerequisites at the next IPL

```
APYPTF LICPGM(5722SS1) SELECT(SI00003 SI00008 SI00012)
DELAYED(*YES) APYREQ(*YES)
```

This command applies the identified PTFs and their corequisites and prerequisites at the next IPL regardless of whether they are defined as delayed or immediate.

Example 6: Applying PTFs and their corequisites and prerequisites as soon as possible

```
APYPTF LICPGM(5722SS1) SELECT(SI00003 SI00008 SI00012)
DELAYED(*IMMDLY) APYREQ(*YES)
```

This command applies the identified PTFs and their corequisites and prerequisites at the time the command is run if the PTFs and their corequisites and prerequisites are defined as immediate. PTFs defined as delayed or defined with corequisites or prerequisites defined as delayed are applied during the next IPL along with the delayed requisites.

Error messages for APYPTF

*ESCAPE Messages

CPF0C4B
Product availability object &2/&1 recovery required.

CPF0C4C
Cannot allocate object &1 in library &2.

CPF0C4D
Error occurred while processing object &1 in library &2.

CPF2150
Object information function failed.

CPF2151
Operation failed for &2 in &1 type *&3.

CPF35AA
Licensed internal code PTF &2 already applied.

CPF35AB
Licensed Internal Code fix &2 not applied.

CPF35A0
Cannot allocate library &1.

CPF35A1
Wrong copy of Licensed Internal Code in use.

CPF35A2
Required hardware changes not installed for PTF &2.

CPF35A3
Licensed Internal Code fix &2 not temporarily applied.

CPF35A5
Licensed Internal Code fix &2 not permanently applied.

CPF35A9
Error occurred while processing Licensed Internal Code fix.

CPF35CF
PTF &1-&2 not applied.

CPF35D0
Licensed Internal Code fix &1-&2 &3 not set to be removed permanently.

CPF35EB
Multiple releases of product &1 installed.

CPF35E3
Interface error detected.

CPF35E4
Information for PTF &1-&2 &3 not complete.

CPF35FA
PTF &1-&2 not applied.

CPF3544
Apply IPL action cannot be removed for PTF &1-&2 &3.

CPF3558
Cannot allocate &1 in &3 type *&2.

CPF3564
PTF &1-&2 damaged.

CPF3583

PTF not applied because error occurred.

CPF3576

Error occurred while applying PTFs for product &1.

CPF3596

PTF numbers in select/omit list not permitted.

CPF3598

PTF function already in process.

CPF3602

PTF &2 not removed because it is permanently applied.

CPF3606

Product &1 &2 not installed.

CPF361D

Apply order of PTFs cannot be determined.

CPF3612

Library &1 not found.

CPF362C

Insufficient storage for Licensed Internal Code fix.

CPF362D

PTF apply completed successfully, but some PTFs need additional actions.

CPF3640

No immediate PTFs applied.

CPF3660

No program temporary fixes identified.

CPF3693

Service function ended because error occurred.

CPF3931

Required programs not found. PTF incomplete.

CPF3945

Records of PTF activity for licensed program are deleted.

CPF8191

Product definition &4 in &9 damaged.

CPF8193

Product load object &4 in &9 damaged.

CPF9845

Error occurred while opening file &1.

CPF9846

Error while processing file &1 in library &2.



APYRMTPTF (Apply Remote Program Temporary Fix) Command Description

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

APYRMTPTF Command syntax diagram

Purpose

The Apply Remote Program Temporary Fix (APYRMTPTF) command allows a service provider to remotely apply PTFs on the service requester's system. When using the APYRMTPTF command, you can request an IPL of the service requester's system. The apply PTF and IPL can be scheduled to occur at a later date and time.

Note: A change request is automatically submitted that can be viewed to determine the status of this command. You can use the Work with Submitted Change Request (WRKSBMCRQ) command to monitor the status.

Restrictions:

1. The apply and IPL functions of this command are only supported when the service requester has the Managed System Services licensed program installed.
2. PTFs that are not marked as delayed are applied immediately. PTFs marked as delayed are scheduled to be applied at the next IPL.
3. PTFs marked as delayed can only be applied permanently if they have been previously applied temporarily. Notice that an IPL would have been required to apply them since they are delayed.
4. Prerequisite PTFs of the same product must already be loaded on the service requester. However, prerequisite PTFs of another product must already be applied on the service requester. Corequisite PTFs of the same product and same option must already be loaded on the service requester. However, corequisite PTFs of another product or another option must already be applied on the service requester.
5. If a NODL value is specified, the node list can only contain entries that have a value of *SNA for the address type.

Required Parameters

PTFID Specifies the PTF that is to be applied. A maximum of 300 PTF identifiers can be specified.

Element 1: PTF Identifier

ptf-identifier: Specify a 7-character PTF identifier.

***ALL**: Applies all PTFs for the specified product.

Element 2: Product Identifier

***ONLY**: Specifies that the PTF identifier is associated with only one product.

product-id: Specify the product of the PTF. The product must be specified when PTF identifiers are not unique across products or the PTF identifier is *ALL.

Element 3: Release Level of Product

release(VxRxMx): Specify the release level of the product. The format is VxRxMx.

DETSRVRQS

Specifies the service requester where this PTF function is to be performed. A single service requester or a list of service requesters can be specified.

Element 1: Network Identifier

***SELECT:** Shows a list of service requesters. From the list, you can select one or more service requesters. *SELECT is not valid when the command is used in a batch environment.

***ALL:** Specifies that this PTF function should occur on all of the service requesters that are defined for this service provider.

***NONE:** The service requesters are identified in a node list object.

***NETATR:** The remote network identifier of the service requester is the same as the remote network identifier of this system.

remote-network-identifier: Specify the remote network identifier of the service requester.

Element 2: Control Point

This element is not valid when the network identifier is *ALL, *SELECT, or *NONE.

remote-control-point: Specify the remote control point name of the service requester.

Optional Parameters

NODL Specifies the node list object name that contains a list of service requesters where this PTF function is to be performed.

***NONE:** The service requesters are identified in the DESTSRVRQS parameter.

The possible library values are:

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

***CURLIB:** Searches for the node list object in the library for the current job.

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

node-list-name: Specify the node list name which contains the list of service requesters where this PTF function is to be performed.

APY Specifies the extent of the change when the PTFs are applied.

***TEMP:** Apply the PTFs temporarily on the service requester.

***PERM:** Apply the PTFs permanently on the service requester.

DLYAPY

Specifies how PTFs are applied. Apply immediate PTFs one at a time while the activity runs on the specified system or later during the next IPL.

***NO:** Applies an immediate PTF at the time the activity runs. If the PTF is marked delayed, it is not applied until the next unattended IPL.

***YES:** Applies both immediate or delayed PTFs during the next unattended IPL.

RMTAPYTIME

Specifies the date and time when this PTF function can occur on the service requester. The current date and time values and next date are determined, when this PTF function is processed on this system, based on this system's date and time.

Element 1: Time Zone

***LCLSYS:** Specifies the remote start time in the time zone of this system.

***MGDSYS:** Specifies the remote start time in the time zone of the service requester.

Element 2: Start After Time

***CURRENT:** Starts this PTF function on the service requester system at any time on or after this PTF function request is processed on this system.

start-after-time: Specify the time when this PTF function can be started on the service requester. The time can be entered as 4 or 6 digits, such as hhmm or hhmmss where hh = hours, mm = minutes, and ss = seconds. Seconds are optional. The time can be specified with or without a 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:** Starts this PTF function on the service requester system on any date on or after the date this PTF function request is processed on this system.

***NEXT:** Starts this PTF function on the service requester system on any date after the date this PTF function is processed on this system.

start-after-date: Specify the date when this PTF function can start on the service requester system. The date must be specified in the job date format.

Note: The special values *CURRENT and *NEXT cannot be specified for the date and time when the time zone value *MGDSYS is specified.

IPL Specifies if an IPL of the service requester's system should occur.

***NO:** An IPL of the service requester's system should not occur.

***YES:** An IPL of the service requester's system should occur after all of the PTFs listed on the command are successfully applied on the service requester.

RMTIPLTIME

Specifies the date and time when the IPL can occur on the service requester. The current date and time values and next date are determined, when this IPL request is processed on this system, based on this system's date and time.

Element 1: Time Zone

***LCLSYS:** Specifies the remote start time in the time zone of this system.

***MGDSYS:** Specifies the remote start time in the time zone of the service requester's system.

Element 2: Start After Time

***CURRENT:** Starts the IPL on the service requester system at any time on or after this IPL request is processed on this system.

start-after-time: Specify the time when the IPL can be started on the service requester. The time can be entered as 4 or 6 digits, such as hhmm or hhmmss where hh = hours, mm = minutes, and ss = seconds. Seconds are optional. The time can be specified with or without a 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:** Starts the IPL on the service requester system on any date on or after the date the IPL request is processed on this system.

***NEXT:** Starts the IPL on the service requester system on any date after the date this IPL request is processed on this system.

start-after-date: Specify the date when the IPL can start on the service requester system. The date must be specified in the job date format.

Note:

The special values *CURRENT and *NEXT cannot be specified for the date and time when the time zone value *MGDSYS is specified.

PWRDWNOPT

Specifies how to manage the end of active jobs during IPL process at the service requester system. This parameter is valid only if IPL(*YES) is specified.

***CNTRLD:** All jobs end when the specified time in the Power down delay (PWRDWNPLY) parameter expires. Any current jobs that are running will perform an end of batch processing.

***IMMED:** Jobs end immediately.

PWRDWNPLY

This parameter is valid only if IPL(*YES) and if PWRDWNOPT(*CNTRLD) are specified. Specifies the amount of time in seconds that the system allows a controlled end to be performed. If the end of job routines are not finished in the specified power down delay, any remaining jobs are ended immediately.

3600: The amount of time in which to complete a controlled end of processing is limited to 3600 seconds.

delay-time: Specify the maximum amount of delay time, in seconds, in which a controlled end can be performed. Valid values range from 1 through 65535 (65,535 seconds).

Examples for APYRMTPTF**Example 1: Applying a Temporary PTF**

```
APYRMTPTF PTFID((123456 *ONLY V3R1M0)) DESTSRVRQS(*SELECT)
NODL(SRVPVLIB/SRVRQSLIST)
```

Apply a PTF temporarily on the selected service requesters system.

Example 2: Applying Permanent PTFs

```
APYRMTPTF PTFID((*ALL PRODUCT1 V5R2M0)) DESTSRVRQS(*NONE)
NODL(SRVPVLIB/SRVRQSLIST)
APY(*PERM) RMTAPYTIME((*MGDSYS (10:00:00 10/31/02))
IPL(*YES)
RMTIPLTIME((*MGDSYS (01:00:00 11/01/02))
```

Apply all PTFs permanently for a specified product, and schedule when the apply and IPL should occur for all of the service requesters in the node list object.

This example applies the PTFs on the service requesters starting at 10:00 a.m. on October 31, 2002. If the PTFs applied successfully, the IPL on the remote system starts at 1:00 a.m. on November 1, 2002. The times specified are in the time zone of the remote system.

Example 3: Applying a Delayed PTF

```
APYRMTPTF PTFID((987654 *ONLY V5R2M0)) DESTSRVRQS(*SELECT)
NODL(SRVPVLIB/SRVRQSLIST)
DLYAPY(*YES) RMTAPYTIME((*MGDSYS (8:00:00 *CURRENT))
```

Apply a delayed PTF temporarily on the selected service requesters system, and schedule when the apply should occur for all of the service requesters in the node list object.

This example applies the PTFs on the service requester systems starting at 8:00 am on the current date. The time specified is in the time zone of the remote system.

Example 4: Doing an IPL after Applying a PTF


```
APYRMTPTF PTFID((*ALL *ONLY V5R2M0)) DESTSRVRQS(*SELECT)
  NODL(SRVPVDLIB/SRVRQSLIST) IPL(*YES)
  RMTIPLTIME((*LCLSYS (*CURRENT 12/24/02))
  PWRDWNOPT(*CNTRLD) PWRDWNPLY(1800)
```

Apply all PTFs temporarily for a specified product, and schedule when the IPL should occur for all of the service requesters in the node list object.

This example applies the PTFs on the service requester systems. If the PTFs are applied successfully, the IPL on the remote system starts at the current time on December 24, 2002. The times specified are in the time zone of the local system.

Example 5: Applying a PTF with two Corequisite PTFs and one Prerequisite PTF

```
APYRMTPTF PTFID((SF00001 *ONLY V5R2M0)) DESTSRVRQS(*SELECT)
  RMTAPYTIME(*LCLSYS (*CURRENT *CURRENT)) IPL(*NO)
```

Apply PTF SF00001. This PTF has two corequisite PTFs and one prerequisite PTF. PTFs SF00002 and SF00003 are corequisite PTFs for PTF SF00001. PTF SF00004 is a prerequisite for PTF SF00001. The system will remove PTFs SF00001, SF00002, SF00003, and SF00004. The system will determine all corequisite PTFs if they were not specified. No IPL will be performed, and this action applies the PTFs on the service requesters. The times specified are in the time zone of the local system.

This example applies the PTFs on the service requesters system starting at 8:00 am on the current date. The time specified is in the time zone of the remote system.

Error messages for APYRMTPTF

*ESCAPE Messages

CPF358A

PTF &1-&2 cannot be applied or removed.



ASKQST (Ask Question) Command Description

ASKQST Command syntax diagram

Purpose

The Ask Question (ASKQST) command shows the Search for Answers display; from this display the user can search for an answer to a question. The user must first search the database to determine if an answer exists before a question can be asked. More information is available in the Basic System Operations topic in the Information Center.

Optional Parameters

QSTDB

Specifies the Question-and-Answer (Q & A) database in which to ask a question.

***SELECT:** The user is asked to specify a Q & A* database. If only one Q & A database exists on the system, it is the default.

question-database: Specify the name of the Q & A database in which to ask a question.

LIB Specifies the name of the library that contains the Q & A database.

The name of the Q & A database can be qualified by one of the following library values:

***QSTLIB:** The library containing the specified Q & A database is used. If *SELECT is specified on the QSTDB parameter, any Q & A database for which the user is authorized can be selected.

library-name: Specify the name of the library to be searched. If *SELECT is specified on the QSTDB parameter, any database for which the user is authorized can be selected.

Example for ASKQST

ASKQST

This command shows the Search for Answers display.

Error messages for ASKQST

None

BCHJOB (Batch Job) Command Description

BCHJOB Command syntax diagram

Purpose

The Batch Job (//BCHJOB) command indicates the beginning of a batch job in a batch input stream. It can also specify different values for the attributes for the job instead of the ones specified in the job description or user profile used for the new job. ➤ The values contained in the job description or in the user profile named in that job description are used for most parameters not coded in the BCHJOB command. Because the spool reader and the batch job must operate in the same name space in order for functions such as syntax checking to operate correctly, the current value of the auxiliary storage pool (ASP) group of the spool reader is used for the initial ASP group parameter of the batch job and the value in the job description is ignored. ⏪

Restrictions:

1. The BCHJOB command cannot be used from a work station.
2. Two slashes must precede this command name when entering it in the data record:
//BCHJOB
3. The user can separate the slashes from this command name with blank spaces, for example:
// BCHJOB.

Optional Parameters

JOB Specifies the name associated with the job when it is processed by the system.

***JOBID:** The simple name of the job description used with this job is the name of the job itself.

job-name: Specify the job's simple name used during processing.

JOBID Specifies the qualified name of the job description used with this job.

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

***LIBL:** All libraries in the library list are searched until the first match is found. If the INLLIBL parameter specifies *JOBID, the library list used to find the job description is the library list for the thread in which the BCHJOB command processing is done. If the

INLLIBL parameter specifies a value other than *JOBDD, the library list used to find the job description consists of the libraries named in the QSYSLIBL system value and the libraries specified by the INLLIBL parameter.

***CURLIB:** If the INLLIBL parameter specifies *JOBDD, the library list used to find the job description is the library list for the thread in which the BCHJOB command processing is done. The current library for the library list is searched. If no library is specified as the current library for the library list, the QGPL library is used. If the INLLIBL parameter specifies a value other than *JOBDD, the QGPL library is used because the library list used to find the job description does not have a current library.

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

QBATCH: The IBM-supplied job description, QBATCH, in the QGPL library is used for the job. (The QGPL library must be in the library list used by the spooling reader that reads the job's input.)

job-description-name: Specify the qualified name of the job description.

JOBQ Specifies the qualified name of the job queue on which this job is placed.

***RDR:** The job queue specified in the start reader or submit jobs command that reads this job is the job queue used.

***JOBDD:** The job queue named in this job description is used.

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

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

***CURLIB:** The current library for the new 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 qualified name of the job queue on which the submitted job is placed.

JOBPTY

Specifies the scheduling priority of the job. Valid values range from 1 through 9, where 1 is the highest priority and 9 is the lowest priority. More information is in Commonly used parameters.

***JOBDD:** The scheduling priority specified in the job description is used for this job.

scheduling-priority: Specify a value, ranging from 1 through 9, that is the scheduling priority for this job.

OUTPTY

Specifies the output priority for spooled files that are produced by this job. The highest priority is 1 and the lowest priority is 9. More information is in Commonly used parameters.

***JOBDD:** The output priority specified in the job description is used for this job.

output-priority: Specify a value, ranging from 1 through 9, for the priority of this job's output files.

PRTTXT

Specifies up to 30 characters of text to be printed at the bottom of each page of output. More information is in Commonly used parameters.

***JOB:** The value specified in the job description is used.

***SYSVAL:** The print text is obtained from the system value QPRTTXT.

***BLANK:** Text is not specified.

'print-text': Specify the character string that is printed at the bottom of each page. Up to 30 characters can be entered, enclosed in apostrophes if necessary.

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 is the process routing step.

QCMDB: The routing data used by the IBM-supplied batch subsystem to route batch jobs to the IBM-supplied control language processor, QCMD, is used.

***JOB:** The routing data used to start the first routing step in the job description used with this job, is used.

***RQSDTA:** The request data (up to 80 characters), specified in the RQSDTA parameter of this command, is used.

'routing-data': Specify the character string that is used as the routing data for starting the first routing step. Up to 80 characters of text can be entered, enclosed in apostrophes if necessary.

RQSDTA

Specifies the request data that is placed as the last entry in this job's message queue. The request data can be a CL command run or a string of characters used by another program. For example, if RTGDTA(QCMDB) is specified, the IBM-supplied batch subsystem, QBATCH, is used, a CL command is supplied, and it becomes a message that is read by the control language processor, QCMD. Or, if a user program is specified in the routing entry, the request data can specify information, such as the record number of the first record in a file processed.

Note:

If a value other than * is specified for this parameter, the data that follows the JOB command is ignored (it is not to be used as request data).

***:** The data following this BCHJOB command is inserted into this job's message queue as request data. For example, the request data may be a group of CL commands that constitute the job.

***JOB:** The request data specified in the job description used by this job is placed as the last entry in this job's message queue.

***NONE:** No request data is placed in the job's message queue.

***RTGDTA:** The routing data in the RTGDTA parameter of this command is placed as the last entry in the job's message queue.

'request-data': Specify the character string that is placed as the last entry in the job's message queue. Up to of 256 characters can be entered, enclosed in apostrophes if necessary. If a CL command is entered, it must be enclosed in single apostrophes, and where apostrophes would normally be used inside the command, double apostrophes must be used.

SYNTAX

Specifies whether requests placed on the job's message queue are checked for syntax as CL commands. When checking for syntax is specified, the commands are checked for syntax when

they are submitted instead of when the job is run, providing an earlier diagnosis of syntax errors. If checking is specified, the message severity that causes a syntax error to end processing of a command is also specified. This parameter is used only if RQSDTA(*) is specified.

***JOB**: The value in the job description used with this job determines whether the request data is checked for syntax and the message severity that is used.

***NOCHK**: The request data for this job is not checked for syntax as CL commands.

message-severity: Specify whether the request data is checked for syntax as CL commands, and, if a syntax error occurs that is equal to or greater than the error message severity specified, the running of the job that contains the command with errors is suppressed. Specify a value, ranging from 00 through 99, for the lowest message severity that causes running of the job to be suppressed. More information on message severity is in Commonly used parameters.

CURLIB

Specifies the name of the library being used as the current library for jobs initiated by this user profile.

***USRPRF**: The current library specified for the job's user profile is used as the current library for the job. The user profile is specified in the job description, which is specified on the JOB parameter.

