Tivoli® System Automation for z/OS
Version 3 Release 3

TWS Automation Programmer's Reference and Operator's Guide

IBM

SC34-2579-00

Tivoli® System Automation for z/OS
Version 3 Release 3

TWS Automation Programmer's Reference and Operator's Guide

IBM

SC34-2579-00

Note!

Before using this information and the product it supports, read the information in "Notices," on page 137.

This edition applies to IBM Tivoli System Automation for z/OS (Program Number 5698-SA3) Version 3 Release 3, an IBM licensed program, and to all subsequent releases and modifications until otherwise indicated in new editions.

| This edition replaces SC33-8269-04.

| IBM welcomes your comments. You may forward your comments electronically, or address your comments to:
| IBM Deutschland Research & Development GmbH
| Department 3248
| Schoenaicher Strasse 220
| 71032 Boeblingen
| Germany

If you prefer to send comments electronically, use one of the following methods:
FAX (Germany): 07031 16-3456
FAX (Other Countries): +49 7031 16-3456
Internet: s390id@de.ibm.com

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© Copyright IBM Corporation 1990, 2010.
US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

© Copyright IBM Corp. 1990, 2010

Figures

Tables

Accessibility

Publications for this product are offered in Adobe Portable Document Format (PDF) and should be compliant with accessibility standards. If you experience difficulties when using PDF files, you may view the information through the z/OS Internet Library website or the z/OS Information Center. If you continue to experience problems, send an email to mhvrcfs@us.ibm.com or write to:

IBM Corporation
Attention: MHVRCFS Reader Comments
Department H6MA, Building 707
2455 South Road
Poughkeepsie, NY 12601-5400
U.S.A.

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use software products successfully. The major accessibility features in z/OS® enable users to:

- Use assistive technologies such as screen readers and screen magnifier software
- Operate specific or equivalent features using only the keyboard
- Customize display attributes such as color, contrast, and font size

Using assistive technologies

Assistive technology products, such as screen readers, function with the user interfaces found in z/OS. Consult the assistive technology documentation for specific information when using such products to access z/OS interfaces.

Keyboard navigation of the user interface

Users can access z/OS user interfaces using TSO/E or ISPF. Refer to *z/OS TSO/E Primer*, *z/OS TSO/E User's Guide*, and *z/OS ISPF User's Guide Vol I* for information about accessing TSO/E and ISPF interfaces. These guides describe how to use TSO/E and ISPF, including the use of keyboard shortcuts or function keys (PF keys). Each guide includes the default settings for the PF keys and explains how to modify their functions.

z/OS information

z/OS information is accessible using screen readers with the BookServer or Library Server versions of z/OS books in the Internet library at:

`http://www.ibm.com/systems/z/os/zos/bkserv/`

© Copyright IBM Corp. 1990, 2010

Dotted decimal syntax diagrams

Syntax diagrams are provided in dotted decimal format for users accessing the Information Center using a screen reader. In dotted decimal format, each syntax element is written on a separate line. If two or more syntax elements are always present together (or always absent together), they can appear on the same line, because they can be considered as a single compound syntax element.

Each line starts with a dotted decimal number; for example, 3 or 3.1 or 3.1.1. To hear these numbers correctly, make sure that your screen reader is set to read out punctuation. All the syntax elements that have the same dotted decimal number (for example, all the syntax elements that have the number 3.1) are mutually exclusive alternatives. If you hear the lines 3.1 USERID and 3.1 SYSTEMID, you know that your syntax can include either USERID or SYSTEMID, but not both.

The dotted decimal numbering level denotes the level of nesting. For example, if a syntax element with dotted decimal number 3 is followed by a series of syntax elements with dotted decimal number 3.1, all the syntax elements numbered 3.1 are subordinate to the syntax element numbered 3.

Certain words and symbols are used next to the dotted decimal numbers to add information about the syntax elements. Occasionally, these words and symbols might occur at the beginning of the element itself. For ease of identification, if the word or symbol is a part of the syntax element, it is preceded by the backslash (\) character. The * symbol can be used next to a dotted decimal number to indicate that the syntax element repeats. For example, syntax element *FILE with dotted decimal number 3 is given the format 3 * FILE. Format 3* FILE indicates that syntax element FILE repeats. Format 3* * FILE indicates that syntax element * FILE repeats.