***CRTDFT**: There is not a current library for the submitted job. If objects are created into the current library, the QGPL library is used as the default current library.

current-library-name: Specify the name of a library used as the current library for this job.

INLLIBL

Specifies the initial user library list that is used to search for any OS/400 system object names that were specified without a library qualifier.

Note:

Duplicating libraries in the library list is not allowed.

***JOB**: The library list in the job description used with this job is used as the initial user library list.

***SYSVAL**: The system default library list is used. It contains the library names that were specified in the system values, QSYSLIBL and QUSRLIBL, at the time that the job is started.

***NONE**: The user portion of the initial user library list is empty; only the system portion is used.

library-name: Specify the names of one or more libraries that are the user portion of the library list and are used by this job. No more than 25 names can be specified; the libraries are searched in the order in which they are listed.

ENDSEV

Specifies the message severity level of escape messages that can cause a batch job to end. The batch job is ended when a request in the batch input stream sends, to the request processing program, an escape message whose severity code is equal to or greater than that specified. (This type of end is considered an abnormal end.) This parameter value is compared with the severity of any unmonitored escape message that occurs as a result of running a noncompiled CL command in a batch job.

***JOB**: The severity limit specified in the job description used with this batch job determines when the job is ended.

message-severity: Specify a value, ranging from 00 through 99, for the message severity of an escape message that results from a request in the batch input stream and that causes the job to

end. Because escape messages sent to users can be up to a severity level of 50, a value of 50 or lower may be specified for a job being ended as a result of an escape message. An unhandled escape message, whose severity is equal to or greater than the value specified, causes the job to end. More information on message severity is in Commonly used parameters.

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: Message level

***JOB**: The value specified for message logging in the job description is used.

message-level: Specify a value, ranging from 0 through 4, that specifies the message logging level used for this job's messages. For more information on the message levels, refer to the *message-level* variable under the CRTJOB command's LOG parameter.

Element 2: Message Severity

***JOB**: The value specified for message logging in the job description is used.

message-severity: Specify a value, ranging from 00 through 99, that is used in conjunction with the logging level to determine which error messages are logged in the job log. More information on message severity is in Commonly used parameters.

Element 3: Message Text Level

***JOB**: The value specified for message logging in the job description is used.

***MSG**: Only message text is written to the job log.

***SECLVL**: Both the message text and the message help (cause and recovery) of the error message are written to the job log.

***NOLIST**: If the job ends normally, no job log is produced. If the job ends abnormally (if the job end code is 20 or higher), a job log is produced. The messages that appear in the job log contain both the message text and the message help.

LOGCLPGM

Specifies whether the commands that can be logged are logged to the job log through the CL program's message queue. This parameter sets the status of the job's logging flag. If *NO is specified, the logging flag status is off and CL commands are not logged. If *YES is specified and the LOG (*JOB) value has been specified in the CRTCLPGM command, all commands in the CL program that can be logged are logged to the job log.

For more information on request logging, refer to the LOG parameter in the CRTCLPGM command description.

***JOB**: The value in the job description is used.

***NO**: The commands in a CL program are not logged to the job log.

***YES**: The commands in a CL program are logged to the job log.

INQMSGRPY

Specifies the way that predefined messages are answered (that is, predefined messages that are sent as a result of running this job). The user can specify that no change is made in the way that predefined messages are answered, that all inquiry messages require a reply, that a default reply be issued, or that the system reply list is checked for a matching reply as each predefined inquiry message is sent. Refer to the Add Reply List Entry (ADDRPYLE) command description for more information.

***JOB**: The inquiry message reply control specified in the job description used with this job is started.

***RQD:** A reply is required by the receiver of the inquiry message for all inquiry messages that occur during the running of this job.

***DFT:** The default reply to the inquiry message is sent. If no default reply is specified in the message description of the inquiry message, the system default reply, *N, is used.

***SYSRPYL:** The system reply list is checked to see if there is an entry for any inquiry message 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.

PRTDEV

Specifies the qualified name of the default printer device for this job. If OUTQ(*DEV) is specified, the file is placed on an output queue with the same name as the printer.

***USRPRF:** The printer device name specified for the job's user profile is used. The user profile is specified in the job description, which is specified on the JOB parameter.

***SYSVAL:** The value specified in the system value QPRTDEV is used.

***JOB:** The printer device name specified in the job description is used.

printer-device-name: Specify the name of the printer device used.

OUTQ Specifies the qualified name of the output queue used for spooled printer files that specify OUTQ(*JOB). This change does not affect files already created in active jobs or files in completed jobs in which the files were spooled.

***USRPRF:** The output queue specified for the job's user profile is used. The user profile is specified in the job description, which is specified on the JOB parameter.

***DEV:** The output queue specified on the PRTDEV parameter is used.

***JOB:** The output queue named in the job description used with this job is the default output queue.

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

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

***CURLIB:** The current library for the new 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.

out-queue-name: Specify the name of the default output queue used by this job.

HOLD Specifies whether jobs using this job description are placed on the job queue in the hold condition. A job placed on the job queue in the hold condition is held until it is either released by the Release Job (RLSJOB) command or canceled by the End Job (ENDJOB) or Clear Job Queue (CLRJOBQ) command. If the job is not run before the next power-down of the system, the job queue can be cleared (and the job ended) when the next initial program load (IPL) is done.

***JOB:** The value specified in the job description determines whether this job is held when it is put on the job queue.

***NO:** The job is not held when it is put on the job queue.

***YES:** The spooled file is held until released by the Release Spool File (RLSSPLF) command.

DATE Specifies the date that is assigned to the job when it is started.

***JOBID:** The date specified in the job description is used.

***SYSVAL:** The value in the QDATE system value at the time the job is started is used.

job-date: Specify the date when the job is started. The value must be entered using the system date format specified by the system value, QDATFMT.

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. For example, if a certain switch is on, another program can be called. The job switches may also be valid in other high-level languages (HLL) programs. Only zeros (off) and ones (on) can be specified in the 8-digit character string.

***JOBID:** The value specified in the job description is the first setting for this job's switches.

switch-settings: Specify any combination of eight zeros and ones that is used as the first switch setting for this job.

MSGQ

Specifies the qualified name of the message queue to which messages are sent.

Note:

If an abnormal ending occurs, the help text of the completion message that is sent specifies the possible causes.

***NONE:** No completion message is sent.

***USRPRF:** The message queue specified on the user profile of the user submitting this job is used.

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

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

***CURLIB:** The current library for the new 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 where the completion message is sent.

SRTSEQ

Specifies the sort sequence table to be used for string comparisons for this job.

***USRPRF:** The sort table specified for the job's user profile is used. The user profile is specified in the job description, which is specified on the JOBID parameter.

***SYSVAL:** The system value QSRTSEQ is used.

***HEX:** A sort sequence table is not used. The hexadecimal values of the characters are used to determine the sort sequence.

***LANGIDUNQ:** A unique-weight sort table is used.

***LANGIDSHR:** A shared-weight sort table is used.

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

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

***CURLIB:** The current library for the new 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.

table-name: Specify the name of the sort sequence table to be used with this job.

LANGID

Specifies the language identifier to be associated with this job. The language identifier is used when *LANGIDUNQ or *LANGIDSHR is specified on the sort sequence prompt (SORTSEQ parameter). If the job CCSID is 65535, this parameter is also used to determine the value of the job default CCSID (DFTCCSID).

***USRPRF:** The language ID specified for the job's user profile is used. The user profile is specified in the job description, which is specified on the JOBID parameter.

***SYSVAL:** The system value QLANGID is used.

language-ID: Specify the language identifier to be used by the job.

CNTRYID

Specifies the country or region identifier to be used by the job.

***USRPRF:** The country or region ID specified for the job's user profile is used. The user profile is specified in the job description, which is specified on the JOBID parameter.

***SYSVAL:** The system value QCNTRYID is used.

country-or-region-ID: Specify the country or region identifier to be used by the job.

CCSID

Specifies the coded character set identifier (CCSID) to be used for the job.

A CCSID is a 16-bit number identifying a specific set of encoding scheme identifiers, character set identifiers, code page identifiers, and additional coding-related information that uniquely identifies the coded graphic representation used.

***USRPRF:** The CCSID specified for the job's user profile is used. The user profile is specified in the job description, which is specified on the JOBID parameter.

***SYSVAL:** The CCSID specified for the QCCSID system value is used.

***HEX:** The CCSID 65535 is used.

coded-character-set-id: Specify the CCSID. More information on valid CCSIDs is in the Globalization topic in the Information Center.

JOBMSGQMX

Specifies the maximum size of the job message queue.

***JOBID:** The value specified in the job description determines the maximum size of the job message queue.

SYSVAL: The value in QJOBMSGQMX (system value), at the time the job is started, determines the maximum size of the job message queue.

maximum-size-of-job-message-queue: Specify a value in the range of 2 to 64 megabytes.

JOBMSGQFL

Specifies the action that should be taken when the job message queue is full.

***JOB**D: The value specified in the job description determines the action that should be taken.

***SYSVAL**: The value specified for the QJOBMSGQFL system value is used.

***NOWRAP**: The message queue does not wrap when it is full. This action ends the job.

***WRAP**: The message queue wraps to the start of the message queue when full and starts filling the message queue again.

***PRTWRAP**: The message queue wraps the job message queue when full and prints the messages that are being overlaid because of wrapping.

Examples for BCHJOB

Example 1: Checking System Reply List for Inquiry Message Entries

```
BCHJOB JOB(DPAYROLL) INQMSGRPY(*SYSRPLY)
```

This command begins the batch job called PAYROLL. An inquiry message that is sent (as a result of running this job) that has an entry in the system reply list is answered according to the reply in that reply list entry. For any inquiry message not represented in the reply list, a reply is required.

The job name is the same as the name of the job description used with the job. **»** The library search list of the thread in which the BCHJOB command is processed determines where the job description PAYROLL is found. The auxiliary storage pool (ASP) group of the thread in which the BCHJOB command is processed is used as the initial ASP group of the new job. The inquiry message control attribute for the job is set to `"*SYSRPLY"`. Values for other job attributes are taken from the job description PAYROLL or from the user profile named in the job description PAYROLL. **«**

Example 2: Setting Job Switches **»**

```
BCHJOB JOB(DQGPL/QBATCH) JOB(DPAYROLL)
      JOBQ(BATCH2) INLLIBL(PAYLIB)
      SWS(00101100) DATE(010188)
```

«

This command begins a batch job called PAYROLL, which is run using attributes from the IBM-supplied job description for batch jobs, QBATCH. The job is placed on the job queue BATCH2. The library PAYLIB is the only library in the user portion of the library list. Switches are set for use in the job, and the date is set at January 1, 1988.

Example 3: Specifying Severity Levels

```
BCHJOB JOB(DCOMPILE) JOBPTY(5)
      SYNTAX(10) INLLIBL(MYCMDS) ENDSEV(40)
```

» This command begins a batch job called COMPILE, which is run using all of the attributes described in the job description also named COMPILE, except for the initial ASP group and the parameters that are changed by this command. **«** The library MYCMDS is the only library in the user portion of the library list to be used when the commands are checked for syntax or run. Syntax errors with a value equal to or greater than 10 end processing of the job. The job is assigned a scheduling priority of 5 and is run as long as no errors are encountered that cause an escape message to be sent that has a severity level of 40 or higher.

Error messages for BCHJOB

*ESCAPE Messages

CALLPRC (Call Bound Procedure) Command Description

CALLPRC Command syntax diagram

Purpose

The Call Bound Procedure (CALLPRC) command calls a bound procedure named on the command, and passes control to it. Optionally, the procedure issuing the CALLPRC command can pass parameters to the called procedure. The CALLPRC command can be used in compiled ILE control language (CL) programs and modules. Upon return, the return code from the called procedure will be placed in the RTNCDE parameter if specified.

Each parameter value passed to the called procedure can be a character string constant, a numeric constant, a logical constant, a floating-point constant, or a CL program variable. If a floating-point constant is specified, the value is converted to double-precision format and passed to the called program. If parameters are passed, the value of the constant or variable is available to the program that is called. Parameters cannot be passed in any of the following forms: lists of values, qualified names, expressions, or keyword parameters. Up to 300 parameters can be passed to the called procedure.

Note:

Although the CALLPRC command will allow up to 300 parameters to be passed, the number that the called procedure can accept will depend on the language of the called procedure. For example, a CL procedure cannot accept more than 40 parameters.

If parameters are passed to a procedure using the CALLPRC command, the values of the parameters are passed in the order in which they appear on the CALLPRC command; this order must match the order in which they appear in the parameter list in the calling procedure.

Parameters in a called procedure can be used in place of its variables. However, no storage in the called procedure is associated with the variables it receives. Instead, if a variable is passed, the storage for the variable is in the procedure in which it was originally declared. If a constant is passed, a copy of the constant is made in the calling procedure and that copy is passed to the called procedure.

The result is that if a variable is passed, the called procedure can change its value and the change is reflected in the calling procedure. If a constant is passed, and its value is changed by the called procedure, the changed value is not known to the calling procedure. Therefore, if the calling procedure calls the same procedure again, the values of constants are set to their original values, but the variables do not change.

Required Parameter

PRC Specifies the name of the procedure being called. The procedure must be in the same program as the calling procedure or in a service program specified at the time the calling program was created. The procedure name may be up to 256 bytes long. The procedure name will be case sensitive. A CL variable cannot be specified for the procedure name.

Optional Parameters

PARM Specifies one or more parameter values that are passed to the called procedure. Each of the values can be specified in only one of the following forms: a character string constant, a numeric constant, logical constant, double-precision floating point constant, or program variable.

***OMIT:** Specifies a null value is passed to another procedure.

parameter-value: Specify one or more parameter values that are passed to the called program. Each of the values can be specified in only one of the following forms: a character string constant, a numeric constant, logical constant, double-precision floating point constant, or program variable.

The type and length of each parameter must be the same in both the calling and called procedures. The order in which they are sent and received must also be the same. The number of parameters specified by the calling procedure does not have to match the number of parameters specified by the called procedure. If the calling procedure specifies more parameters than the called procedure, the extra parameters are ignored. If the calling procedure specifies fewer parameters than the called procedure and the called procedure references the missing parameters, runtime results will be unpredictable.

Parameters can be passed and received as follows:

- Character string constants are neither padded with blanks or null terminated. The operational descriptor for the parameter will have the length of the string. The character string “*OMIT” may not be specified as a constant value; when “*OMIT” is specified a null pointer will be passed to the called procedure.

The called procedure can receive less than the number of bytes passed (in this case, no message is sent). For example, if a procedure specifies that 4 characters are to be received and ABCDEF is passed, only ABCD is accepted and used by the procedure. Quoted character strings can also be passed.

- Decimal constants are passed in packed form and with a length of (15 5), where the value is 15 digits long, of which 5 digits are decimal positions. If a parameter of 12345 is passed, the called procedure must declare the decimal field as (15 5); the parameter is received as 1234500000 (which is 12,345.00000).
- Logical constants are passed as 1 byte with a logical value of 'F1'X or 'F0'X.
- Floating-point literals and floating-point special values (*NAN, *INF, and *NEGINF) are passed as double-precision floating-point numbers in IEEE format, which occupy 8 bytes and are specified in the form $\pm n.n E \pm n$; for example, 2.47E3). A single-precision floating-point number cannot be passed to a called procedure.
- A procedure variable can be passed, in which case the called procedure must declare the field to match the variable defined in the calling procedure. For example, if a CL procedure defines a decimal variable named &CHKNUM as (5 0), the called procedure must declare the field as packed with 5 digits total, with no decimal positions.

If either a decimal constant or a program variable can be passed to the called procedure, the parameter should be defined as (15 5), and any calling procedure must adhere to that definition. If the type, number, order, and length of the parameters do not match between the calling and called procedures (other than the length exception noted previously for character constants), unpredictable results will occur.

- Operational descriptors will always be built for character arguments passed on the PARM parameter. The called procedure can use the information in the descriptor to determine the length of the argument. For character string constants, the length will be the actual length of the constant. For character variables, the length will be the declared length of the variable.

RTNVAL

Specifies the variable to contain the return value from the called procedure. If the value returned by the called procedure is a binary number (types int or short in C/400), you must specify the %BINARY built-in function on the return value parameter.

***NONE**: The called procedure does not return a value.

return-variable-name: The name of the variable that is to contain the return value from the called procedure. This may be either a decimal or a character variable. Variables used as return variables will be aligned on a 16-byte boundary.

Examples for CALLPRC

Example 1: Calling a Procedure

```
CALLPRC PRC(PAYROLL)
```

The procedure named PAYROLL is called with no parameters being passed to it. The PAYROLL procedure does not return a value

Example 2: Defining a Character Constant

```
CALLPRC PRC(PAYROLL) PARM('1')
```

The procedure named PAYROLL is called with a character constant passed as a quoted string. The PAYROLL procedure does not return a value

Example 3: Passing Parameters

```
CALLPRC PRC(PAYROLL) PARM(CHICAGO 1234 &VAR1)
RTNVAL(*NONE)
```

The procedure named PAYROLL. The calling procedure passes three parameters: a character string (CHICAGO), a decimal value (1234.00000), and the contents of the CL variable &VAR1. The attributes of the variable determine the attributes of the third parameter. The PAYROLL procedure does not return a value.

Example 4: Calling Procedure with Floating-Point Values

```
CALLPRC PRC(PRC1) PARM(1.5E3 *INF) RTNVAL(&RVAL)
```

The procedure named PRC1 is called with two double-precision floating-point values being passed to it. The returned value is stored in variable &RVAL.

Example 5: Ignoring the Return Value of a Procedure

```
CALLPRC PRC(PRC1) PARM(1.5E3 *INF) RTNVAL(*NONE)
```

The procedure named PRC1 is called with two double-precision floating-point values being passed to it. The returned value is ignored and therefore unavailable to the calling procedure.

Example 6: Calling a Procedure that Returns a Binary Number

```
CALLPRC PRC(RTNINT) RTNVAL(% BIN($RTNV 1 4))
```

The procedure named RTNINT returns a 4-byte binary value. It is stored in the first four bytes of variable \$RTNV. Variable &RTNV is of type *CHAR and has a length of at least 4.

Error messages for CALLPRC

*ESCAPE Messages

CPF0806

Error found when procedure started.

CALL (Call Program) Command Description

CALL Command syntax diagram

Purpose

The Call (CALL) command calls a program named on the command, and passes control to it. Optionally, the program or user issuing the CALL command can pass parameters to the called program. The CALL


command can be used in batch jobs, in interactive jobs, and in both compiled and interpreted control language (CL). When the called program finishes processing, it can return control to the calling program using the RETURN command.

If the CALL command is issued by a CL program, each parameter value passed to the called program can be a character string constant, a numeric constant, a logical constant, a floating-point constant, or a CL program variable. If a floating-point constant is specified, the value is converted to double-precision format and passed to the called program. If parameters are passed, the value of the constant or variable is available to the program that is called. Parameters cannot be passed in any of the following forms: lists of values, qualified names, expressions, null parameters (that is, a parameter whose value is null, specified by *N), or keyword parameters. >> Up to 99 parameters can be passed to the called program. <<

If parameters are passed to a program using the CALL command, the values of the parameters are passed in the order in which they appear on the CALL command; this order must match the order in which they appear in the parameter list in the calling program.

Parameters in a called program can be used in place of its variables. However, no storage in the called program is associated with the variables it receives. Instead, if a variable is passed, the storage for the variable is in the program in which it was originally declared. If a constant is passed, a copy of the constant is made in the calling program and that copy is passed to the called program.

The result is that if a variable is passed, the called program can change its value and the change is reflected in the calling program. If a constant is passed, and its value is changed by the called program, the changed value is not known to the calling program. Therefore, if the calling program calls the same program again, the values of constants are set to their original values, but the variables do not change.

Information on passing variable parameters using the CALL command within a Submit Job (SBMJOB) command is in the Work Management  book.

Restriction: The user must have object operational authority to the program being called. The user must also have a data authority.

Required Parameter

PGM Specifies the qualified name of the program being called.

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 program being called.

Optional Parameter

PARM Specifies one or more parameter values that are passed to the called program. Each of the values can be specified in only one of the following forms: a character string constant, a numeric constant, logical constant, double-precision floating point constant, or program variable.

The type and length of each parameter must be the same in both the calling and receiving programs. The number of parameters and the order in which they are sent and received must also be the same. If the CALL command is entered interactively or in a noncompiled batch environment, the type and length expected by the called program must match that of each parameter being passed on the command.

Parameters can be passed and received as follows:

- Character string constants of 32 bytes or less are *always* passed with a length of 32 bytes (padded on the right with blanks). If a character constant is longer than 32 bytes, the whole length of the constant is passed. If the parameter is defined to contain more than 32 bytes, the calling program must pass a constant that contains exactly that number of bytes. Constants longer than 32 characters are *not* padded to the length expected by the receiving program. The receiving program can receive less than the number of bytes passed (in this case, no message is sent). For example, if a program specifies that 4 characters are to be received and ABCDEF is passed (padded with blanks in 26 positions), only ABCD is accepted and used by the program. Quoted character strings can also be passed.
- Decimal constants are passed in packed form and with a length of (15 5), where the value is 15 digits long, of which 5 digits are decimal positions. If a parameter of 12345 is passed, the receiving program must declare the decimal field as (15 5); the parameter is received as 1234500000 (which is 12,345.00000).
- Logical constants are passed as 1 byte with a logical value of '1' or '0'.
- Floating-point literals and floating-point special values (*NAN, *INF, and *NEGINF) are passed as double-precision floating-point numbers, which occupy 8 bytes and are specified in the form
n.n E n;

for example, 2.47E3). A single-precision floating-point number cannot be passed to a called program.

- A program variable can be passed if the call is made from a CL program, in which case the receiving program must declare the field to match the variable defined in the calling CL program. For example, if a CL program defines a decimal variable named &CHKNUM as (5 0), the receiving program must declare the field as packed with 5 digits total, with no decimal positions.

If either a decimal constant or a program variable can be passed to the called program, the parameter should be defined as (15 5), and any calling program must adhere to that definition. If the type, number, order, and length of the parameters do not match between the calling and receiving programs (other than the length exception noted previously for character constants), unpredictable results will occur.

The value *N cannot be used to specify a null value because a null value cannot be passed to another program.

Examples for CALL

Example 1: Calling a Program

```
CALL PGM(PAYROLL)
```

The program named PAYROLL is called with no parameters being passed to it. The library list is used to locate the called program.

Example 2: Defining a Character Constant

```
CALL PAYROLL '1'
```

The program named PAYROLL is called with a character constant passed as a quoted string. The program must declare a field of up to 32 characters to receive the constant. The library list is used to locate the called program.

Example 3: Passing Parameters

```
CALL LIB1/PAYROLL (CHICAGO 1234 &VAR1)
```

The program named PAYROLL located in library LIB1 is called. The calling program passes three parameters: a character string (CHICAGO), a decimal value (1234.00000), and the contents of the CL variable &VAR1. The attributes of the variable determine the attributes of the third parameter.

Example 4: Calling Program with Floating-Point Values

```
CALL PGM1 (1.5E3 *INF)
```

The program named PGM1 is called with two double-precision floating-point values being passed to it.

Error messages for CALL

*ESCAPE Messages

CPD0783

Variable &3 for parameter &2 must be TYPE(*DEC), LEN(&4,&5).

CPF0005

Returned command string exceeds variable provided length.

CPF0006

Errors occurred in command.

CPF0805

Error found when program &1 in &2 started.


CPF0806

Error found when procedure started.

CHGACGCDE (Change Accounting Code) Command Description

CHGACGCDE Command syntax diagram

Purpose

The Change Accounting Code (CHGACGCDE) command changes the accounting code of a job. The job can be on a job queue, or it can be active in a subsystem. This command has no effect if the job is on an output queue. If the command is entered when system value QACGLVL indicated that job accounting (*JOB) should be performed when the job entered the system, accounting information is journaled and a new accounting segment is started for the job. If the command is entered when the system value QACGLVL did not indicate job accounting should be performed, the accounting code is changed, but no journal entry is made. More information is in the Work Management  book.

Restrictions:

1. The command must be issued from within the job being changed, or the issuer of the command must be running under a user profile which is the same as the job user identity of the job being changed, or the issuer of the command must be running under a user profile which has job control (*JOBCTL) special authority.

The job user identity is the name of the user profile by which a job is known to other jobs. It is

described in more detail in the Work Management  book.

2. This command is conditionally thread safe. Access will be denied if the target job (either the job that the command is issued in or another job on the system) has secondary threads active. This command may be issued from either the initial thread or a secondary thread of a multi-threaded job if the target job is single threaded.

Optional Parameters

JOB Specifies the name of the job whose accounting code is changed.

A job identifier is a qualified name with up to three elements. For example:

job-name

user-name/job-name

job-number/user-name/job-name

More information is in Commonly used parameters.

*****: The job whose accounting code is changed is the job where this CHGACGCDE command is issued.

job-name: Specify the qualified name of the job whose accounting code is changed. If no job qualifier is given, all jobs currently in the system are searched for the job name. If more than one of the specified name is found, a qualified job name must be specified.

user-name: Specify the name of the user of the job whose accounting code is changed.

job-number: Specify the number of the job whose accounting code is changed.

ACGCDE

Specifies the accounting code used for the job.

***SAME**: The value does not change.

***BLANK**: The accounting code is changed to all blanks.

accounting-code: Specify the 15-character accounting code used for the next accounting segment. The accounting code may contain alphabetic or numeric characters. Blanks may also be used if the accounting code is enclosed in apostrophes.

DUPJOBPT

Specifies the action taken when duplicate jobs are found by this command.

***SELECT**: The selection display is shown when duplicate jobs are found during an interactive session. Otherwise, a message is issued.

***MSG**: A message is issued when duplicate jobs are found.

Example for CHGACGCDE

```
CHGACGCDE JOB(123581/DEPT2/WS1) ACGCDE(123456789)
```

This command changes the accounting code for job WS1, with user profile DEPT2, and job number 123581, to accounting code 123456789 for the next accounting segment. A job resource usage journal entry is written to the system accounting journal, QSYS/QACGJRN.

Error messages for CHGACGDCE

*ESCAPE Messages

CPF1314

Value &1 for parameter &2 not allowed.

CPF1317

No response from subsystem for job &3/&2/&1.

CPF1321

Job &1 user &2 job number &3 not found.

CPF1332

End of duplicate job names.

CPF1334

Must be an interactive job for requested change.

CPF1336

Errors on CHGJOB command for job &3/&2/&1.

CPF1337

&3/&2/&1 not authorized to change parameters.

CPF1340

Job control function not performed.

CPF1341

Reader or writer &3/&2/&1 not allowed as job name.

CPF1343

Job &3/&2/&1 not valid job type for function.

CPF1344

Not authorized to control job &3/&2/&1.

CPF1351

Function check occurred in subsystem for job &3/&2/&1.

CPF1352

Function not done. &3/&2/&1 in transition condition.

CPF180B

Function &1 not allowed.

CHGACTSCDE (Change Activation Schedule Entry) Command Description

CHGACTSCDE Command syntax diagram

Purpose

The Change Activation Schedule Entry (CHGACTSCDE) command allows you to activate a user profile for sign on only for a specific period of time on specific days. A user profile will not be made available at its specified time if it has reached the maximum number of invalid sign-on attempts.

Restriction: You must have *ALLOBJ, *SECADM, and *JOBCTL special authorities to use this command.

Required Parameters**USRPRF**

Specifies the name of the user profile to be activated for a period of time.

ENBTIME

Specifies the time on the specified days at which the job to enable the user profile will be submitted.

***NONE:** The profile is not to be enabled.

enable-time: Specify the time of day that the user profile will be enabled.

DSBTIME

Specifies the time on the specified days at which the job to disable the user profile will be submitted.

***NONE**: The profile is not to be disabled.

disable-time: Specify the time of day that the user profile will be disabled.

Optional Parameter

DAYS Specifies the days of the week on which the job to enable and/or disable the user profile will 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.

Example for CHGACTSCDE

```
CHGACTSCDE  USRPRF(GARRY)  ENBTIME('07:00:00')
             DSBTIME('18:00:00')  DAYS(*MON,*TUE,*WED,*THU,*FRI)
```

The user profile GARRY will be enabled at 7:00 AM on every Monday, Tuesday, Wednesday, Thursday and Friday, and disabled at 6:00 PM on every Monday, Tuesday, Wednesday, Thursday, and Friday. The user profile will remain disabled over the weekend.

Error messages for CHGACTSCDE

***ESCAPE Messages**

CPDB305

User &1 not found in list.

CPFB304

User does not have required special authorities.

CHGACTPRFL (Change Active Profile List) Command Description

CHGACTPRFL Command syntax diagram

Purpose

The Change Active Profile List (CHGACTPRFL) command adds or removes users from the list of profiles that will always be considered active by the Analyze Profile Activity (ANZPRFACT) command. These user profiles will never be disabled even if they have been inactive for the specified number of days.

It is recommended that you add to this list any profiles that have been created to own application objects and are not used to sign on. You will also want to add any other IBM ("Q") user profiles to this list that you do not want disabled. It is not necessary to add any of the user profiles in the following list since they will not be considered inactive.

The following user profiles will never be considered inactive:

QAUTPROF	QMSF
QDBSHR	QSECOFR
QDFTOWN	QSNADS
QDIRSRV	QSPL
QDOC	QSPLJOB
QDSNX	QSRV
QFNC	QSRVBAS
QGATE	QSYS
QLPAUTO	QTCP
QLPINSTALL	QTMHHTTP
QNETSPLF	QTMHHTTP1
QNFSANON	QTSTRQS

Restriction: You must have *ALLOBJ special authority to use this command.

Required Parameter

USRPRF

Specifies the names of the user profiles to be added to or removed from the list of active users.

user-profile-name: Specify up to 10 user profile names to be added to or removed from the list of profiles always considered active by the Analyze Profile Activity (ANZPRFACT) command function.

Optional Parameter

ACTION

Specifies if the user profile is to be added to or removed from the file containing the list of users who will always be considered active and therefore will not be disabled by the Analyze Profile Activity (ANZPRFACT) command function.

***ADD:** The profile is to be added to the list. It will never be considered inactive.

***REMOVE:** The profile will be removed from the list. It will now be considered inactive after the number of days specified for the INACDAYS parameter of the ANZPRFACT command.

Example for CHGACTPRFL

```
CHGACTPRFL USRPRF(JMBLOCK GARRY)
ACTION(*ADD)
```

The user profiles JMBLOCK and GARRY will be added to the list of profiles that are always considered active by the Analyze Profile Activity (ANZPRFACT) command.

Error messages for CHGACTPRFL

*ESCAPE Messages

CPDB305

User &1 not found in list.


CPFB304

User does not have required special authorities.

CHGALRACNE (Change Alert Action Entry) Command Description

CHGALRACNE Command syntax diagram

Purpose

The Change Alert Action Entry (CHGALRACNE) command allows the user to change an action entry in the specified alert filter. More information on alerts is in the Alerts Support  book.

Required Parameters

FILTER

Specifies the qualified name of the filter which contains the action entry being changed.

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 of actions being changed.

***DEFAULT:** The default action entry that was automatically added when the filter was created is changed.

group-name: Specify the name of the group to which the defined actions are to be applied.

Optional Parameters

LOG Specifies whether to log the alert.

***SAME:** The LOG action is not changed.

***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.

***SAME:** The ASNUSER action is not changed.

***NONE:** No user is assigned to the alert.

assigned-user: Specify a user name.

SEND Specifies the destination to which the alert is sent.

***SAME:** The destination does not change.

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

Element 1: Network Identifier

***FOCALPT:** The alert is sent 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 system.

Element 2: Control Point Name

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

SNDDTAQ

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

***SAME**: The data queue does not change.

***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.

***SAME**: The GENTRAP action is not changed.

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

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

Example for CHGALRACNE

```
CHGALRACNE FILTER(MYLIB/MYFILTER) GROUP(CHICAGO)
LOG(*SAME) ASNUSER(CHICAGOOPR)
SEND((*FOCALPT)(*NETATR.MILWKEE)) SNDDTAQ(*SAME)
```

This command changes actions for group CHICAGO to the following:

1. Use the same LOG action.
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. Use the same SNDDTAQ action.
5. Assign the alert to user CHICAGOOPR.

Error messages for CHGALRACNE

***ESCAPE Messages**

CPF812F

Filter damaged.

CPF91DD

Action entry for group &4 not found.

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.

CHGALRD (Change Alert Description) Command Description

CHGALRD Command syntax diagram

Purpose

The Change Alert Description (CHGALRD) command allows the user to change an alert description added previously by the add alert description command. More information on alerts is in the Work Management



book.

Required Parameters

MSGID

Specifies the message ID to which this alert description corresponds.

ALRTBL

Specifies the alert table in which this alert description is to be 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 to be used.

Optional Parameters

ALRTYPE

Specifies the code point for the alert type.

***SAME**: The alert type code point does not change.

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

alert-type-code-point: Specify the alert type code point to be used.

ALRD Specifies the code point for the alert description.

***SAME**: The value does not change.

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

alert-description-code-point: Specify the alert description code point to be used.

PBLCAUSE

Specifies up to 99 code points for probable causes, which are listed in order of decreasing probability.

***SAME**: The probable cause code point does not change.

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

probable-cause-code-point: Specify the probable cause code point to be listed.

CAUSE

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

***SAME**: The value does not change.

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

Element 1: Type of Cause 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 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 ID 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 detailed data. A substitution variable from the corresponding message description can be specified and the message data will be substituted into the alert description when the alert is created. Detailed data identifiers can be specified up to 3 times in each session.

Element 5: Product Identifier

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

***SNDHDW:** Indicates the sender hardware (always iSeries 400).

***SNDSFW:** Indicates the sender software, specified in the PRDID keyword of the Create Alert Table (CRTALRTBL) command.

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

Note:

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

ACTION

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

***SAME:** The value does not change.

***NONE:** There are no 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 ID 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 3 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 detailed data. A substitution variable from the corresponding message description can be specified and the message data will be substituted into the alert description when the alert is created. Detailed data identifiers can be specified up to 3 times in each session.

Element 5: Product Identifier

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

***SNDHDW:** Indicates the sender hardware (always iSeries 400).

***SNDSFW:** The sender software, specified in the PRDID 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 enter either 0 to 3 detailed data identifiers or one product identifier, but not both.

Example for CHGALRD

```
CHGALRD MSGID(USR1234) ALRTBL(USER/USRMSG)
ALRTYPE(*SAME) ALRD(*SAME)
PBLCAUSE(1000 3121 6302)
CAUSE(*SAME) ACTION(*SAME)
```

This command adds probable cause 6302 to the alert description illustrated in the Add Alert Description (ADDALRD) command example.

Error messages for CHGALRD

*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.

CPF1A03

Internal processing error occurred.

CPF2499

Message identifier &1 not allowed.

CPF7BB1

Alert description not found.

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.

CHGALRSLTE (Change Alert Selection Entry) Command Description

CHGALRSLTE Command syntax diagram

Purpose

The Change Alert Selection Entry (CHGALRSLTE) command allows the user to change an alert selection entry that was added previously using the Add Alert Selection Entry (ADDALRSLTE) command. More

information on alerts is in the Alerts Support  book.

Required Parameters

FILTER

Specifies the qualified name of the filter in which the selection entry being changed.

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.

SEQNBR

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

***LAST:** Allows the user to change the last selection. This entry is added automatically when the filter is created and will match any alert.

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

Optional Parameters

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.

***SAME:** The value does not change.

***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 400 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, *vvis* 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, *vvis* 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.
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. 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 changed only if *ALL or *ALLUSR library values can be specified for the name. See generic names for additional information.

GROUP

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

*SAME:

***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 CHGALRSLTE

```
CHGALRSLTE FILTER(MYLIB/MYFILTER) SEQNBR(10)
  SELECT(*SAME) GROUP(NEWSTUFF)
```

Error messages for CHGALRSLTE

*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.

CPF91DC

Selection entry with sequence number &4 not found.

CPF91DE

Filter &1/&2 at maximum size.

CPF91DF

The SELECT keyword cannot be changed for *LAST entry.

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.

CHGALRTBL (Change Alert Table) Command Description

CHGALRTBL Command syntax diagram

Purpose

The Change Alert Table (CHGALRTBL) command is used to change one of the values defined by the Create Alert Table (CRTALRTBL) command. Alert tables define alerts, which are problem notifications in a network. The CHGALRTBL command can be used to change the product identification, product text, or object text for an alert table. The typical user of the CHGALRTBL command is the system or network programmer or operator responsible for network management. More information on alerts is in the Alerts

Support  book.

Required Parameter

ALRTBL

Specifies the name of the alert table in which the alert description is changed.

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.

Optional Parameters

LICPGM

Specifies the licensed program that the alert table is identifying. This program is included in the alert as product identification for the alert sender.

***SAME:** The value does not change.

***NONE:** There is no licensed program for this alert table. This value is allowed for products that do not have a licensed program.

licensed-program: Specify a 7-character identifier for the licensed program. This identifier retrieves release and level information for the program.

Note:

The licensed program is not necessarily an IBM* Licensed Program. Any 7-character identifier that is meaningful to the use of the alerts can be specified (for example, USR1234). If the value is assigned to the system, the identifier and release level information are included in the alert. If the value is not known, then only the identifier and text specified in LICPGMTXT are included in the alert.

LICPGMTXT

Specifies descriptive text for the alert table product identifier (for example, OS/400). This text is included in the alert as product identification for the alert sender.

***SAME:** The value does not change.

***BLANK:** Text is not specified.

'licensed-program-text': Specify up to 30 characters of text, enclosed in apostrophes, describing the licensed program text.

TEXT Specifies the text that briefly describes the alert table. More information is in Commonly used parameters.

***SAME:** The value does not change.

***BLANK:** Text is not specified.

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

Example for CHGALRTBL

```
CHGALRTBL ALRTBL(ALRTBLLIB/ALRTBLNBR1)
  LICPGMTXT('OS/400--customer defined')
```

This command changes the licensed program for the alert table in library ALRTBLLIB called ALRTBLNBR1.

Error messages for CHGALRTBL***ESCAPE Messages****CPF9801**

Object &2 in library &3 not found.

CPF9802

Not authorized to object &2 in &3.

CPF9810

Library &1 not found.

CPF9820

Not authorized to use library &1.

CHGASPA (Change ASP Attribute) Command Description

CHGASPA Command syntax diagram

Purpose

The Change Auxiliary Storage Pool Attributes (CHGASPA) command allows the user to change attributes that control the behavior of an auxiliary storage pool (ASP).

Restrictions: You must have *ALLOBJ special authority to use this command.

Required Parameter

» ASP

Specifies the auxiliary storage pool (ASP) for which the ASP attributes are to be changed. A value must be specified for the ASP parameter or the ASPDEV parameter.

***ALLUSR:** The specified attributes will be changed for all basic ASPs (ASP numbers 2-32) defined. The system ASP (ASP number 1) will not be changed.

auxiliary-storage-pool-number: Specify the ASP for which the specific attribute is to be changed. Valid ASP numbers are 2 to 32. Up to 32 ASP numbers may be specified.

ASPDEV

Specifies the name of the auxiliary storage pool (ASP) device for which the ASP attributes are to be changed. A value must be specified for the ASP parameter or the ASPDEV parameter.

***ALLAVL:** The specified attributes will be changed for all ASP devices that currently have a status of 'Available'.

auxiliary-storage-device-name: Specify the name of the independent ASP device for which the specific attribute is to be changed. Up to 32 ASP device names may be specified. <<

Optional Parameter

CPRRCYPCY

Specifies what the compression recovery policy for the ASP will be. This policy is how the system will handle ASP overflow situations when the ASP contains compressed units. See the Backup and Recovery topic in the Information Center for further detail about the compression recovery policy.

***SAME:** The compression recovery policy does not change if it was previously set. Otherwise, *OVERFLOW is used.

***OVERFLOW:** When the system detects a condition where the ASP capacity is about to be exceeded, data will immediately overflow into the system ASP. The system default compression recovery policy is *OVERFLOW.

***RETRY:** When the system detects a condition where the ASP capacity is about to be exceeded, the system posts SRC A6xx 0277 in the system control panel and waits for space to become available in the ASP. Once space becomes available, the SRC is removed from the system control panel and normal operation will resume for that ASP. If space cannot be made available in the ASP, the SRC is removed from the system control panel and the data overflows into the system ASP. Normal operation will then resume.

***WAIT:** When the system detects a condition where the ASP capacity is about to be exceeded, the system posts an SRC A6xx 0277 in the system control panel and will wait indefinitely for space to become available. Normal operations against this ASP will not resume until the user takes action. Some possible actions that the user could take will include changing the compression recovery policy to allow the ASP to overflow or deleting objects in the ASP.