Characters such as commas, which are used to separate a string of syntax elements, are shown in the syntax just before the items they separate. These characters can appear on the same line as each item, or on a separate line with the same dotted decimal number as the relevant items. The line can also show another symbol giving information about the syntax elements. For example, the lines 5.1*, 5.1 LASTRUN, and 5.1 DELETE mean that if you use more than one of the LASTRUN and DELETE syntax elements, the elements must be separated by a comma. If no separator is given, assume that you use a blank to separate each syntax element.

If a syntax element is preceded by the % symbol, this indicates a reference that is defined elsewhere. The string following the % symbol is the name of a syntax fragment rather than a literal. For example, the line 2.1 %OP1 means that you should refer to separate syntax fragment OP1.

The following words and symbols are used next to the dotted decimal numbers:

- ? means an optional syntax element. A dotted decimal number followed by the ? symbol indicates that all the syntax elements with a corresponding dotted decimal number, and any subordinate syntax elements, are optional. If there is only one syntax element with a dotted decimal number, the ? symbol is displayed on the same line as the syntax element, (for example 5? NOTIFY). If there is more than one syntax element with a dotted decimal number, the ? symbol is displayed on a line by itself, followed by the syntax elements that are optional. For example, if you hear the lines 5 ?, 5 NOTIFY, and 5 UPDATE, you know that syntax elements NOTIFY and UPDATE are optional; that is, you can choose one or none of them. The ? symbol is equivalent to a bypass line in a railroad diagram.

- ! means a default syntax element. A dotted decimal number followed by the ! symbol and a syntax element indicates that the syntax element is the default option for all syntax elements that share the same dotted decimal number. Only one of the syntax elements that share the same dotted decimal number can specify a ! symbol. For example, if you hear the lines 2? FILE, 2.1! (KEEP), and 2.1 (DELETE), you know that (KEEP) is the default option for the FILE keyword. In this example, if you include the FILE keyword but do not specify an option, default option KEEP will be applied. A default option also applies to the next higher dotted decimal number. In this example, if the FILE keyword is omitted, default FILE(KEEP) is used. However, if you hear the lines 2? FILE, 2.1, 2.1.1! (KEEP), and 2.1.1 (DELETE), the default option KEEP only applies to the next higher dotted decimal number, 2.1 (which does not have an associated keyword), and does not apply to 2? FILE. Nothing is used if the keyword FILE is omitted.
- * means a syntax element that can be repeated 0 or more times. A dotted decimal number followed by the * symbol indicates that this syntax element can be used zero or more times; that is, it is optional and can be repeated. For example, if you hear the line 5.1* data area, you know that you can include one data area, more than one data area, or no data area. If you hear the lines 3*, 3 HOST, and 3 STATE, you know that you can include HOST, STATE, both together, or nothing.

  Notes:
  1. If a dotted decimal number has an asterisk (*) next to it and there is only one item with that dotted decimal number, you can repeat that same item more than once.
  2. If a dotted decimal number has an asterisk next to it and several items have that dotted decimal number, you can use more than one item from the list, but you cannot use the items more than once each. In the previous example, you could write HOST STATE, but you could not write HOST HOST.
  3. The * symbol is equivalent to a loop-back line in a railroad syntax diagram.
- + means a syntax element that must be included one or more times. A dotted decimal number followed by the + symbol indicates that this syntax element must be included one or more times; that is, it must be included at least once and can be repeated. For example, if you hear the line 6.1+ data area, you must include at least one data area. If you hear the lines 2+, 2 HOST, and 2 STATE, you know that you must include HOST, STATE, or both. Similar to the * symbol, the + symbol can only repeat a particular item if it is the only item with that dotted decimal number. The + symbol, like the * symbol, is equivalent to a loop-back line in a railroad syntax diagram.

How to send your comments to IBM

We appreciate your input on this publication. Feel free to comment on the clarity, accuracy, and completeness of the information or give us any other feedback that you might have.

Use one of the following methods to send us your comments:
1. Send an email to s390id@de.ibm.com
2. Visit the SA z/OS home page at http://www.ibm.com/systems/z/os/zos/features/system_automation/
3. Visit the Contact z/OS web page at http://www.ibm.com/systems/z/os/zos/webqs.html
4. Mail the comments to the following address:
   IBM Deutschland Research & Development GmbH
   Department 3248
   Schoenaicher Str. 220
   D-71032 Boeblingen
   Federal Republic of Germany
5. Fax the comments to us as follows:
   From Germany: 07031-16-3456
   From all other countries: +(49)-7031-16-3456

Include the following information:
- Your name and address
- Your email address
- Your telephone or fax number
- The publication title and order number:
  IBM Tivoli System Automation for z/OS V3R3.0 TWS Automation
  Programmer's Reference and Operator's Guide
  SC34-2579-00
- The topic and page number related to your comment
- The text of your comment.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

IBM or any other organizations will only use the personal information that you supply to contact you about the issues that you submit.