Examples for CHGASPA

Example 1: Change All User ASPs

```
CHGASPA ASP(*ALLUSR) CPRRCYPCY(*WAIT)
```

This command changes the compression recovery policy for all user auxiliary storage pools to *WAIT.

Example 2: Change Specific ASPs

```
CHGASPA ASP(2 5) CPRRCYPCY(*RETRY)
```

This command changes the compression recovery policy for auxiliary storage pools 2 and 5 to *RETRY.

Example 3: Change Specific ASP Devices

```
CHGASPA ASPDEV(MYASP1) CPRRCYPCY(*RETRY)
```

This command changes the compression recovery policy for auxiliary storage pool ASP device MYASP1.

Error messages for CHGASPA

*ESCAPE Messages

CPF1890

*ALLOBJ authority required for requested operation.

CPF9829

Auxiliary storage pool &2 not found.

CHGATR (Change Attribute) Command Description

CHGATR Command syntax diagram

Purpose

The Change Attribute (CHGATR) command allows a single attribute to be changed for a single object or a group of objects. An object name pattern can be used to change a single attribute for a group of related objects.

The Change Attribute command can also change an attribute of a directory tree where the directory, its contents, and the contents of all of its subdirectories have the attribute changed. A subtree change attribute will attempt to change the attribute for as many objects as possible. A diagnostic message will be sent for each object that could not have its attribute changed and when all of the objects have been attempted, an escape message will be sent. If all of the objects had the attribute changed with no errors, then a completion message will be sent.

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. The user must have execute authority to the directories in the path name prefixes.
2. When doing subtree processing, the user must have read and execute authority to the path name and all subdirectories within that path.
3. For all file systems, except QSYS.LIB >> and independent ASP QSYS.LIB <<, the user must have object management authority to the object when changing the *ALWCKPWRT, *USECOUNT, >> *DISKSTGOPT or *MAINSTGOPT << attributes.
4. For all file systems, except QSYS.LIB >> and independent ASP QSYS.LIB <<, the user must have write authority to the object when changing any attribute, except the *ALWCKPWRT, *USECOUNT, >> *DISKSTGOPT or *MAINSTGOPT << attributes.
5. QSYS.LIB >> and independent ASP QSYS.LIB file systems << require the user to have object operational and object management to change the *USECOUNT attribute if the object type is *FILE, to have execute and object management to change *USECOUNT if the object is a database file member, and to have object management to change *USECOUNT if the object is neither a *FILE or database file member.
6. >> Thread safety. This function will fail when all the following conditions are true:
 - Where multiple threads exist in the job.

- The object on which this function is operating resides in a file system that is not threadsafe. Only the following file systems are threadsafe for this function:
 - Root
 - QOpenSys
 - User-defined
 - QNTC
 - QSYS.LIB
 - Independent ASP QSYS.LIB
 - QOPT <<

Required Parameters

OBJ Specifies the path name of the object or a pattern to match the name of the object to have the attribute changed. The object path name can be either a simple name or a name that is qualified with the name of the directory in which the object is located. A pattern can be specified in the last part of the path name. An asterisk (*) matches any number of characters and a question mark (?) matches a single character. If the path name is qualified or contains a pattern, it must be enclosed in apostrophes.

For more information on specifying path names, refer to path names.

ATR Specifies the attribute to be changed.

***READONLY:** Whether the object can be written to or deleted, have its extended attributes changed or deleted, or have its size changed.

Allowed values for the VALUE parameter are:

***YES:** The object cannot be changed or deleted.

***NO:** The object can be changed or deleted.

***HIDDEN:** Whether the object can be displayed using an ordinary directory list.

Allowed values for the VALUE parameter are:

***YES:** The object is hidden and cannot be displayed using an ordinary directory listing.

***NO:** The object is hidden and cannot be displayed using an ordinary directory listing.

***PCSYSTEM:** Whether the object is a system file and is excluded from normal directory searches.

Allowed values for the VALUE parameter are:

***YES:** The object is a PC system file.

***NO:** The object is not a PC system file.

***PCARCHIVE:** Whether the object has changed since the last time the file was saved or reset by a PC client.

Allowed values for the VALUE parameter are:

***YES:** The object has changed.

***NO:** The object has not changed.

***SYSARCHIVE:** Whether the object has changed and needs to be saved. It is set on when an object's change time is updated, and set off when the object has been saved.

Allowed values for the VALUE parameter are:

***YES:** The object has changed and does need to be saved.

***NO:** The object has not changed and does not need to be saved.

***CCSID:** The code character set identifier (CCSID) of the data and extended attributes of the object. NOTE: Changing the CCSID does not convert the data contained in the object to the new CCSID. Changing the CCSID only changes the value associated with the object. This also applies to the data contained in the extended attributes associated with the object.

Allowed value for the VALUE parameter is:

CCSID: The CCSID of the data and extended attributes of the object.

***ALWCKPWRT:** Whether the stream file (*STMF) can be shared with readers and writers during the save-while-active checkpoint processing. Changing this attribute's current value may cause unexpected results. Please refer to the Backup and Recovery topic in the Information Center for details on this attribute.

Allowed values for the VALUE parameter are:

***YES:** The object can be shared with readers and writers.

***NO:** The object can be shared with readers only.

***USECOUNT:** The count of the number of days an object has been used. Usage has different meanings according to the file system and according to the individual object types supported within a file system. Usage can indicate opening or closing of a file or can refer to adding links, renaming, restoring, or checking out of an object. When this attribute is changed, the count of the number of days used will be reset to zero and the use count date will be set to the current date.

Allowed value for the VALUE parameter is:

***RESET:** The count of the number of days used will be reset to zero and the use count date will be set to the current date.



***DISKSTGOPT:** This determines how auxiliary storage is allocated by the system for the specified object. The option will take effect immediately and be part of the next auxiliary storage allocation for the object. This option can only be specified for stream files in the root (/), QOpenSys and user-defined file systems. This option will be ignored for *TYPE1 byte stream files.

Allowed values for the VALUE parameter are:

***NORMAL:** The auxiliary storage will be allocated normally. That is, as additional auxiliary storage is required, it will be allocated in logically sized extents to accommodate the current space requirement, and anticipated future requirements, while minimizing the number of disk I/O operations. If the *DISKSTGOPT attribute has not been specified for an object, this value is the default.

***MINIMIZE:** The auxiliary storage will be allocated to minimize the space used by the object. That is, as additional auxiliary storage is required, it will be allocated in small sized extents to accommodate the current space requirement. Accessing an object composed of many small extents may increase the number of disk I/O operations for that object.

***DYNAMIC:** The system will dynamically determine the optimum auxiliary storage allocation for the object, balancing space used versus disk I/O operations. For example, if a file has many small extents, yet is frequently being read and written, then future auxiliary storage allocations will be larger extents to minimize the number of disk I/O operations. Or, if a file is frequently truncated, then future auxiliary storage allocations will be small extents to minimize the space used. Additionally, information will be maintained on the stream file sizes for this system and its activity. This file size information will also be used to help determine the optimum auxiliary storage allocations for this object as it relates to the other objects sizes.

***MAINSTGOPT:** This determines how main storage is allocated and used by the system for the specified object. The option will take effect the next time the specified object is opened. This option can only be specified for stream files in the root (/), QOpenSys and user-defined file systems.

Allowed values for the VALUE parameter are:

***NORMAL:** The main storage will be allocated normally. That is, as much main storage as possible will be allocated and used. This minimizes the number of disk I/O operations since the information is cached in main storage. If the *MAINSTGOPT attribute has not been specified for an object, this value is the default.

***MINIMIZE:** The main storage will be allocated to minimize the space used by the object. That is, as little main storage as possible will be allocated and used. This minimizes main storage usage while increasing the number of disk I/O operations since less information is cached in main storage.

***DYNAMIC:** The system will dynamically determine the optimum main storage allocation for the object depending on other system activity and main storage contention. That is, when there is little main storage contention, as much storage as possible will be allocated and used to minimize the number of disk I/O operations. And when there is significant main storage contention, less main storage will be allocated and used to minimize the main storage contention. This option only has an effect when the storage pool's paging option is *CALC. When the storage pool's paging option is *FIXED, the behavior is the same as *NORMAL. When the object is accessed thru a file server, this option has no effect. Instead, its behavior is the same as *NORMAL. <<

VALUE

The value used to change the specified attribute of the object.

***YES:** Allowed for the *READONLY, *HIDDEN, *PCSYSTEM, *PCARCHIVE, *SYSARCHIVE, and *ALWCKPWRT attributes. See the corresponding attribute in the ATR parameter for a description of what this value means for each of the attributes.

***NO:** Allowed for the *READONLY, *HIDDEN, *PCSYSTEM, *PCARCHIVE, *SYSARCHIVE, and *ALWCKPWRT attributes. See the corresponding attribute in the ATR parameter for a description of what this value means for each of the attributes.

***RESET:** Allowed for the *USECOUNT attribute. The count of the number of days used will be reset to zero and the use count date will be set to the current date. >>

***NORMAL:** Allowed for the *DISKSTGOPT and *MAINSTGOPT attributes. See the corresponding attribute in the ATR parameter for a description of what this value means for each of the attributes.

***MINIMIZE:** Allowed for the *DISKSTGOPT and *MAINSTGOPT attributes. See the corresponding attribute in the ATR parameter for a description of what this value means for each of the attributes.

***DYNAMIC:** Allowed for the *DISKSTGOPT and *MAINSTGOPT attributes. See the corresponding attribute in the ATR parameter for a description of what this value means for each of the attributes.

CCSID: Specify a value from 1 through 65535. << Allowed for the *CCSID attribute. The CCSID of the data and extended attributes of the object.

Optional Parameters

SUBTREE

Specifies whether or not to change the specified attribute of the objects within the subtree if the object specified by OBJ is a directory.

***NONE:** The objects specified by OBJ have the attribute changed. If the object is a directory, it has the attribute changed, but its contents do not have the attribute changed.

***ALL:** The objects specified by OBJ have the attribute changed. If the object is a directory, it contents as well as the contents of all of its subdirectories have the attribute changed. NOTE: Pattern matching from the Object (OBJ) parameter only applies to the first level objects. If the first level object is a directory, the pattern matching does not apply to it contents or the contents of its subdirectories.

SYMLNK

If the last component in the path name is a symbolic link, specifies whether or not to change the attribute of the symbolic link or of the object pointed to by the symbolic link.

***NO:** The attribute of the symbolic link object is not changed. The attribute of the object pointed to by the symbolic link is changed.

***YES:** If the object is a symbolic link, the attribute of the symbolic link is changed. The attribute of the object pointed to by the symbolic link is not changed.

Example for CHGATR

Example 1: Change Attribute for a Directory Subtree

```
CHGATR OBJ('/MYINFO')
      ATR(*HIDDEN)
      VALUE(*YES)
      SUBTREE(*ALL)
```

The object MYINFO will have its *HIDDEN attribute changed so it is a hidden object. If MYINFO is a directory, then all of the objects this directory contains as well as all of the objects contained in the subdirectories will have their PC hidden attribute changed because *ALL is specified for the SUBTREE parameter.

Error messages for CHGATR

*ESCAPE Messages

CPFA0AD

Function not supported by file system.

CPFB414

Attributes changed for &1 objects. &2 objects not changed.

CHGAUD (Change Auditing Value) Command Description

CHGAUD Command syntax diagram

Purpose

The Change Auditing Value (CHGAUD) command sets up or changes auditing on an object if you have *AUDIT special authority. Users with *AUDIT special authority can turn auditing on or off for an object regardless of whether they have authority to the object.

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.

Required Parameters

OBJ Specifies the path name of the objects for which auditing values are being changed. See path names for more information on specifying path names.

OBJAUD

Specifies the object auditing value for this object.

***NONE:** Using or changing this object does not cause an audit entry to be sent to the security journal.

***USRPRF:** The user profile of the user accessing this object is used to determine if an audit record is sent for this access. The OBJAUD parameter of the Change User Audit (CHGUSRAUD) command is used to turn auditing on for a specific user.

***CHANGE:** All change accesses to this object by all users are logged.

***ALL:** All change or read accesses to this object by all users are logged.

Example for CHGAUD

```
CHGAUD OBJ('/QSYS.LIB/PAYROLL.LIB/PAYFILE.FILE')
      OBJAUD(*CHANGE)
```

This command changes the object auditing value of the PAYFILE file in the PAYROLL library. The auditing value of the PAYFILE file is changed so that all change access to the file by all users is logged by the system.

Error messages for CHGAUD

*ESCAPE Messages

CPDA080

User profile name too long.

CPE3450

Descriptor not valid.

CPF91EB
Filter type &3 not correct for this operation.

CPFA0AA
Error occurred while attempting to obtain space.

CPFA0AB
Object name not a directory.

CPFA0AC
Request cannot be completed. Directory contains objects.

CPFA0AD
Function not supported by file system.

CPFA0A0
Object name already exists.

CPFA0A1
An input or output error occurred.

CPFA0A2
Information passed to this operation was not valid.

CPFA0A3
Path name resolution causes looping.

CPFA0A4
Too many open files for process.

CPFA0A5
Too many open files.

CPFA0A6
Number of links exceeds maximum allowed for the file system.

CPFA0A7
Path name too long.

CPFA0A9
Object not found.

CPFA0B1
Requested operation not allowed. Access problem.

CPFA0C0
Buffer overflow occurred.

CPFA0DA
Object name is a directory.

CPFA0D4
File system error occurred.

CPFA0D9
Character string not converted.

CPFA0E2
System unable to establish a communications connection to a file server.

CPFA0E4
The communications connection with the file server was abnormally ended.

CPFA0E5
The communications connection with the file server was abnormally ended.

- CPFA0E6**
Object handle rejected by file server.
- CPFA0E7**
System cannot establish a communications connection with a file server.
- CPFA08B**
Path name cannot begin with *.
- CPFA08C**
Pattern not allowed in path name directory.
- CPFA085**
Home directory not found for user &1.
- CPFA086**
Matching quote not found in path name.
- CPFA087**
Path name contains null character.
- CPFA088**
Path name pattern not valid.
- CPFA09C**
Not authorized to object.
- CPFA09D**
Error occurred in program &1.
- CPFA09E**
Object in use.
- CPFA09F**
Object damaged.
- CPFA091**
Pattern not allowed in user name.
- CPFA092**
Path name not converted.
- CPFA093**
Name matching pattern not found.
- CPFA094**
Path name not specified.
- CPF1F05**
Directory handle not valid.
- CPF1F41**
Severe error occurred while addressing parameter list.
- CPF1F4A**
Value for number of directory entries not valid.
- CPF1F53**
Value for length of data buffer not valid.
- CPF22B0**
Not authorized to change the auditing value.
- CPF2203**
User profile &1 not correct.

CPF2225

Not able to allocate internal system object.

CPF2227

One or more errors occurred during processing of command.

CPF223A

Not all objects changed.

CPF9801

Object &2 in library &3 not found.

CPF9802

Not authorized to object &2 in &3.

CPF9803

Cannot allocate object &2 in library &3.

CHGAUT (Change Authority) Command Description

CHGAUT Command syntax diagram

Purpose

The Change Authority (CHGAUT) command is used to change a user's authority for the object named in this command.

Authority can be given to:



- Named users
- *PUBLIC users who do not have authority specifically given to them either for the object or for the authorization list
- The NetWare Inherited Rights filter for the file (used only by the QNetWare file system).
- Groups of users who do not have any authority to the object or are not on the authorization list that secures the object
- Users on an established authorization list

The *AUTL value on the DTAAUT parameter specifies the authority for the following users:

- Users who do not have authority specifically given to them for an object.
- Users who are not on the authorization list that secures the object.
- Users whose groups do not have authority specifically given to it.
- Users whose groups are not on the authorization list that secures the object.

DTAAUT(*AUTL) is allowed only with USER(*PUBLIC). User profiles cannot be secured by an authorization list.

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: If changing authority for an object in the QSYS.LIB  or independent ASP QSYS.LIB  file system:

1. A user must either be the owner of the object or have *ALLOBJ special authority to use this command on an object.
2. This command must get an exclusive lock on a database file before read or object operational authority can be given to a user.

3. If a user requests authority for another specified user to a device currently in use by another authorized user, authority to the device is not given.
4. This command should not be used to change the authority for an authorization list object (/QSYS.LIB/*authorization-list-name*.AUTL).
5. DTAAUT(*AUTL) is only valid with USER(*PUBLIC).
6. Before you give authorities to use a device, controller, or line description, the associated device, controller, or line must be varied on.
7. For display stations or for work station message queues associated with the display station you can either: (1) enter this command at the device for which authorities are being granted or (2) precede this command with the Allocate Object (ALCOBJ) command and follow this command with the Deallocate Object (DLCOBJ) command.

If changing authority for an object in the QLANSRV file system, you cannot specify a value for the AUTL parameter.

Required Parameters

OBJ Specifies the path name of the objects for which specific authorities are given to one or more users or to an authorization list. See path names for more information on specifying path names.

USER Specifies the user names of one or more users to whom authorities for the named object are being given. If user names are specified, the authorities are given specifically to those users.

***PUBLIC:** All users who do not have authority specifically given to them for the object, who are not on the authorization list, whose user group does not have any authority, or whose user group is not on the authorization list, are authorized to use the object as specified in the DTAAUT and OBJAUT parameters.

***NTWIRF:** The NetWare Inherited Rights Filter for the file is authorized to use the object as specified in the DTAAUT and OBJAUT parameters.

Note: This value is used only by the QNetWare file system.

user-profile-name: Specify the user names of one or more users who have specific authority for the object. Up to 50 user profile names can be specified.

DTAAUT

Specifies the data authorities being given to the users specified in the user parameter. If a value other than *SAME is specified, the value replaces any authorities (*OBJOPR, *READ, *ADD, *UPD, *DLT, and *EXECUTE) that the users currently have to the objects.

***SAME:** The users' data authorities to the objects do not change.

***NONE:** The users do not have any of the data authorities to the objects.

***RWX:** The users have *RWX authority to the objects. The users are given *RWX authority to perform all operations on the object except those limited to the owner or controlled by object existence, object management, object alter, and object reference authority. The user can change the object and perform basic functions on the object. *RWX authority provides object operational authority and all the data authorities.

***RX:** The users are given *RX authority to perform basic operations on the object, such as run a program or display the contents of a file. The user is prevented from changing the object. *RX authority provides object operational authority and read and execute authorities.

***RW:** The users are given *RW authority to view the contents of an object and changes the contents of an object. *RW authority provides object operational authority and data read, add, update, and delete authorities.

***WX:** The users are given *WX authority to change the contents of an object and runs a program or searches a library or directory. *WX authority provides object operational authority and data add, update, delete, and execute authorities.

***R:** The users are given *R authority to view the contents of an object. *R authority provides object operational authority and data read authority.

***W:** The users are given *W authority to change the contents of an object. *W authority provides object operational authority and data add, update, and delete authorities.

***X:** The users are given *X authority to run a program or searches a library or directory. *X authority provides object operational authority and data execute authority.

***EXCLUDE:** Exclude authority prevents the user from accessing the object.

***AUTL:** The public authority of the authorization list specified in the AUTL parameter is used for the public authority for the object.

OBJAUT

Specifies the object authorities being given to the users specified in the user parameter. If a value other than *SAME is specified, the value replaces any object authorities (*OBJEXIST, *OBJMGT, *OBJALTER, and *OBJREF) that the users currently have to the objects.

***SAME:** The users' object authorities to the objects do not change.

***NONE:** The users do not have any other object authorities (existence, management, alter, or reference). If *EXCLUDE or *AUTL is specified for the DTAAUT parameter, this value must be specified.

***ALL:** All of the other object authorities (existence, management, alter, and reference) are given to the users.

Or specify up to four (4) of the following values:

***OBJEXIST:** The users have object existence authority to the object.

***OBJMGT:** The users have object management authority to the object.

***OBJALTER:** The users have object alter authority to the object.

***OBJREF:** The users have object reference authority to the object.

AUTL Specifies the name of the authorization list whose users are given authority for the object specified in the OBJ parameter.

***NONE:** The current authorization list is removed from the object.

authorization-list-name: Specify the name of the authorization list to secure this object.

Examples for CHGAUT

Example 1: Changing Authority to All Users

```
CHGAUT  OBJ('/QSYS.LIB/USERLIB.LIB/PROGRAM1.PGM')
        USER(*PUBLIC) DTAAUT(*RW)
```

This command gives authority to use and change the object named PROGRAM1 to all users of the system who do not have authorities specifically given to them, who are not on an authorization list, whose user groups do not have authority to the object, or whose user groups are not on the authorization list. The object is a program (*PGM) located in the library named USERLIB. Because the OBJAUT parameter is not specified, any object authorities *PUBLIC already has remain.

Example 2: Changing Authority to Users on Authorization List

```
CHGAUT  OBJ('/QSYS.LIB/MYLIB.LIB/PRGM3.PGM')
        AUTL(KLIST)
```

This command gives to users the authority specified for them on authorization list KLIST for the object named PRGM3. The object is a program located in library MYLIB.

Error messages for CHGAUT

***ESCAPE Messages**

CPDA080

User profile name too long.

CPE3101

A non-recoverable I/O error occurred.

CPE3408

The address used for an argument was not correct.

CPE3418

Possible APAR condition or hardware failure.

CPE3474

Unknown system state.

CPFA0AA

Error occurred while attempting to obtain space.

CPFA0AB

Object name not a directory.

CPFA0AD

Function not supported by file system.

CPFA0A0

Object name already exists.

CPFA0A1

An input or output error occurred.

CPFA0A2

Information passed to this operation was not valid.

CPFA0A3

Path name resolution causes looping.

CPFA0A4

Too many open files for process.

CPFA0A5

Too many open files.

CPFA0A7

Path name too long.

CPFA0A9

Object not found.

CPFA0B1

Requested operation not allowed. Access problem.

CPFA0C0

Buffer overflow occurred.

CPFA0C1

CCSID &1 not valid.

CPFA0CE

Error occurred with path name parameter specified.

CPFA0DD
Function was interrupted.

CPFA08B
Path name cannot begin with *.

CPFA08C
Pattern not allowed in path name directory.

CPFA085
Home directory not found for user &1.

CPFA086
Matching quote not found in path name.

CPFA087
Path name contains null character.

CPFA088
Path name pattern not valid.

CPFA09C
Not authorized to object.

CPFA09D
Error occurred in program &1.

CPFA09E
Object in use.

CPFA09F
Object damaged.

CPFA091
Pattern not allowed in user name.

CPFA092
Path name not converted.

CPFA093
Name matching pattern not found.

CPFA094
Path name not specified.

CPF1F05
Directory handle not valid.

CPF1F41
Severe error occurred while addressing parameter list.

CPF1F4A
Value for number of directory entries not valid.

CPF1F53
Value for length of data buffer not valid.

CPF2203
User profile &1 not correct.

CPF2225
Not able to allocate internal system object.

CPF223A
Not all objects changed.

CPF9801

Object &2 in library &3 not found.

CPF9802

Not authorized to object &2 in &3.

CPF9803

Cannot allocate object &2 in library &3.

CHGAUTLE (Change Authorization List Entry) Command Description

CHGAUTLE Command syntax diagram

Purpose

The Change Authorization List Entry (CHGAUTLE) command changes the authorities for users on authorization lists. The authorities that the users have on the authorization list are replaced with the authorities specified on the command. The authorization list must already exist and the users must be on the list. If the user specified is not on the list, a message is issued.

The users who can use this command to change the authorization list are: the owner of the authorization list, a user with authorization list management authorities on the list, or a user with all object authority.

When the CHGAUTLE command is used to change a user's authorities, the user must specify the name of the authorization list, a list of users, and a list of authorities. All users specified in the list are given the same authorities. The authorities of each user on the list given to the command are changed to the authorities specified on the command. Authority can be specified for all users who do not have specific authority, who are not on the authorization list, and whose groups have no authority, by giving a user profile name of *PUBLIC.

Restrictions:

1. Only the owner of the list or a user with *ALLOBJ authority can change a user's authorities to include *AUTLMGT.
2. A user with *AUTLMGT authority can change a user's authority. They must also have the specific authority being added or removed.

Required Parameters

AUTL Specifies the name or generic name of the authorization list for which users' authorities are to be changed. The authorization list must already exist.

authorization-list-name: Specify the name of the authorization list used.

generic-authorization-list-name*: Specifies the generic name of the authorization list. 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.

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

***PUBLIC**: Authority is given to all users who have no specific authority, are not on the authorization list, and whose group does not have any authority.

user-ID: Specify a list of user profile names whose authorities are 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 object 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. This parameter is not valid when USER(*PUBLIC) is specified.

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

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

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

***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 the object.

Single Value

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

Example for CHGAUTLE

```
CHGAUTLE AUTL(DEPT48X)
  USER(KARENG KARENS JEFF JULIE DARL)
  AUT(*CHANGE)
```

This command changes the authority that users KARENG, KARENS, JEFF, JULIE, and DARL have on the authorization list to *CHANGE. *CHANGE gives the users object operational authority and all data authorities to the objects secured by the authorization list.

Error messages for CHGAUTLE

*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.

CPF2281

The users specified do not exist on the system.

CPF2283

Authorization list &1 does not exist.

CPF2284

Not authorized to change authorization list &1.

CPF2286

*PUBLIC cannot be given *AUTLMGT authority.

CPF2287

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

CPF2289

Unable to allocate authorization list &1.

CPF2290

*EXCLUDE cannot be specified with another authority.

CHGAJE (Change Autostart Job Entry) Command Description

CHGAJE Command syntax diagram

Purpose

The Change Autostart Job Entry (CHGAJE) command is used to specify a different job description for a previously defined job entry that starts automatically in the specified subsystem description.

Restriction: To use this command, the user must have object operational and object management authorities for the subsystem description and object operational authority for the job description.

Required Parameters

SBSD Specifies the qualified name of the subsystem description containing the job entry being changed that starts automatically.

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 that contains the job entry.

JOB Specifies the simple name that identifies the job entry that starts automatically in the subsystem description whose attributes are being changed.

Optional Parameter

JOB Specifies the qualified name of the job description used for the job that is started by this autostart job entry.

***SAME:** The value does not change.

***SBSD:** The job description having the same qualified name as the subsystem description, specified by the SBSBD parameter, is used for the job that is started automatically.

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 used for the job being started by this job entry that starts automatically. If the job description does not exist when the entry is changed, a library qualifier must be specified because the qualified job description name is retained in the subsystem description.

Example for CHGAJE

```
CHGAJE SBSB(QGPL/PAYROLL) JOB(INIT) JOBD(MANAGER)
```

This command changes the JOBD parameter, for the job entry INIT that starts automatically, to MANAGER. The work entry is in the PAYROLL subsystem description in the QGPL library. The library list is used to locate the job description MANAGER. When the correct library is determined, the qualified job description name is placed in the subsystem description for this autostart job entry.

Error messages for CHGAJE

***ESCAPE Messages**

CPF1619

Subsystem description &1 in library &2 damaged.

CPF1697

Subsystem description &1 not changed.

CHGBCKUP (Change Backup Options) Command Description

CHGBCKUP Command syntax diagram

Purpose

The Change Backup Options (CHGBCKUP) command allows the user to change the options in one of the predefined backups. More information on backup is in the Backup, Recovery, and Availability topic in the Information Center.

Required Parameter

BCKUPOPT

Specifies the backup options to be changed.

***DAILY:** The options for the daily backup are changed.

***WEEKLY:** The options for the weekly backup are changed.

***MONTHLY:** The options for the monthly backup are changed.

Optional Parameters

DEV Specifies the tape device to use for the backup.

***SAME:** The tape device name stored in the specified options is not changed.

tape-device-name: Specify a list of tape device names to use with the specified backup options.

TAPSET

Specifies the name of the tape set to be used.

***SAME:** The tape set name stored in the specified options is not changed.

***ANY:** The tapes mounted on the backup devices are used for the backup. Tape volume IDs are not checked.

tape-volume-set-name: Specify a list of 4-character names of tape volume sets to be rotated for the backup. The tape volume IDs for the backup are generated by concatenating sequential numbers starting with '01' to the specified prefix.

CLRTAP

Specifies whether to clear the tape and start the save at sequence number 1.

***SAME:** The Clear Tape indicator stored in the specified options is not changed.

***YES:** The tape is cleared and the save starts at sequence number 1 (equivalent to CLEAR(*ALL) SEQNBR(1) on the SAVxxx commands).

***NO:** The tape is not cleared and the save starts after the last active file on the tape (equivalent to CLEAR(*NONE) SEQNBR(*END) on the SAVxxx commands).

SBMJOB

Specifies whether to submit the backup as a batch job when the RUNBCKUP menu is used to run a backup using these options.

Note:

This parameter is ignored when the RUNBACKUP command is used to run a backup.

***SAME:** The Submit Job indicator stored in the specified options is not changed.

***YES:** The backup is submitted as a batch job when the menu is used to perform the backup.

***NO:** The backup is run interactively when the menu is used to perform the backup.

CHGONLY

Specifies whether to save only changed objects in the libraries and folders being backed up.

***SAME:** The Save Changed Only indicator stored in the specified options is not changed.

***YES:** Only objects changed since the last backup are saved.

***NO:** All of the objects in the requested libraries and folders are backed up.

PRTRPT

Specifies whether a detailed list of saved objects is printed. A summary report is always printed.

***SAME:** The Print Report indicator stored in the specified options is not changed.

***YES:** A detailed list of saved objects and a summary report are printed.

***NO:** A summary report is printed.

LIB

Specifies which libraries are backed up.

***SAME:** The libraries specified in the options are not changed.

» ***ALLUSR:** User libraries are all libraries with names that do not begin with the letter Q except for the following:«

#CGULIB	#DSULIB	#SEULIB
#COBLIB	#RPLIB	
#DFULIB	#SDALIB	

» Although the following libraries with names that begin with the letter Q are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are also considered user libraries:«

QDSNX	» QSYS2xxxx«	QUSROND
QGPL	QS36F	QUSRPOSGS
QGPL38	QUSER38	QUSRPOSSA
QMPGDATA	QUSRADSM	QUSRPYMSVR
QMQMDATA	QUSRBRM	QUSRRDARS
QMQMPROC	QUSRDIRCL	QUSRSYS
QPFRDATA	QUSRDIRDB	QUSRVI
QRCL	QUSRIJS	QUSRVxRxMx
» QRCLxxxx«	QUSRINFSKR	
» QSYS2«	QUSRNOTES	

Notes:

- » 'xxxx' is the number of a primary auxiliary storage pool.«

2. A different library name, of the form QUSRVxRxMx, can be created by the user for each release that IBM supports. VxRxMx is the version, release, and modification level of the library.

***FROMLIST:** The libraries selected for backup in the library backup list are backed up.

***NONE:** No libraries are backed up.

FLR Specifies which folders are backed up.

***SAME:** The folders specified in the options are not changed.

***ALL:** All folders are backed up.

***FROMLIST:** The folders selected for backup in the folder backup list are backed up.

***NONE:** No folders are backed up.

SECDTA

Specifies whether to save the system security data.

***SAME:** The Security Data indicator stored in the specified options is not changed.

***YES:** Security data is saved when this backup is run.

***NO:** Security data is not saved.

CFG Specifies whether to save the system configuration data.

***SAME:** The Configuration Data indicator stored in the specified options is not changed.

***YES:** Configuration data is saved when this backup is run.

***NO:** Configuration data is not saved.

MAIL Specifies whether to save OfficeVision* mail. This parameter is ignored if FLR(*ALL) is specified.

***SAME:** The Mail indicator stored in the specified options is not changed.

***YES:** Mail is saved when this backup is run.

***NO:** Mail is not saved.

CAL Specifies whether to save OfficeVision calendar data. OfficeVision calendars are also saved when QUSRSYS is saved.

***SAME:** The calendar indicator stored in the specified options is not changed.

***YES:** Calendars are saved when this backup is run.

***NO:** Calendars are not saved.

EXITPGM

Specifies the user program to call before the backup begins and again after the backup is complete.

***SAME:** The program name stored in the specified options is not changed.

***NONE:** No exit program is called.

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 program to call before and after the backup.

Examples for CHGBCKUP

Example 1: Changing the Daily Backup Options

```
CHGBCKUP BCKUPOPT(*DAILY) MAIL(*YES) CAL(*YES)
```

This command changes the daily backup to save the OfficeVision mail and calendar data.

Example 2: Changing the Monthly Backup Options

```
CHGBCKUP BCKUPOPT(*MONTHLY) DEV(TAP01 TAP02)  
TAPSET(RED GRN BLU)
```

This command changes the monthly backup to use tape devices TAP01 and TAP02 and tape sets RED, GRN, and BLU.

Error messages for CHGBCKUP

*ESCAPE Messages

CPF1EEA

Not authorized to library backup list.

CPF1EEB

Not authorized to folder backup list.

CPF1EE0

Device &1 specified more than once.

CPF1EE1

Tape set name &1 specified more than once.

CPF1EE2

Cannot specify *ANY and a tape set name.

CPF1EE3

Not authorized to backup options.

CPF1EE4

Not authorized to run backup.

CPF1EE5

Device &1 not a tape device.

CPF1E6C

Backup options in use.

CPF1E6E

Nothing selected for backup.

CPF1E6F

Tape set name &1 is not valid.

CPF1E67

Backup options and library backup list damaged.

CPF1E99

Unexpected error occurred.

CHGBPA (Change BOOTP Attributes) Command Description

CHGBPA Command syntax diagram

Purpose

The Change BOOTP Server Attributes (CHGBPA) command is used to change the Bootstrap Protocol (BOOTP) attributes. The changes take effect the next time the BOOTP server is started either by the Start TCP/IP (STRTCP) command or by the Start TCP/IP Server (STRTCPSVR) command.

Restriction: You must have *IOSYSCFG special authority to use this command.

Optional Parameters

AUTOSTART

Specifies whether to automatically start the BOOTP server when TCP/IP is started by the STRTCP command. When the BOOTP server is started by the STRTCPSVR command, the AUTOSTART parameter is ignored and the BOOTP server is started regardless of the values of this parameter.

***SAME:** The AUTOSTART value does not change if it was previously set. Otherwise, *NO is used.

***YES:** Start the BOOTP server when the STRTCP command is called.

***NO:** Do not start the BOOTP server when the STRTCP command is called. If you do not intend to use the BOOTP server, set AUTOSTART to *NO.

Example for CHGBPA

```
CHGBPA AUTOSTART(*YES)
```

This command indicates that the next time the STRTCP command is issued to start up TCP/IP and to automatically start the TCP/IP applications, the BOOTP server will be automatically started.

Error messages for CHGBPA

*ESCAPE Messages

None. >>

CHGCCSA (Change Change Control Server Attributes) Command Description

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

CHGCCSA Command syntax diagram

Purpose

The Change Change Control Server Attributes (CHGCCSA) command allows the user to change the attributes value defined to control the operation of the change control server function. This function provides the software distribution support for locally attached change control clients using the Managed System Services licensed program.

Note: If you change the change control server attributes while the managed system functions are active, you must end the program (ENDMGDSYS command) and then start the program again (STRMGDSYS command) before the changes take effect.

Parameters

All parameters are optional.

MSGLOGLVL

Message log level defines the log level that should be used to log messages for change control clients before they establish a connection to the change control server and discover the level configured for them there.

***SAME:** The current value is not changed.

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

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

***DIAG:** This value should only be selected if a collection of logs is being performed to help solve a problem. It logs detailed information about the change control server process.

LOGSIZE

Log file size. Specifies the maximum size in K-bytes of the message log file. When the log file is full, it is automatically backed up, and a new log file is started.

***SAME:** The current value is not changed.

log-size: Specify the size of the amount of data in kilo-bytes (1 kilobyte equals 1024 bytes).

RBATRCSIZE

RBAPI trace file size. Specifies the size in K-bytes of the RBAPI trace file. The RBAPI trace file contains a trace of the Request Block API (RBAPI) operation. When the trace file is full, it is automatically backed up, and a new trace file is started. The Request Block API provides the change control server function to change control clients.

The possible values are:

***SAME:** The current value is not changed.

api-trace-size: Specify the size of the amount of data in kilo-bytes (1 kilobyte equals 1024 bytes).

TRCSIZE

Trace space size. Specifies the maximum size in K-bytes of the internal trace space. The trace space contains a trace of the processing of executable programs. Use this trace value for debugging or diagnostic purposes.

The possible values are:

***SAME:** The current value is not changed.

trace-space: Specifies the size of the amount of data in kilo-bytes (1 kilobyte equals 1024 bytes).

MAXTGT

Maximum targets. Specifies the maximum number of targets allowed (local plus remote).

***SAME:** The current value is not changed.

maximum-number-targets: Specifies a value that corresponds to the number of targets allowed. You can have from 1 to 65335 targets.

MAXCNN

Maximum connections. Specifies the maximum number of simultaneously connected local targets, where *connected* means that the target is in the process of performing a distribution or a change management request.

***SAME:** The current value is not changed.

maximum-number-connections: Specify a numeric value for the maximum number of connections. You can have from 1 to 512 connections. This value cannot be greater than the *maximum number of targets* value.

MAXUSRIFC

Maximum user interfaces. Specifies the maximum number of simultaneously connected user interfaces. A user interface may or may not be configured for each one of the attached change control clients. The decimal number specified here corresponds to the maximum number of user interfaces simultaneously supported by the server. Change control clients with no user interface configured are not considered.

***SAME:** The current value is not changed.

maximum-number-user-interfaces: Specify the maximum number of user interfaces allowed. You can have from 1 to 65435 user interfaces.

TCPPORT

TCP/IP port. The TCP/IP port number of the change control server. It must be a decimal number. The value should match the port specified for the OS/400 change control server service entry. You can use the Work with Service Table Entries (WRKSRVTBLE) command to verify this value.

***SAME:** The current value is not changed.

tcpip-port: Specifies the port number identifying the port of the change control server. Valid values range from 1 through 32767.

AUTMODE

Authorize mode. The authorization mode associated with the system.

***SAME:** The current value is not changed.

***NONE:** No target is authorized to install a change-file or to execute a data file unless it is explicitly authorized with the *auth* command. The *auth* command is a NetView for AIX command. You can use this command to authorize a change file for installation on a target group or to authorize a data file for processing by a target or group. This command must be issued before a change file can be installed.

***ALL:** Any target is authorized to install a change-file or to execute a data file unless it is explicitly unauthorized with the *unauth* command. The *unauth* command is a NetView for AIX command. You can use this command to remove authorization of a catalog entry on a target or group. When you use this command, the target systems cannot view or use the catalog entry.

MAXRQS

Maximum requests. Maximum number of outstanding requests on the change control server.

***SAME:** The current value is not changed.

maximum-request: Specify a value ranging from 1 to 65536.

MAXLCLTGT

Maximum number of local targets.

***SAME:** The current value is not changed.

maximum-local-targets: Specify a value ranging from 1 to 2000.

IDLETIME

Idle time. Time in seconds after which an idle NetView/DM Agent (change control client agent) connection is considered failed.

***SAME:** The current value is not changed.

idle-time: Specify a value ranging from 1 to 32767 seconds.

RTYTIME

Retry time. Time in seconds before a failed NetView/DM Agent (change control client) connection is retried.

***SAME:** The current value is not changed.

retry-time: Specify a value ranging from 1 to 32767 seconds.

MAXLOGIN

Maximum login attempts. Specifies the maximum number of failed attempts to connect to the server before the system temporarily disconnect the access to the server.

***SAME:** The current value is not changed.

maximum-login-attempts: Specify a value ranging from 1 to 5.

STRMGDSYS

Specify if the change control server functions must be enabled for this system. When you specify STRMGDSYS, the change control server functions are started when the Start Managed System Services (STRMGDSYS) command runs. With this command, you can perform managed system functions such as receiving objects, running programs, running commands, and applying program temporary fixes (PTFs).

The possible values are:

***SAME:**

The current value does not change.

***STRCCSJOB**

Enable change control server functions.

***NOSTRCCSJOB:**

Do not enable change control server functions.

AUTOTGTREG

Automatic target registration. This field is composed of two sub-fields, the Automatic Registration and the Registration Mode:

AUTOMATIC REGISTRATION.

This keyword enables clients to automatically register themselves in the server database as one of the server's local targets. The registration is performed the first time a client connects to the server, if the client is not already configured.

***SAME:** The current value is not changed.

***NO:** Does not enable targets to be registered at the server.

***YES:** Enables a target to be registered at the server.

REGISTRATION MODE.

Specify the default mode assigned to targets registered automatically, when the Automatic target registration parameter is YES.

***SAME:** The current value is not changed.

***PULL:** Target is a pull mode target. Change control operations on a pull mode target are controlled by the administrator at a change control server or from a focal or manager target.