If you have a technical problem

Do not use the feedback methods listed above. Instead, do one of the following:
- Contact your IBM service representative
- Call IBM technical support
- Visit the IBM zSeries support web page at www.ibm.com/systems/z/support/.

About This Book

This book describes how to customize and operate TWS Automation. The TWS Automation part of IBM® Tivoli® System Automation for z/OS (SA z/OS) brings together batch and online console automation to a common focal point. TWS Automation automates, simplifies, and standardizes console operations and the management of component, application, and production related tasks.

Who Should Use This Book

This book is intended for the following user groups:

- System programmers, system designers, and application designers who will customize TWS Automation.

  For these users, all three parts of the book will be of interest.

  Installing and customizing TWS Automation requires a programmer's understanding of NetView®, TWS, SA z/OS, and TWS Automation, because most of the definitions take place in these programs. Also, you will modify JCL, command lists, and programs for some of the automation functions
- Operators and administrators who manage and monitor TWS.

  For operators, a working knowledge of TWS will be assumed.

What's in This Book?

This book contains the following:

Part 1, "Introducing TWS Automation"
Explains some main concepts and describes the functions of TWS Automation.

Part 2, "Operator's Guide"
Describes the actions that an operator can perform with TWS Automation commands.

Part 3, "Programmer's Reference"
This part describes the information needed to install and customize TWS Automation and the programming interface of TWS Automation.

Notation for Format Descriptions

The reference sections of this manual contain format descriptions of commands and of entries in the SA z/OS policy database. The notation used for these descriptions is as follows:

- Items shown in braces { } represent alternatives. You must choose one. For example,

  ```
  {A|B|C}
  ```

  indicates that you must specify one item only: A, B, or C.
- Items shown in brackets [ ] are optional. You may choose one. For example,

  ```
  [A|B|C]
  ```

  indicates that you may enter A, B, or C, or you may omit the operand.
- A series of three periods (...) indicates that a variable number of items may be included in the list.

© Copyright IBM Corp. 1990, 2010

- An underscored item shows the default that the system will choose if you do not specify an item. For example,

  ```
  [A|B|C]
  ```

  indicates that if no operand is specified, B is assumed.
- Lowercase italicized items are variables; substitute your own value for them.
- Uppercase items must be entered exactly as shown.
- Parentheses must be entered as shown.
- Where operands can be abbreviated, the abbreviations are shown in capital letters. For example, ALL can be entered as A or ALL.
- Commas are used as delimiters between parameters. The last parameter does not require a comma after it. Because of this, we place the comma in front of a parameter to show that if you add this parameter, you need a comma, as for example in

  ```
  XYZ    [A[,B[,C]]]
  ```

  However, the comma follows the preceding parameter and needs to be on the same line as that parameter.

Related Publications

The System Automation for z/OS Library

Table 1 shows the information units in the System Automation for z/OS library:

Table 1. System Automation for z/OS Library

Title	**Order Number**
IBM Tivoli System Automation for z/OS Planning and Installation	SC34-2571
IBM Tivoli System Automation for z/OS Customizing and Programming	SC34-2570
IBM Tivoli System Automation for z/OS Defining Automation Policy	SC34-2572
IBM Tivoli System Automation for z/OS User's Guide	SC34-2573
IBM Tivoli System Automation for z/OS Messages and Codes	SC34-2574
IBM Tivoli System Automation for z/OS Operator's Commands	SC34-2575
IBM Tivoli System Automation for z/OS Programmer's Reference	SC34-2576
IBM Tivoli System Automation for z/OS Product Automation Programmer's Reference and Operator's Guide	SC34-2569
IBM Tivoli System Automation for z/OS TWS Automation Programmer's Reference and Operator's Guide	SC34-2579
IBM Tivoli System Automation for z/OS End-to-End Automation Adapter	SC34-2580
IBM Tivoli System Automation for z/OS Monitoring Agent Configuration and User's Guide	SC34-2581

The System Automation for z/OS books are also available on CD-ROM as part of the following collection kit:

IBM Online Library z/OS Software Products Collection (SK3T-4270)

SA z/OS Home Page

For the latest news on SA z/OS, visit the SA z/OS home page at http://www.ibm.com/systems/z/os/zos/features/system_automation

Related Product Information

You can find books in related product libraries that may be useful for support of the SA z/OS base program by visiting the z/OS Internet Library at http://www.ibm.com/systems/z/os/zos/bkserv

Using LookAt to look up message explanations

LookAt is an online facility that lets you look up explanations for most of the IBM messages you encounter, as well as for some system abends and codes. Using LookAt to find information is faster than a conventional search because in most cases LookAt goes directly to the message explanation.

You can use LookAt from these locations to find IBM message explanations for z/OS elements and features, z/VM®, z/VSE®, and Clusters for AIX® and Linux:

- The Internet. You can access IBM message explanations directly from the LookAt Website at www.ibm.com/systems/z/os/zos/bkserv/lookat/index.html
- Your z/OS TSO/E host system. You can install code on your z/OS or z/OS.e systems to access IBM message explanations using LookAt from a TSO/E command line (for example: TSO/E prompt, ISPF, or z/OS UNIX System Services).
- Your Microsoft Windows workstation. You can install LookAt directly from the *z/OS Collection* (SK3T-4269) or the *z/OS and Software Products DVD Collection* (SK3T-4271) and use it from the resulting Windows graphical user interface (GUI). The command prompt (also known as the DOS > command line) version can still be used from the directory in which you install the Windows version of LookAt.
- Your wireless handheld device. You can use the LookAt Mobile Edition from www.ibm.com/systems/z/os/zos/bkserv/lookat/lookatm.html with a handheld device that has wireless access and an Internet browser (for example: Internet Explorer for Pocket PCs, Blazer or Eudora for Palm OS, or Opera for Linux handheld devices).

You can obtain code to install LookAt on your host system or Microsoft Windows workstation from:

- A CD-ROM in the *z/OS Collection* (SK3T-4269).
- The *z/OS and Software Products DVD Collection* (SK3T-4271).
- The LookAt Website (click **Download** and then select the platform, release, collection, and location that suit your needs). More information is available in the LOOKAT.ME files available during the download process.

Summary of Changes for SC34-2579-00

This document contains information previously presented in System Automation for z/OS V3R2.0 TWS Automation Programmer's Reference and Operator's Guide, SC33-8269-04.

You may notice changes in the style and structure of some content in this document—for example, headings that use uppercase for the first letter of initial words only, and procedures that have a different look and format. The changes are ongoing improvements to the consistency and retrievability of information in our documents.

This document contains terminology, maintenance, and editorial changes. Technical changes or additions to the text and illustrations are indicated by a vertical line to the left of the change.

The "Readers' Comments - We'd Like to Hear from You" section at the back of this publication has been replaced with a new section "How to send your comments to IBM" on page xv. The hardcopy mail-in form has been replaced with a page that provides information appropriate for submitting comments to IBM.

Part 1. Introducing TWS Automation

This part describes some main concepts of SA z/OS, including some NetView-related information, and gives an overview of the facilities offered by TWS Automation.

Subtopics:

© Copyright IBM Corp. 1990, 2010

Chapter 1. Functions of TWS Automation

This chapter describes the basic concept of TWS Automation, explains some aspects of its implementation, and sketches possible uses of TWS Automation.

For information about SA z/OS refer to *IBM Tivoli System Automation for z/OS User's Guide*.

Basic Concepts

TWS Automation is an extension of SA z/OS that capitalizes on the strengths of NetView, SA z/OS, and TWS by providing the ability to greatly expand job execution, scheduling, monitoring, and alert notification capabilities.

Tivoli Workload Scheduler for z/OS Automation consists of two basic functions:

1. Requests from TWS to SA z/OS and associated status updates
2. Requests from SA z/OS to TWS and associated status updates

TWS to SA z/OS Functions

Tivoli Workload Scheduler needs to be able to request desired state changes to subsystems under the control of SA z/OS. There are three interfaces that allow you to do this:

1. A request interface
2. A *command* request interface
3. A command or batch interface

Conventional Request Interface

This interface uses general workstation named NV*xx* and the TWS exit EQQUX007 to pass a request via the NetView PPI to SA z/OS. NV*xx* workstations can make full use of the TWS scheduling capabilities: they can have predecessor links, time dependencies and special resources.

Request types are START, STOP, CANCEL, or any user defined type. The command behind the request type is defined in the SA z/OS policy database. The job name is used to determine the subsystem that is associated with the request. There can be multiple commands associated with the request type. In any case, the subsystem as well as the request types must be predefined in the policy database. For the standard request types START, STOP, and CANCEL, the appropriate default command is issued when no corresponding entry is found in the policy database.