***PUSH:** Target is a push mode target. Change control operations on a push mode target are controlled by the administrator at a change control server or from a focal or manager target.

***NOMODE:** There is no mode defined for the target. The target is only used as user interface. This means that the workstation is used by an administrator to do administrative functions on the change control servers in the network or to schedule distributions to targets. A user-interface-only target can be used to run only the software distribution user interfaces. This type of target is useful when you have an environment with more than one change control server. No mode can be configured for user-interface-only targets because these targets cannot receive change control instructions from the same change control server for which they are defined as a user interface.

CCSWRKSTN

CCS workstation name. The name of this system, where the change control server is running. When using TCP/IP as the connection media, the value specified for this parameter must match the local host and domain names specified in the TCP/IP configuration information.

***SAME:** The current value is not changed.

***LCLSNA:** Local control point name and network ID. Control point name and network ID for the local change control server are concatenated to form the CCS workstation name. Control point name and network ID values are taken from the system's local SNA configuration (Network Attributes).

***LCLTCP:** Local host and domain names. Host and domain names for the local change control server (CCS) are concatenated to form the CCS workstation name. Host and domain names are taken from the system's local TCP/IP configuration.

CCS-workstation-name: Specify the change control server's (CCS) workstation name. The CCS workstation name can be up to 64 characters.

DFTUSRPRF

The default user profile used for the change request activity to the change control server. This profile is used to check the authority on the change control server and is also the owner of distributed objects.

The default user profile is used to process the activity.

***SAME:** The current value is not changed.

Name: Specify a default user profile name. This profile is used for object distribution requests.

***SBM:** Specify the user profile of the submitter of the object distribution requests.

ENBTMESVR

Specifies if the change control server can support NetFinity clients.

***SAME:** The current value is not changed.

***YES:** The change control server can support connections from NetFinity clients.

***NO:** The change control server is not enabled to support connections from NetFinity clients.

CODEPAGE

Specifies the number of the change control client's code page for this change control server's domain.

***SAME:** The current value is not changed.

***DFT:** The default code page number is used. The program uses the code page associated with the ASCII CCSID that is the most like the EBCDIC CCSID of the system. If the QCCSID system value is 65535, then the EBCDIC CCSID is taken from the CCSID that corresponds to the QLANGID system value. If the QCCSID system value is not 65535, then the EBCDIC CCSID is taken from the QCCSID system value.

code-page-number: The number of the client's code page.

Examples for CHGCCSA

Example 1: Changing the Maximum Connections, the Maximum Targets and the Registration Mode of the attributes for Change Control Server

```
CHGCCSA MAXTGT(300) MAXCNN(312) AUTOTGTREG(*YES *PUSH)
```

This example indicates 300 targets, 312 connections, and that the registration is in push mode and automatic.

Example 2: Changing the starting of the change control server

```
CHGCCSA STRMGDSYS(*NOSTRCCSJOB)
```

This command specifies that the user does not want the change control server jobs to be started, so there will be no change control server functions running when the user runs the STRMGDSYS command.

Error messages for CHGCCSA

*ESCAPE Messages

MSS0431

Unable to change the change control server attributes.

MSS0437

MAXCNN cannot be greater than MAXTGT nor MAXLCLTGT.

MSS0439

*LCLTCP for parameter CCSWRKSTN cannot be specified.

MSS0441

Value &1 not valid for parameter CCSWRKSTN.

MSS0447

Client code page parameter value was not changed.

MSS0448

Error on the CC Server database.

MSS044A

Number of registered targets has been reached.

CPD0084

&3 not valid for parameter &2.

CPFA0A9

Object not found.



CHGCRQA (Change Change Request Activity) Command Description

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

CHGCRQA Command syntax diagram

Purpose

The Change Change Request Activity (CHGCRQA) command modifies 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 can also be scheduled to run at a date and time in the future.

Restrictions:

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

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 name of the activity to change in the change request description.

***LAST:** The activity will be the last to run in the change request.

activity-ID: A 10 character activity name.

ACTION

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

***SAME:** The action value does not change.

***ACP:** 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 will perform an initial program load (IPL) of a managed iSeries server or activate 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:** Installs the objects previously packaged for installation on the specified managed system or systems. You can install up to seven objects. Only installable objects can be installed. The objects are treated as corequisites, which means that either all installations succeed or all do not. The activity will use 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 are possible.

***RMV:** This 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 executed 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 the objects installed on the specified managed system or systems. Only objects installed through an installable object can be uninstalled.

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 used on that 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.

***SAME:** The value does not change.

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.

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 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 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) or ACTION(*UNINS) 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.

***SAME:** The value does not change.

***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.

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.

***SAME:** The value does not change.

***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 one of the following:

***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.

***SAME:** The value does not change.

***NONE:** The systems on which this activity is 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.

***SAME:** The value does not change.

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

***GEN:** A description is generated based on the action selected. The 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

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

ACTFRC

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

***SAME:** Activation force values do not change.

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: Specify 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.

delay-period: Delay period range is 1-65535.

***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

Specifies which components altered by changes will be used during the activation.

***SAME:** Activation use on activate value does not change.

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

***PROD:** Production version only.

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

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

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 either or both the transfer state or store state parameter specify compressed, this parameter must be present and one (and only one) of the adaptive compression, SNA compression, or user compression algorithms may be requested.

***SAME:** Compression algorithm does not change.

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

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

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

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

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.

***SAME:** SNA compression does not change.

***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.

***SAME:** User compression does not change.

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.

***SAME:** Compression store state value does not change.

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

***COMPRESS:** Store the object in compressed format at the managed system. When this value is specified, the compression algorithm must also be specified. If 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.

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

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.

***SAME:** Compression transfer state value does not change.

***DECOMPRESS:** Transfer the object in decompressed format at 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 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.

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

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, then removability must be *YES.

***SAME:** Activation use on activate value does not change.

***TRIAL:** Trial version only.

***PROD:** Production 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 not, then such action is to be deferred until the next activation. This parameter is only allowed for *SNDINS action.

***SAME:** Alter active components value does not change.

***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.

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

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 test 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.

***SAME:** Auto accept value does not change.

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

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

auto accept: Auto acceptance possible values:

***YES:** Perform automatic acceptance.

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

delay-days: Specifies the number of days the entry point is expected to wait before accepting the object automatically. 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 either installation or a test fails.

***SAME:** Auto remove value does not change.

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

***YES:** Perform automatic removal.

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

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

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.

***SAME:** Pretest value does not change.

***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.

***SAME:** Posttest value does not change.

***YES:** Perform a posttest.

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

***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 them).

***SAME:** Removability value does not change.

***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 when initiating an object.

***SAME:** The parameter value does not change.

parameter data: Specifies the parameters to be passed when starting the program. Each parameter is a 1 to 253 character. The prompt panel will initially allow 25 characters to be entered. By entering an & in position 1, the field will be expanded for larger 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.

***SAME:** Object disposition value does not change.

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

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

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

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 system and managed systems.

***SAME:** Corequisite change name list does not change.

***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 that the object should be replaced if it already exists.

***SAME:** Replace value does not change.

***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, it means that the token is ignored when searching for the correct object. If multiple objects are found matching the tokens specified an error is returned.

***SAME:** The to be deleted name does not change.

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

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

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

***HIGHEST:** The object with the highest token value will have the action performed upon it. The token must be ordered. Useful when a token in a global name is used to indicate a different version of the object and you wish to manipulate the object with the highest version level.

***LOWEST:** The object with the lowest token value will have the action performed upon it. The token must be ordered. Useful when a token in a global name is used to indicate a different version of the object and you wish 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 network ID 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: One of a series of 1-16 character tokens that uniquely identify the object on which the action is to be performed. Characters A-Z and 0-9 may be used. Other special values (@, #, and \$) may 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.

REFLVL

Specifies the level of the software component to be uninstalled.

***SAME:** Refresh level value does not change.

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

refresh-level: Specify a numeric character string of 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 some object in a pending installation execution or in a not terminal status.

***SAME:** Force uninstallation value does not change.

***NO:** Do not allow uninstallation if there are objects, for the software component, in pending installation execution.

***YES:** Allow uninstallation even if there are objects, for the software component, in pending installation execution.

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 the next date values are determined when the activity begins running at the central site systems based on the central site date and time.

Element 1: Time Zone

The time zone of the remote start time.

***SAME:** The value does not change.

***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.

***SAME:** The value does not change.

***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

This is the start after date.

***SAME:** The value does not change.

***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 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.

***SAME:** The value does not change.

***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 run before this activity.

Element 2: Relational Operator

This element is the relational operator to use when comparing the end code from an 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

The 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 can run for that same node even though the conditioning activity specified cannot have completed for all other nodes. In the case where this activity can run for that node, the condition is ignored.

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

STARTIME

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

Element 1: Start After Time

***SAME:** The value does not change.

***CURRENT:** This activity can start 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

***SAME:** The value does not change.

***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.

***SAME:** The value does not change.

***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 data element. This value cannot be specified if the start before date is *CURRENT.

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

***SAME:** The value does not change.

***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.

***SAME:** The value does not change.

***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 CHGCRQA

The following examples illustrate how to use the CHGCRQA 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 by their activity actions:

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

Example 1: Accept Actions

The examples shown here match the first two examples 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(*PROD)
ALTACTIONCOMP(*NOTALLOWED)
AUTOACP((*YES))
ALWRMV(*YES)
COREQCHGNL((REXX PROC UPDATE CONFIG))
```

Instead of installing the objects on the single system MARYPWS1, the objects should be installed on all the managed systems controlled by the change control server ROMSERV1. (All the managed systems are listed in the node list ROMCLIENTS stored in library ROMLIB.)

The activity is changed in the following example so the objects can be installed in a removable manner, if removability is supported on the managed systems.

```

CHGCRQA CRQD(CCLIB/CRQACPRMV)
  ACTIVITY(INSACT01)
  ACTION(*SAME)
  GLBNAME(*SAME)
  NODL(ROMLIB/ROMCLIENTS)
  CPNAME(*NONE)
  ACTUSEINS(*SAME)
  ALTACTIONCOMP(*SAME)
  AUTOACP(( *NO))
  ALWRMV(*DESIRED)
  COREQCHGNL(*SAME)

```

The objects IBM 1234567 PMGRAB UPD 1 2 and REXX PROC UPDATE CONFIG are installed in a removable manner. The objects can be accepted by changing the accept activities. The activities were added to the change request as follows:

```

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))

```

But the activities need to be changed to:

```

CHGCRQA CRQD(CCLIB/CRQACPRMV)
  ACTIVITY(ACCACT01)
  ACTION(*SAME)
  GLBNAME(*SAME)
  NODL(ROMLIB/ROMCLIENTS)
  CPNAME(*NONE)
  COND((INSACT03 *EQ *SUCCESS *ALLNODES))

```

```

CHGCRQA CRQD(CCLIB/CRQACPRMV)
  ACTIVITY(ACCACT02)
  ACTION(*SAME)
  GLBNAME(*SAME)
  NODL(ROMLIB/ROMCLIENTS)
  CPNAME(*NONE)
  COND((INSACT03 *EQ *SUCCESS *ALLNODES))

```

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

```

ADDCRQA CRQD(CCLIB/CRQACPRMV)
  ACTIVITY(INSACT02)
  ACTION(*INS)
  GLBNAME(EURO WORDPROD UPD 2 3 US)
  CPNAME((EUROITAL FREDSW))
  ALWRMV(*YES)

```

Instead of installing the objects on the single system MARYPWS1, the objects should be installed on all the managed systems controlled by the change control server EUROITAL. (All the managed systems are listed in the node list EUROCLIENT stored in library EUROLIB.) To install the objects on all managed systems, the activity is changed to:

```

CHGCRQA CRQD(CCLIB/CRQACPRMV)
  ACTIVITY(INSACT02)
  ACTION(*SAME)

```

```
GLBNAME(*SAME)
NODL(EUROLIB/EUROCLIENT)
CPNAME(*NONE)
ALWRMV(*YES)
```

To accept the object EURO WORDPROD UPD 2 3 US on all the managed systems, the activity ACCACT01 is added to the change request CRQACPRMV:

```
ADDCRQA CRQD(CCLIB/CRQACPRMV)
ACTIVITY(ACCACT01)
ACTION(*ACP)
GLBNAME(EURO WORDPROD UPD 2 3 US)
CPNAME((EUROITAL FREDWS))
COND((INSACT05 *EQ *SUCCESS *SAMENODE))
```

Should be changed as follows:

```
CHGCRQA CRQD(CCLIB/CRQACPRMV)
ACTIVITY(ACCACT01)
ACTION(*SAME)
GLBNAME(*SAME)
NODL(EUROLIB/EUROCLIENT)
CPNAME(*NONE)
COND((INSACT05 *EQ *SUCCESS *ALLNODES))
```

Example 2: Activate Actions

- Change an activate activity to use trial a production version during the activation.

```
CHGCRQA CRQD(CCLIB/CRQINSACT)
ACTIVITY(ACTACT01)
ACTION(*ACT)
CPNAME((ROMSERV1 MARYPWS1))
ACTFRC(*NO *HOURS 6)
ACTUSEACT(*BOTH)
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 will wait up to six hours before proceeding with the activation if the system is still active.

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

The delay units and delay period are ignored because activation force is *YES.

```
CHGCRQA CRQD(CCLIB/CRQINSACT)
ACTIVITY(ACTACT02)
ACTION(*ACT)
CPNAME((EUROITAL FREDWS))
ACTFRC((*YES *MINUTES 30))
RMTSTRIME(*MGDSYS (23:00:00 04/15/02))
```

Example 3: Delete Actions

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

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

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

```

CHGCRQA CRQD(CCLIB/CRQSNDDL)
ACTIVITY(DLTACT02)
ACTION(*ACT)
GLBNAME(EURO SPELLCHECK TEST *ANY)
CPNAME((ROMSERV1 MARYPWS1)
        (ROMSERV1 MARYPWS2)
        (ROMSERV1 MARYPWS3)
        (ROMSERV1 MARYPWS4))
COND((SNDACT01 *EQ *SUCCESS *ALLNODES))

```

This activity occurs only if the object EURO SPELLCHECK EXE 1 US is successfully sent to those systems:

```

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

```

Example 5: Run Actions

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

```

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

```

- Change 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.

```

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

```

Example 6: Install Actions

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

- The install activity is:

```

ADDCRQA CRQD(CCLIB/CRQACPRMV)
ACTIVITY(INSACT01)
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))

```


Instead of installing the objects on the single system MARYPWS1, the objects should be installed on all the managed systems controlled by the change control server ROMSERV1. (All the managed systems are listed in the node list ROMCLIENTS stored in library ROMLIB.) To install the objects in a removable manner, the install activity is changed to:

```
CHGCRQA CRQD(CCLIB/CRQACPRMV)
  ACTIVITY(INSACT01)
  ACTION(*SAME)
  GLBNAME(*SAME)
  NODL(ROMLIB/ROMCLIENTS)
  CPNAME(*NONE)
  ACTUSEINS(*SAME)
  ALTACTCOMP(*SAME)
  AUTOACP(( *NO))
  ALWRMV(*DESIRED)
  COREQCHGNL(*SAME)
```

- The install activity is:

```
ADDCRQA CRQD(CCLIB/CRQACPRMV)
  ACTIVITY(INSACT02)
  ACTION(*INS)
  GLBNAME(EURO WORDPROD UPD 2 3 US)
  CPNAME((EUROITAL FREDSW))
  ALWRMV(*YES)
```

Instead of install the objects on the single system MARYPWS1, the objects should be installed on all the managed systems controlled by the change control server EUROITAL. (All the managed systems are listed in the node list EUROCLIENT stored in library EUROLIB.) To install the objects on all the managed systems, the install activity is changed to:

```
CHGCRQA CRQD(CCLIB/CRQACPRMV)
  ACTIVITY(INSACT02)
  ACTION(*SAME)
  GLBNAME(*SAME)
  NODL(EUROLIB/EUROCLIENT)
  CPNAME(*NONE)
  ALWRMV(*YES)
```

- Change 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.

```
CHGCRQA CRQD(CCLIB/CRQINS)
  ACTIVITY(INSACT03)
  ACTION(*INS)
  GLBNAME(IBM 1234567 PMGRAB UPD 1 2)
  CPNAME((ROMSERV1 MARYPWS1))
  ACTUSEINS(*TRIAL)
  ALWRMV(*YES)
```

- Change 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.

```
CHGCRQA CRQD(CCLIB/CRQINS)
  ACTIVITY(INSACT04)
  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))
```

- Change 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 objects are installed to the active area in a removable manner. The installation is not automatically accepted. The installation is scheduled for 3 p.m. on 1 January 2002.

```
CHGCRQA CRQD(CCLIB/CRQRTVINS)
  ACTIVITY(INSACT05)
  ACTION(*INS)
  GLBNAME(EURO WORDPROC REF 2 US)
  CPNAME((EUROITAL FREDSSWS))
  ALTACTCOMP(*ALLOWED)
  COREQCHGNL(EURO WORDPROC UPD 2 3 US)
  RMTSTRTIME(*MGDSYS (15:00:00 1/01/02))
  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 BRIGSWS.

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

- Change 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 installation fails, the object is removed automatically.

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

Example 7: Remove Actions

The examples shown here match the first two examples shown for the install examples and demonstrate how some of 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(*PROD)
  ALTACTCOMP(*NOTALLOWED)
  AUTOACP((*YES))
  ALWRMV(*YES)
  COREQCHGNL((REXX PROC UPDATE CONFIG))
```

Instead of installing the objects on the single system MARYPWS1, the objects should be installed on all the managed systems controlled by the change control server ROMSERV1. (All the managed systems are listed in the node list ROMCLIENTS stored in library ROMLIB.) The objects should be installed in a removable manner, if all managed systems support removability. To install the objects in a removable manner, the install activity is changed to:

```
CHGCRQA CRQD(CCLIB/CRQACPRMV)
  ACTIVITY(INSACT01)
  ACTION(*SAME)
  GLBNAME(*SAME)
  NODL(ROMLIB/ROMCLIENTS)
  CPNAME(*NONE)
```

```
ACTUSEINS(*SAME)
ALTACTIONCOMP(*SAME)
AUTOACP(( *NO))
ALWRMV(*DESIRED)
COREQCHGNL(*SAME)
```

The objects IBM 1234567 PMGRAB UPD 1 2 and REXX PROC UPDATE CONFIG are all installed in a removable manner. You can remove the objects by changing the accept activities to the following:

The activities were added to the change request as follows:

```
ADDCRQA CRQD(CCLIB/CRQINSACC)
  ACTIVITY(RMVACT01)
  ACTION(*RMV)
  GLBNAME(IBM 1234567 PMGRAB UPD 1 2)
  CPNAME((ROMSERV1 MARYPWS1))
  COND((INSACT01 *EQ 20 *SAMENODE))
```

```
ADDCRQA CRQD(CCLIB/CRQINSACC)
  ACTIVITY(RMVACT02)
  ACTION(*RMV)
  GLBNAME(REXX PROC UPDATE CONFIG)
  CPNAME((ROMSERV1 MARYPWS1))
  COND((INSACT01 *EQ 20 *SAMENODE))
```

But they need to be changed to:

```
CHGCRQA CRQD(CCLIB/CRQINSACC)
  ACTIVITY(RMVACT01)
  ACTION(*SAME)
  GLBNAME(*SAME)
  NODL(ROMLIB/ROMCLIENTS)
  CPNAME(*NONE)
  COND((INSACT01 *EQ *SUCCESS *ALLNODES))
```

```
CHGCRQA CRQD(CCLIB/CRQINSACC)
  ACTIVITY(RMVACT02)
  ACTION(*SAME)
  GLBNAME(*SAME)
  NODL(ROMLIB/ROMCLIENTS)
  CPNAME(*NONE)
  COND((INSACT01 *EQ *SUCCESS *ALLNODES))
```

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

```
ADDCRQA CRQD(CCLIB/CRQACPRMV)
  ACTIVITY(INSACT02)
  ACTION(*INS)
  GLBNAME(EURO WORDPROD UPD 2 3 US)
  CPNAME((EUROITAL FREDSW))
  ALWRMV(*YES)
```

Instead of installing the objects on the single system MARYPWS1, the objects should be installed on all the managed systems controlled by the change control server EUROITAL. (All the managed systems are listed in the node list EUROCLIENT stored in library EUROLIB.) The activity is changed to:

```
CHGCRQA CRQD(CCLIB/CRQACPRMV)
  ACTIVITY(INSACT02)
  ACTION(*SAME)
  GLBNAME(*SAME)
  NODL(EUROLIB/EUROCLIENT)
  CPNAME(*NONE)
  ALWRMV(*YES)
```

To remove the object EURO WORDPROD UPD 2 3 US on all the managed systems, the activity RMVACT01 is added to change request CRQACPRMV:

```

ADDCRQA CRQD(CCLIB/CRQACPRMV)
  ACTIVITY(RMVACT01)
  ACTION(*RMV)
  GLBNAME(EURO WORDPROD UPD 2 3 US)
  CPNAME((EUROITAL FREDSW))
  COND((INSACT04 *EQ *SUCCESS *SAMENODE))

```

should be changed as follows:

```

CHGCRQA CRQD(CCLIB/CRQACPRMV)
  ACTIVITY(RMVACT01)
  ACTION(*SAME)
  GLBNAME(*SAME)
  NODL(EUROLIB/EUROCLIENT)
  CPNAME(*NONE)
  COND((INSACT04 *EQ *SUCCESS *ALLNODES))

```

- Change an activity to schedule the removal of the previously installed object EURO.WORDPROD.REF.2.US from managed system FREDSW with immediate effect but requiring activation.

```

CHGCRQA CRQD(CCLIB/CR6)
  ACTIVITY(ACT01)
  ACTION(*RMV)
  GLBNAME(EURO WORDPROD REF 2 US)
  CPNAME((EUROITAL FREDSW))
  ALTACTCOMP(*NOTALLOWED)

```

Example 8: Retrieve Actions

- Change 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 algorithm is used.

```

CHGCRQA CRQD(CCLIB/CRQRTV)
  ACTIVITY(RTVACT01)
  ACTION(*RTV)
  GLBNAME(EURO WORDPROC UPD 2 3 US)
  CPNAME((EUROITAL JOHNSWS))
  CPRTYPE(*SNA)
  CPRSTGTT(*DECOMPRESS)
  CPRTFRSTT(*COMPRESS)
  REPLACE(*ALLOWED)

```

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

```

CHGCRQA CRQD(CCLIB/CRQRTV)
  ACTIVITY(RTVACT02)
  ACTION(*RTV)
  GLBNAME(EURO PCSOFT UPD 2 3 US)
  CPNAME((EUROITAL JOHNSWS))
  CPRTYPE(*USER)
  CPRSTGTT(*COMPRESS)
  CPRTFRSTT(*COMPRESS)
  USRCPRINF(NVDMLZW 37 '/D')
  REPLACE(*ALLOWED)

```

- Change an activity to retrieve the most recent nightly 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 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. All files retrieved must be added to the distribution repository when they are retrieved.

```

CHGCRQA CRQD(CCLIB/CRQRTV)
  ACTIVITY(RTVACT03)
  ACTION(*RTV)

```

```

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 9: Send, Send and Run, and Send and Install Actions

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

```

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

```

- Change an activity to send and install the two objects EURO WORDPROC REF 2 US and EURO WORDPROC UPD 2 3 US on managed system FREDWS with the attributes that follow. 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.

```

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

```

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

```

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

```

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

```

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

CHGCRQA 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 10: Uninstall Actions

- Change an activity to uninstall the component EURO WORDPROC from managed system FREDSWs 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.

```
CHGCRQA CRQD(CCLIB/CRQUNINS)
  ACTIVITY(UNSACT01)
  ACTION(*UNINS)
  COMPNAME(EURO WORDPROC)
  CPNAME((EUROITAL FREDSWs))
  ALTACTION(*NOTALLOWED)
  FRCUNINS(*NO)
  REFLVL("2")
  RMTSTRTIME(*MGDSYS (14:00:00 05/20/02))
```

Error messages for CHGCRQA

*ESCAPE Messages

None <<

CHGCRQD (Change Change Request Description) Command Description

CHGCRQD Command syntax diagram

Purpose

The Change Change Request Description (CHGCRQD) command is used to change a change request description. A change request description contains a list of activities that are performed to complete the specified change. A prompt override is called to provide the current values.

Restrictions:

1. You must have *CHANGE authority to the change request description object.
2. To change the user profile, you must be the owner of the object or must have *ALLOBJ and *SECADM authority.

Required Parameter

CRQD Specifies the name and the library of the change request description being changed.

The name of the change request 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.

change-request-description-name: Specify the name of the change request description being changed.

Optional Parameters

USRPRF

Specifies the profile to use when the change request is submitted.

***SAME:** The value does not change.

***SBM:** The user profile of the submitter is used when the change request is submitted.

***OWNER:** The user profile that created the change request description is used when the change request is submitted.

PRBID

Specifies the ID of the problem to be associated with this change request description. Problems with different origin systems can have the same identifier.

***SAME:** The value does not change.

***NONE:** There is not a problem ID associated with this change request description.

problem-identifier: Specify the ID of the problem to be associated with the change request description.

PRBORG

Specifies the origin system of the problem ID.

Element 1: Network Identifier

***SAME:** The value does not change.

***NETATR:** The network ID is the same as the one defined in the network attributes for this system.

network-identifier: Specify a network ID.

Element 2: Control Point Name

***NETATR:** The control point name is the same as the local control point name defined in the network attributes for this system.

control-point-name: Specify a control point name.

TEXT Specifies the text that briefly describes the object. More information is in Commonly used parameters.

***SAME:** The value does not change.

***BLANK:** Text is not specified.

'description': Specify a maximum of 50 characters, enclosed in apostrophes.

Examples for CHGCRQD

Example 1: Changing the Text Description in Your Own Library

```
CHGCRQD CRQD(MYLIB/CHG001)
TEXT('This is the change')
```

This command changes the text description for the change request description in MYLIB with the name CHG001.

Example 2: Changing the Associated Problem ID

```
CHGCRQD CRQD(*LIBL/CHG002)
PRBID(1234567890)
```

This command changes the associated problem ID to the change request description in the library list named CHG002.

Error messages for CHGCRQD

***ESCAPE Messages**

CPF969B

Change request description changed, but warnings exist.

CPF969C

Not authorized to change USRPRF.

CHGCLS (Change Class) Command Description

CHGCLS Command syntax diagram

Purpose

The Change Class (CHGCLS) command changes the attributes contained in a class object. The class defines the processing attributes for jobs that use the class. The class used by a job is specified in the subsystem description routing entry used to start the job. If a job consists of multiple routing steps, the class used by each subsequent routing step is specified in the routing entry used to start the routing step.

Required Parameter

CLS Specifies the qualified name of the class.

Note:

The following IBM-supplied objects are not valid on this parameter: QARBCLS, QLPINSTALL, and QMONCLS.

More information on this parameter is in Commonly used parameters.

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.

Optional Parameters

RUNPTY

Specifies the run priority of jobs that use the class being changed. Run priority is a value, ranging from 1 (highest priority) through 99 (lowest priority), that represents the priority at which the job competes for the processing unit relative to other jobs that are active at the same time. This value represents the relative importance, not the absolute importance, of the job. For example, a job with a run priority of 25 is not twice as important as one with a run priority of 50. This value is the highest run priority allowed for any thread within the job. Individual threads within the job may have a lower priority.

***SAME:** The value does not change.

run-priority: Specify the run priority of the job that uses this specified class.

TIMESLICE

Specifies the maximum amount of processor time (in milliseconds) given to each thread in a job using this class before other threads in the job or other jobs are given the opportunity to run. The time slice establishes the amount of time needed by a thread in the job to accomplish a meaningful amount of processing. At the end of the time slice, the thread might be put in an inactive state so that other threads can become active in the storage pool.

***SAME:** The value does not change.

time-slice: Specify the maximum number of milliseconds that each thread in a job that uses this class can have to run. Valid values range from 1 through 9999999 (that is, 9 999 999 milliseconds or 9999.999 seconds).

Note:

Although you can specify a value of less than 8, the system takes a minimum of 8 milliseconds to run a process. If you display a job's run attributes, the time slice value is never less than 8.

PURGE

Specifies whether the job is marked as eligible to be moved out of main storage and put into auxiliary storage at the end of a time slice or when there is a long wait (such as waiting for a work station user's response). This attribute is ignored when more than one thread is active within the job.

***SAME:** The value does not change.

***YES:** The job is eligible to be moved out of main storage and put into auxiliary storage. However, a job with multiple threads is never purged from main storage.

***NO:** The job is not eligible to be moved out of main storage and put into auxiliary storage. However, when main storage is needed, pages belonging to a thread in this job may be moved to auxiliary storage. Then, when a thread in this job runs again, its pages are returned to main storage as they are needed.

DFTWAIT

Specifies the default maximum time (in seconds) that a thread in the job waits for a system instruction, such as the LOCK machine interface (MI) instruction, to acquire a resource. This default wait time is used when a wait time is not otherwise specified for a given situation. Normally, this would be the amount of time the system user is willing to wait for the system before the request is ended.

If the wait time for an instruction is exceeded, an error message is either displayed or it can be automatically handled by a Monitor Message (MONMSG) command.

***SAME:** The value does not change.

***NOMAX:** There is no maximum wait time.

seconds-to-wait: Specify a value, ranging from 0 through 9999999 seconds, that specifies the maximum time the system waits for the system instruction to acquire a resource.

Note:

Although a 0 default wait time is allowed, it is not recommended. Some system instructions require the use of system resources that may be in use and with a 0 default time, will cause the instruction to fail. When a system instruction fails (exceeds the default wait time) unexpected results may occur for the thread. Most system resources will only be in use for a short time, so having a small default wait time will not noticeably degrade the performance of the thread.

CPUTIME

Specifies the maximum processing unit time (in milliseconds) that the job can use. If the job consists of multiple routing steps, each routing step is allowed to use this amount of processing unit time. If the maximum time is exceeded, the job is ended.

***SAME:** The value does not change.

***NOMAX:** There is no limit on the processing unit time used.

maximum-CPU-time: Specify the maximum amount of processing unit time (in milliseconds) that can be used. Valid values range from 1 through 9999999 milliseconds (that is, 9 999 999 milliseconds or 9999.999 seconds).

MAXTMPSTG

Specifies the maximum amount of temporary (auxiliary) storage (in kilobytes) that the job can use. If the job consists of multiple routing steps, this is the maximum temporary storage that the routing step can use. This temporary storage is used for storage required by the program itself and by implicitly created internal system objects used to support the job. (It does not include storage in the QTEMP library.) If the maximum temporary storage is exceeded, the job is ended. This parameter does not apply to the use of permanent storage, which is controlled through the user profile.

***SAME:** The value does not change.

***NOMAX:** The system maximum is used.

maximum-temporary-storage: Specify the number of kilobytes (ranging from 1 through 2147483647) that specifies the maximum amount of temporary storage that can be used.

Note:

Although the value is specified in kilobytes, the specified value is stored in the class rounded up to the nearest megabyte.

MAXTHD

Specifies the maximum number of threads that a job using this class can run with at any time. If multiple threads are initiated simultaneously, this value may be exceeded. If this maximum value is exceeded, the excess threads will be allowed to run to their normal completion. Initiation of additional threads will be inhibited until the maximum number of threads in the job drops below this maximum value.

***SAME:** The value does not change.

***NOMAX:** There is no maximum number of threads.

maximum-threads: Specify a value (ranging from 1 through 32767) that specifies the maximum number of threads for a job.

Note:

Depending upon the resources used by the threads and the resources available on the system, the initiation of additional threads may be inhibited before this maximum value is reached.

TEXT Specifies the text that briefly describes the class. More information is in Commonly used parameters.

***SAME:** The value does not change.

***BLANK:** Text is not specified.

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

Example for CHGCLS

```
CHGCLS CLS(CLASS1) RUNPTY(60) TIMESLICE(900)
```

This command changes a class called CLASS1 in the library on the job's library list. The run priority for the class is changed to 60 and a time slice of 900 milliseconds.

Error messages for CHGCLS

*ESCAPE Messages

CPF1169

Class &1 in library &2 not changed.

CHGCOSD (Change Class-of-Service Description) Command Description

CHGCOSD Command syntax diagram

Purpose

The Change Class-of-Service Description (CHGCOSD) command changes a class-of-service description.

Required Parameter

COSD Specifies the name of the class-of-service description changed. This name ranges from 1 through 8 characters.

Optional Parameters

TMSPTY

Specifies the transmission priority for this class-of-service description. Valid priority levels are listed below.

***SAME:** This value does not change.

***LOW:** The lowest transmission priority for this class-of-service description is used.

***MED:** Medium transmission priority for this class-of-service description is used.

***HIGH:** The highest transmission priority for this class-of-service description is used.

ROW1LINE - ROW8LINE

Specifies the list of line-related criteria used for the first through the eighth rows of the class-of-service description. These rows describe the attributes of the line connection between two nodes in the APPN network. The rows are examined in order (from 1 through 8) to define a network routing path.

Element 1: Line Weight Factor

This is the weighting factor used in the computation.

***SAME:** This value does not change.

line-weighting-factor: Specify the relative weight of this row for lines. The weight, which ranges from 0 through 255, indicates the relative cost of a line connection.

Element 2: Minimum Link Speed

This is the minimum link speed for a line connection that is accepted by this row criteria.

***SAME:** This value does not change.

***MIN:** The minimum link speed is specified.

***MAX:** The maximum link speed is specified.

minimum-link-speed: Specify the minimum link speed. Valid values are *MIN, 1200, 2400, 4800, 7200, 9600, 14400, 19200, 48000, 56000, 64000, 112000, 128000, 168000, 192000, 224000, 256000, 280000, 320000, 336000, 384000, 448000, 499000, 576000, 614000, 691000, 768000, 845000, 922000, 998000, 1075000, 1152000, 1229000, 1382000, 1536000, 1690000, 1843000, 1997000, 4M, 10M, 16M, or *MAX.

Element 3: Maximum Link Speed

This is the maximum link speed for a line connection that is accepted by this row criteria.

***SAME**: This value does not change.

***MIN**: The minimum link speed is specified.

***MAX**: The maximum link speed is specified.

maximum-link-speed: Specify the maximum link speed. See the list of valid values under minimum link speed in Element 2.

Element 4: Minimum Cost Per Connect Time

***SAME**: This value does not change.

cost/connect-minimum: Specify the minimum relative cost per connect time that is accepted by this row criteria. Valid costs range from 0 through 255.

Element 5: Maximum Cost Per Connect Time

***SAME**: This value does not change.

cost/connect-maximum: Specify the maximum relative cost per connect time that is accepted by the row criteria. Valid costs range from 0 through 255.

Element 6: Minimum Cost Per Byte

***SAME**: This value does not change.

cost/byte-minimum: Specify the minimum relative cost per byte that is accepted by this row criteria. Valid costs range from 0 through 255.

Element 7: Maximum Cost Per Byte

***SAME**: This value does not change.

cost/byte-maximum: Specify the maximum relative cost per byte that is accepted by this row criteria. Valid costs range from 0 through 255.

Element 8: Minimum Security Level

The minimum security level that is accepted by this row criteria. Valid values are in order from least to most secure.

***SAME**: This value does not change.

***NONSECURE**: No security is the minimum security level accepted by this row criteria.

***PKTSWTNET**: Packet switched network is the minimum security level accepted by this row criteria.

***UNDGRDCBL**: Underground cable is the minimum security level accepted by this row criteria.

***SECURECND**: Secure conduit is the minimum security level accepted by this row criteria.

***GUARDCND**: Guarded conduit is the minimum security level accepted by this row criteria.

***ENCRYPTED**: Encrypted line is the minimum security level accepted by this row criteria.

***MAX**: Guarded conduit, protected against physical and radiation tapping is the minimum security level accepted by this row criteria.

Element 9: Maximum Security Level

The maximum security level that is accepted by this row criteria. Valid values are in order from least to most secure.

***SAME:** This value does not change.

***NONSECURE:** No security is the maximum security level accepted by this row criteria.

***PKTSWTNET:** Packet switched network is the maximum security level accepted by this row criteria.

***UNDGRDCBL:** Underground cable is the maximum security level accepted by this row criteria.

***SECURECND:** Secure conduit is the maximum security level accepted by this row criteria.

***GUARDCND:** Guarded conduit is the maximum security level accepted by this row criteria.

***ENCRYPTED:** Encrypted line is the maximum security level accepted by this row criteria.

***MAX:** Guarded conduit, protected against physical and radiation tapping is the maximum security level accepted by this row criteria.

Element 10: Minimum Propagation Delay

The minimum propagation delay that is accepted by this row criteria. The valid values are in order from least to longest delay.

***SAME:** This value does not change.

***MIN:** Minimum propagation delay is accepted by this row criteria.

***LAN:** Local Area Network propagation delay is the minimum propagation delay accepted by this row criteria.

***TELEPHONE:** Telephone propagation delay is the minimum propagation delay accepted by this row criteria.

***PKTSWTNET:** Packet switched network propagation delay is the minimum propagation delay accepted by this row criteria.

***SATELLITE:** Satellite propagation delay is the minimum propagation delay accepted by this row criteria.

***MAX:** Maximum propagation delay is the minimum propagation delay accepted by this row criteria.

Element 11: Maximum Propagation Delay

The maximum propagation delay that is accepted by this row criteria. Valid values are in order from least to longest delay.

***SAME:** This value does not change.

***MIN:** Minimum propagation delay is the maximum propagation delay accepted by this row criteria.

***LAN:** Local Area Network propagation delay is the maximum propagation delay accepted by this row criteria.

***TELEPHONE:** Telephone propagation delay is the maximum propagation delay accepted by this row criteria.

***PKTSWTNET:** Packet switched network propagation delay is the maximum propagation delay accepted by this row criteria.

***SATELLITE:** Satellite propagation delay is the maximum propagation delay accepted by this row criteria.

***MAX:** Maximum propagation delay is the maximum propagation delay accepted by this row criteria.

Element 12: User's First Minimum Line Connection Criteria

***SAME:** This value does not change.

user-1-minimum: Specify the user's own line connection criteria. Valid values range from 0 through 255.

Element 13: User's First Maximum Line Connection Criteria

***SAME:** This value does not change.

user-1-maximum: Specify user's own line connection criteria. Valid values range from 0 through 255.

Element 14: User's Second Minimum Line Connection Criteria

***SAME:** This value does not change.

user-2-minimum: Specify user's own line connection criteria. Valid values range from 0 through 255.

Element 15: User's Second Maximum Line Connection Criteria

***SAME:** This value does not change.

user-2-maximum: Specify user's own line connection criteria. Valid values range from 0 through 255.

Element 16: User's Third Minimum Line Connection Criteria

***SAME:** This value does not change.

user-3-minimum: Specify user's own line connection criteria. Valid values range from 0 through 255.

Element 17: User's Third Maximum Line Connection Criteria

***SAME:** This value does not change.

user-3-maximum: Specify user's own line connection criteria. Valid values range from 0 through 255.

ROW1NODE - ROW8NODE

Specifies the list of node-related criteria used for the first through the eighth rows of the class-of-service description. This row describes the attributes of a node in the APPN network. The rows are examined in order (from 1 to 8) to define a network routing path.

Element 1: Node Weighting Factor

***SAME:** This value does not change.

node-weighting-factor: Specify the relative weight of this row for nodes. The weight ranges from 0 through 255.

Element 2: Minimum Route Additional Resistance

***SAME:** This value does not change.

route-addition-resistance-minimum: Specify the minimum route additional resistance accepted by this row criteria. Valid values range from 0 through 255.

Element 3: Maximum Route Additional Resistance

***SAME:** This value does not change.

route-addition-resistance-maximum: Specify the maximum route additional resistance accepted by this row criteria. Valid values range from 0 through 255.

Element 4: Minimum Congestion Level

Specifies the minimum level of congestion tolerated.

***SAME:** This value does not change.

***LOW:** Low congestion is the minimum level tolerated.

***HIGH:** High congestion is the minimum level tolerated.

Element 5: Maximum Congestion Level

Specifies the maximum level of congestion tolerated.

***SAME:** This value does not change.

***LOW:** Low congestion is the maximum level tolerated.

***HIGH:** High congestion is the maximum level tolerated.

TEXT Specifies the text that briefly describes the description. More information is in Commonly used parameters.

***SAME:** The value does not change.

***BLANK:** Text is not specified.

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

Example for CHGCOSD

```
CHGCOSD COSD(COSD1) ROW4LINE(80 *SAME *SAME 15)
```

This command changes Row 4 line weight to 80 and Row 4 minimum cost/connect time to 15 for class-of-service description COSD1.

Error messages for CHGCOSD

*ESCAPE Messages

CPF2621

Class-of-service description &1 not changed.

CHGCLNUP (Change Cleanup) Command Description

CHGCLNUP Command syntax diagram

Purpose

The Change Cleanup (CHGCLNUP) command allows the user to specify the cleanup options. The cleanup options control the following:

- If cleanup is allowed to run.
- When cleanup is run each day.
- What objects are cleaned up.

If cleanup is not active when this command is entered, the new values specified for this command are used the next time cleanup is started (with the Start Cleanup (STRCLNUP) command).

Restriction: The user must have *ALLOBJ and *JOBCTL special authorities to use this command.

Optional Parameters

ALWCLNUP

Specifies whether cleanup can be run on the system. If ALWCLNUP(*YES) is specified, cleanup can be started with the STRCLNUP command.

***SAME:** The value does not change.

***YES:** Cleanup can be run on this system.

***NO:** Cleanup cannot be run on this system.

STRTIME

Specifies the time when cleanup will run each day.

***SAME:** The value does not change.

***SCDPWROFF:** Cleanup starts at the time of the scheduled power off. The power off takes place when cleanup finishes, whether or not cleanup was completed successfully.

***NONE:** No cleanup start time is scheduled. The cleanup batch jobs are not submitted.

time-of-day: Specify the time when daily cleanup will run each day.

USRMSG

Specifies whether messages on user profile message queues are cleaned up. The cleanup function deletes messages on the user message queues that have remained on the system longer than the number of days specified.

***SAME:** The value does not change.

***KEEP:** Messages are not deleted.

number-of-days: Specify the number of days messages are kept before they are deleted. Valid values range from 1 through 366.

SYSMMSG

Specifies that messages on the QSYSOPR message queue and on work station message queues are cleaned up. The cleanup function deletes messages on the QSYSOPR message queue and on work station message queues that have remained on the system longer than the number of days specified.

***SAME:** The value does not change.

***KEEP:** Messages are not deleted.

number-of-days: Specify the number of days messages are kept before they are deleted. Valid values range from 1 through 366.

SYSPRT

Specifies that job logs and other system output are cleaned up.

To prevent this output from being mixed with the user's output, the output queue of the printer file for job logs (QPJOBLOG) is changed to QUSRSYS/QEZJOBLOG to receive the job log. The output queues, QPPGMDMP, QPBASDMP, and QPSRVDMP are changed to QUSRSYS/QEZDEBUG. All entries in QEZJOBLOG and QEZDEBUG that are older than the number of days specified on this parameter are deleted.

If the cleanup operation is ended, the output queues named QEZJOBLOG and QEZDEBUG continue to be used for job logs, service dumps, and program dumps.

***SAME:** The value does not change.

***KEEP:** Job logs and other system output are not deleted.

number-of-days: Specify the number of days job logs and other system output are kept before they are deleted. Valid values range from 1 through 366.

SYSLOG

Specifies that system journal receivers, history log files, problem log files and problem log entries, alert database entries, and program temporary fixes are cleaned up.

Journal receivers:

Journal receivers that are used for one of the following system journals and are older than the number of days specified on this parameter are cleaned up:

QAOSDIAJRN

Journal for DIA files

QDSNX

Journal for DSNX logs

QSNADS

Journal for SNADS files

QSXJRN

Journal for problem databases

QPFRAJRN

Journal for performance adjustment data

QACGJRN

Journal for job accounting data

QX400

OSI Message Services/400

QCQJMJRN

Journal for Managed System Services/400

QO1JRN

Journal for Application Enabler OFC files

ADJRNLO

Journal for application program driver files

QSNMP

Journal for SNMP

QLYJRN

Journal for Application Development Manager Transactions

QLYPRJLOG

Journal for project logs

QMAJRN

Journal for work order requests

QZMF Journal for QMSF job

Note:

The journal receiver for job accounting (QACGJRN) is cleaned up only if the journal is created by Operational Assistant*.

History log files:

History log files that meet both of the following conditions are deleted:

1. History log files that are older than the number of days specified on this parameter.
2. History log files named QSYS/QHST*.

Problem log files and problem log entries:

Any problem log entries older than the number of days specified on this parameter are deleted. The Delete Problem (DLTPRB) command is run to delete the problem log entries. When the DLTPRB command is run, the number of days specified for this parameter is used on the DAYS parameter of the DLTPRB command. The defaults are used for all other parameters on the DLTPRB command. Refer to the DLTPRB command description for more details on how this command runs.

Note:

If the number of days specified for this parameter is less than the number of days specified for the system value QPRBHLDTIV (Problem Log Hold Interval), the value for QPRBHLDTIV is used instead for cleaning up problem logs.

In addition to the problem log entries being deleted, the following problem log files are reorganized:

Note:

The following files are in library QUSRSYS.

QASXCALL	QASXFRU	QASXNOTE
QASXPROB	QASXPTF	QASXYMP
QASXEVT		

Alert database entries: Any alert database entries older than the number of days specified on this parameter are deleted. The Delete Alert (DLTALR) command is run to delete the alert database entries. When the DLTALR command is run, the number of days specified for this parameter is used on the DAYS parameter of the DLTALR command. The defaults are used for all other parameters on the DLTALR command. Refer to the DLTALR command description for more details on how this command runs.

In addition to the alert database entries being deleted, the file QUSRSYS/QAALERT is reorganized.

Program temporary fixes:

The following PTFs are deleted:

- Temporary objects named:
 - QPZA000000 through QPZA999999
 - QPZR000000 through QPZR999999
 - QPZI000000 through QPZI999999
 - QSCA000000 through QSCA999999
 - QSCR000000 through QSCR999999
- Exit programs shipped with PTFs
- Physical files in QUSRSYS
- QAPZPTF
- QAPZREQ
- QAPZSYM

If the library, QSMU, exists on the system, then only the PTFs for the current release are cleaned up. If the QSMU library does not exist on the system, then PTFs for this and all previous releases are cleaned up.

***SAME:** The value does not change.

***KEEP:** System journals and system logs are not deleted.

number-of-days: Specify the number of days system journals and system logs are kept before they are deleted. Valid values range from 1 through 366.

CALITM

Specifies whether OfficeVision* calendar items are kept or deleted by the cleanup operation after a specified number of days.

***SAME:** The value does not change.

***KEEP:** Calendar items are not deleted.

number-of-days: Specify the number of days calendar items are kept before they are deleted. Valid values range from 1 through 366.

JOBQ Specifies the qualified name of the job queue on which this job is placed.

***SAME:** The job queue does not change.

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.

QCTL: Batch cleanup jobs are submitted to the QCTL job queue.

job-queue-name: Specify the qualified name of the job queue on which the submitted job is placed.

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. For example, a routing step with a run priority of 25 is not twice as important as one with a run priority of 50. If the run priority has never been explicitly set, OA will use 55.

***SAME:** The value does not change.

machine-running-priority: Specify the run priority, ranging from 1 through 99, that the routing step uses.

JRNRCVSIZ

Specifies a value ranging from 1 through 1,919,999 in kilobytes (KB) of storage. Each 1000 KB specifies 1,024,000 bytes of storage space. When the size of the space for the journal receiver is larger than the size specified by this value, Operational Assistant (OA) automatic cleanup function will detach it. If the size has never been explicitly set, OA will use 5000KB.

OA also automatically detaches receivers that are older than the number of days you have specified on the SYSLOG parameter.

Journal receivers are deleted only after they have been detached longer than the number of days specified on the SYSLOG parameter.

***SAME:** The value does not change.

journal-receiver-size: Specify a value ranging from 1 through 1,919,999 in kilobytes (KB) of storage. If this value is exceeded when OA automatic cleanup is run, the journal receiver will be detached.

Examples for CHGCLNUP

Example 1: Keeping User Messages During Cleanup

```
CHGCLNUP ALWCLNUP(*YES) USRMSG(*KEEP) STRTIME(0700)
```

This command changes the cleanup options so that user messages are kept and not deleted when cleanup is performed. This command sets cleanup start time at 7:00 A.M.

Example 2: Cleanup of System Journals and System Logs

```
CHGCLNUP ALWCLNUP(*YES) SYSMSG(10) SYSLOG(3)
```

This command changes the cleanup options so that system messages are kept for ten days, and system journals and system logs are kept for three days, before being deleted.

Example 3: Changing Run Priority of Cleanup Job.

```
CHGCLNUP RUNPTY(50)
```

This command changes the run priority to 50.

Example 4: Changing Journal Receiver Size

```
CHGCLNUP JRNRCVSIZ(6000)
```

This changes the journal receiver size limit to 6000 KB of storage, (6144000 bytes).

Error messages for CHGCLNUP

*ESCAPE Messages

CPF1E2A

Unexpected error in QSYSSCD job.

CPF1E2B

Power scheduler and cleanup options not found.

CPF1E3C

Job queue &2/&1 not found.

CPF1E3D

Library &1 for JOBQ parameter not found.

CPF1E32

Not authorized to change cleanup options.

CPF1E33

Cleanup options or power schedule in use by another user.

CPF1E99

Unexpected error occurred.



CHGCLUCFG (Change Cluster Configuration Tuning) Command Description

CHGCLUCFG Command syntax diagram

Purpose

The Change Cluster Configuration Tuning (CHGCLUCFG) command is used to tune cluster performance and configuration parameters. The command provides a base level of tuning support where the cluster will adjust to a predefined set of values identified for maximum, minimum, and normal timeout and messaging interval values. If an advanced level of tuning is desired, usually anticipated with the help of IBM support personnel, then individual parameters may be tuned over a predefined range of values using the Change Cluster Resource Services (QcstChgClusterResourceServices) API.

Values for current settings may be retrieved using the Display Cluster Information (DSPCLUINF) command.

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. This command must be called on a cluster node with a status of Active.
4. The tuning parameters defined under the Change Cluster Resource Services API must match exactly in both partitions for a merge will to be allowed.

Required Parameter

CLUSTER

Specifies the name of the cluster for which the configuration and tuning will be changed.

cluster-name: Specify the name of the cluster.

Optional Parameter

LEVEL

Specifies the desired cluster communications configuration tuning sensitivity level. Provides a simple way to set cluster performance and configuration parameters.

***NORMAL:** Default values are used for cluster communications performance and configuration parameters. This setting may be used to return all parameters to the original default values.

***MIN:** Adjustments are made to cluster communications to increase the heartbeating interval and increase the various message timeout values. With fewer heartbeats and longer timeout values, the cluster will be slower to respond (minimum sensitivity) to communications failures.

***MAX:** Adjustments are made to cluster communications to decrease the heartbeating interval and decrease the various message timeout values. With more frequent heartbeats and shorter timeout values, the cluster will be quicker to respond (maximum sensitivity) to communications failures.

Example for CHGCLUCFG

```
CHGCLUCFG CLUSTER(MYCLUSTER) LEVEL(*MIN)
```

This command changes the configuration and tuning level of the cluster MYCLUSTER. The tuning level is set to the minimum sensitivity level.

Error messages for CHGCLUCFG

*ESCAPE Messages



CHGCLUNODE (Change Cluster Node Entry) Command Description

CHGCLUNODE Command syntax diagram

Purpose

The Change Cluster Node Entry (CHGCLUNODE) command is used to change cluster membership information for a cluster node entry. The information that can be changed are the cluster interface addresses defined for the node and status of the node. The node entry which is being changed may or may not have Cluster Resource Services started.

You can add, remove, or replace an interface address for the cluster node. The cluster interface address is an IP address that is used by Cluster Resource Services to communicate with other nodes in the cluster. The address is in dotted decimal format.

You can also change the status of a node to Failed. Using this CL command to change the status of a node to Failed provides a way to tell Cluster Resource Services that a node has really failed. There are certain failure conditions that Cluster Resource Services cannot detect as a node failure. Rather, the problem appears to be a communication problem and the cluster has become partitioned. By telling Cluster Resource Services that a node has failed, it makes recovery from the partition state simpler since a backup node from the remaining active cluster nodes can then be assigned as the primary node.

When you change the status of a node to Failed, the role of nodes in the recovery domain for each cluster resource group in the partition may be reordered by assigning the specified node as the last backup. If an exit program is specified for the cluster resource group, it will be called with an action code of Change Node Status. If multiple nodes have failed and their status needs to be changed, the order in which the nodes are changed will affect the final order of the recovery domain's backup nodes in the cluster resource group.

If the node status is changed to Failed and the node was the primary node for a cluster resource group, the first active backup will be reassigned as the new primary node. When this occurs for a device cluster resource group, ownership of the hardware will be moved to the new primary node.

If a problem is detected and the command does not complete successfully, the command can be run again once the problem is corrected. Any cluster resource group that had already had the status of a node changed from Partition to Failed and the recovery domain order changed will not be affected by running this command again.

Warning:

Changing the node status to Failed when, in fact, the node is still active and a true partition has occurred should not be done. Doing so allows a node in each partition to become the primary node for a cluster resource group. When two nodes think they are the primary node, data such as files or data bases could become corrupted if two different nodes are each making independent changes to copies of their files. In addition, the two partitions cannot be merged back together when a node in each partition has been assigned the primary role.

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. This command must be called from a program running on a cluster node with a status of Active.
4. If the cluster is in a partitioned state, this operation can only be performed within the partition running the command.
5. Only one cluster interface address can be changed at a time. If the cluster is in partitioned state, the change cluster interface address is only allowed for a node within the same partition.
6. To change the cluster node status, only a node that has a status of Partition or Failed can be changed and it can only be changed to Failed status.

Required Parameters

CLUSTER

Specifies the name of the cluster that contains the node being changed.

cluster-name: Specify the name of the cluster.

NODE Specifies the node identifier being changed.

node-identifier: Specify the name of the node being changed.

OPTION

Indicates what is being changed.

***ADDIFC**: Add an interface address for the specified node.

***RMVIFC**: Remove an interface address for the specified node.

***CHGIFC**: Replace an interface address for the specified node with a different existing interface address.

***CHGSTS**: Change cluster node status to Failed.

OLDINTNETA

Specifies the cluster interface address which is being replaced or removed. The address is in dotted decimal format.

old-interface-address: Specify the cluster interface address to be replaced and removed.

NEWINTNETA

Specifies the cluster interface address which is being added to the node information or replacing an old cluster interface address. The address is in dotted decimal format.

new-interface-address: Specify the cluster interface address which is to be used to communicate with the node.

Example for CHGCLUNODE

```
CHGCLUNODE CLUSTER(MYCLUSTER) NODE(NODE01) OPTION(*CHGSTS)
```

This command changes the status of node NODE01 in cluster MYCLUSTER to Failed.

Error messages for CHGCLUNODE

*ESCAPE Messages

CPF0001

Error found on &1 command.



CHGCLURCY (Change Cluster Recovery) Command Description

CHGCLURCY Command syntax diagram

Purpose

The Change Cluster Recovery (CHGCLURCY) command is used only for problem analysis. Use this CL command only when directed by IBM Service personnel. It directs a node in a cluster resource group to perform a specific recovery action. The recovery action can cancel the current operation, force a recovery of the cluster resource group object associated with the cluster resource group, force a rejoin with the cluster resource group, or end the job that is associated with the cluster resource group.

The actions are intended to be used when a cluster resource group is experiencing a problem, and you need to force some recovery action onto the group. The problem may not be due to Cluster Resource Services. For example, Cluster Resource Services submits a job to invoke user exit programs. If the job is held, then it appears to a user that the cluster resource group is hung. A user may not know what exit program job was submitted, and so cannot perform any recovery outside of the cluster. Performing the appropriate recovery action with CHGCLURCY can satisfy Cluster Resource Services so it can fail the protocol that invoked the exit program job and continue.

Restrictions

1. To use this command you must have *JOBCTL authority, and either *SERVICE authority or be authorized to the Service Trace function of the operating system through iSeries Navigator's Application Administration support. You must also have *CHANGE authority to any cluster resource group object that is to be acted upon with this command.
2. The cluster must be at version 3 or greater for this command to work remotely (work on any node other than the node issuing the command).
3. Cluster Resource Services must either be active or in the process of starting on the node that this command is issued from.
4. Only nodes that have a job for the desired cluster resource group may participate in this command.
5. To determine if this command succeeded, check the job logs of the affected cluster jobs for a CPDBB06 message indicating the recovery action performed.

Warning

Use caution with this command, recovery actions cannot be undone or canceled.

Required Parameters

CLUSTER

Specifies the name of the cluster that is to be operated upon.

cluster-name: Specifies the cluster name.

CRG Specifies the name of the cluster resource group that is to be operated upon. Possible values are:

***ALL**: All groups, including the reserved groups QCSTCTL and QCSTCRGM.

cluster-resource-group-name: Specifies the name of the cluster resource group. The reserved names for the cluster control and cluster resource group manager groups, QCSTCTL and QCSTCRGM respectively, may also be specified.

NODE Specifies the cluster node that is to be operated upon. Possible values are:

***ALL**: All active nodes in the cluster.

node-identifier: Specifies the cluster node identifier.

ACTION

Specifies a recovery action for the specified cluster resource group on the specified node.

Notes:

1. The only valid value for NODE(*ALL) is *END.
2. The only valid value for CRG(*ALL) is *END.
3. For a CRG parameter value of QCSTCTL or QCSTCRGM, actions *RESTART or *REJOIN will cause clustering to end on the specified node, and then the node will attempt to automatically start. For *END, clustering will end on the node and clustering will not be automatically restarted.

***CANCEL:** Cancels the current protocol request. If no protocol is in-progress, the action is ignored.

***RESTART:** Specifies to restart the specified node in the specified cluster resource group by performing a start action without the cluster resource group doing any failover of the node. The cluster resource group object associated with the cluster resource group will be recovered from another node in the recovery domain of the group. If there is not another active node in the recovery domain, then the cluster resource group will go inactive and no further operations on it will occur until another node in the recovery domain starts that has a valid cluster resource group object.

***REJOIN:** Specifies to rejoin the specified node in the specified cluster resource group. The cluster resource group will first do a failover if needed, and then the cluster resource group will attempt to automatically start the node only for that cluster resource group.

***END:** The job on the specified node that corresponds to the specified cluster resource group is ended. This may cause a failover in the cluster resource group. The cluster resource group will not automatically start the node, and no further cluster resource group operations can be performed on the node without first ending, then starting, clustering on the node.

Examples for CHGCLURCY

Example 1: Recovery action for one node in one cluster resource group

```
CHGCLURCY CLUSTER(EXAMPLE) CRG(CRG1) NODE(NODE1) ACTION(*CANCEL)
```

This command cancels the current protocol in cluster resource group CRG1 on node NODE1 in cluster EXAMPLE.

Example 2: Recovery action for a reserved cluster resource group

```
CHGCLURCY CLUSTER(EXAMPLE) CRG(QCSTCRGM) NODE(NODE1) ACTION(*RESTART)
```

This command causes node NODE1 of cluster resource group QCSTCRGM in cluster EXAMPLE to end clustering on NODE1. Clustering on NODE1 will attempt to automatically restart itself as though NODE1 was started with the Start Cluster Node (STRCLUNOD) command.

Example 3: Recovery action for all nodes

```
CHGCLURCY CLUSTER(EXAMPLE) CRG(CRG1) NODE(*ALL) ACTION(*END)
```

This command causes all cluster jobs on all nodes associated with cluster resource group CRG1 in cluster EXAMPLE to end. No further operations on CRG1 can be performed on a given node until that node has clustering ended, and then started. The End Cluster Node (ENDCLUNOD) and STRCLUNOD commands may be used for ending and starting a cluster node.

Example 4: Recovery action for all groups

```
CHGCLURCY CLUSTER(EXAMPLE) CRG(*ALL) NODE(NODE1) ACTION(*END)
```

This command causes all cluster jobs on node NODE1 in cluster EXAMPLE to end. This has the effect of ending clustering on NODE1. To start the cluster jobs on NODE1 requires NODE1 to have clustering started on it. The STRCLUNOD command may be used for starting a cluster node.

Error messages for CHGCLURCY

***ESCAPE Messages**

CPF0001

Error found on &1 command.

CPF222E

&1 special authority is required.

CPF98A2

Not authorized to &1 command.

CPFBB02

Cluster &1 does not exist.

CPFBB09

Cluster node &1 does not exist in cluster &2.

CPFBB0F

Cluster resource group &1 does not exist in cluster &2.

CPFBB70

Request &1 not compatible with current cluster version.

CPFBBA0

Node &1 in cluster resource group &2 is not responding.





Printed in U.S.A.