The Tivoli Workload Scheduler status change exit supports WTO'd status changes. The purpose of the exit is to provide information to SA z/OS, which in turn will notify operators via NMC and SDF.

Command Request Interface

This interface uses an *automation* workstation and the TWS exit EQQUXSAZ to pass a command request via the NetView PPI to SA z/OS. Automation workstations can make full use of the TWS scheduling capabilities: they can have predecessor links, time dependencies and special resources.

A TWS user can use the command request interface to issue free-format SA z/OS or NetView commands (up to 255 characters) from the TWS application description panels. The command must be pipeable. Other information can be specified that is necessary to execute the command (the automated function and security element name) and to return whether or not the command executed successfully (maximum wait time, maximum acceptable return code, user-supplied completion checking routine).

This gives the user the ability to manipulate application groups or monitoring resources in the same way as for subsystems, providing out-of-box integration with no installation setup being required. It allows for the definition of automation requests in TWS without the need for the SA z/OS policy database to be updated.

Command Interface

The command interface is designed to work with TWS in a way that is natural for TWS. It takes the form of a batch job that can be used to execute any NetView or SA z/OS command on any NetView interconnected in the enterprise.

The batch job may execute on any system in the sysplex that contains an SA z/OS Agent or NetView Agent running the TWS PPI batch command receiver. This command receiver is a NON-MVS SA z/OS subsystem and may be controlled via the same interfaces as any other SA z/OS subsystem.

SA z/OS to TWS Functions

System Automation for z/OS needs to be able to control the operation of Tivoli Workload Scheduler. TWS operations can be made to wait until a previously requested SA z/OS state change has occurred. SA z/OS will make subsystem, application group, system, system group and monitor resources statuses available to TWS. This is done by reflecting the SA z/OS status in a pair of TWS Special Resources. TWS operations may be coded to wait until the appropriate special resource is in the desired state, thus preventing the batch job stream from proceeding until a TWS request to SA z/OS has been completed.

The INGOPC/INGTWS command allows an SA z/OS operator or automation function to request changes to the TWS current plan. This interface may be used for any Controller defined to SA z/OS or any Tracker (defined to SA z/OS) that represents a *foreign* Controller. A *foreign* Controller is one that resides outside the sysplex or that is not defined to SA z/OS.

Obtaining Information from TWS

The application program interface of OPC/ESA allows TWS Automation to act directly on the TWS current plan. Using this interface, TWS Automation directly requests and updates TWS-based information.

Recovery operations provide one possible use of the TWS API by TWS Automation. For example, if a communications link to a target NetView is not available, the operator can use the OPCACOMP or OPCAPOST command to manually post events that have occurred.

You can also use this interface when manual intervention is required, for example, when a user sequence error is detected. Use the INGOPC REQ=LIST operator command to manually access the information from TWS about operations in the current plan through the TWS API. This allows the determination and resolution of the sequence error to occur from a single NetView console.

The host with the TWS Controller task provides the only availability to the TWS API interface. Because of this, all recovery is done only in the NetView on the processor running the TWS Controller task. Sometimes, when a user-written module utilizes this function, another NetView requires the information. If the destination workstation is an *automation* workstation, you can directly define the NetView domain ID. Otherwise, TWS Automation maintains an entry in the policy database to allow TWS Automation to determine the domain ID of the TWS NetView that the request is sent to (CONTROLLER DETAILS policy object, OCS entry type). This entry has the domain ID of the NetView on the processor running the TWS Controller. Thus all the SA z/OS applications can easily determine where to direct TWS Controller requests.

A user-written task or CLIST can also access the TWS Controller. Your own routines can list operations in the current plan with the INGOPC REQ=LIST command. You can modify the data in current plan operations with INGOPC. You can query the TWS calendar with INGOPC. You can synchronize TWS with OPCACOMP, or OPCAPOST if you write your own automation routines, and you can update the status of special resources in TWS with the SRSTAT command. In this way you can trigger operations to run that have a special resource dependency in TWS.

Defining SA z/OS to Tivoli Workload Scheduler

There are no required Tivoli Workload Scheduler definitions.

Conventional Request Interface

To define a TWS request that TWS Automation can use, a workstation that represents the target NetView domain is required. This workstation, which is defined with TWS using the standard TWS dialogs, should be a general, automatic reporting workstation. Its name *must* have the format NV*xx*.

Note: Reserve NV*xx* workstations for TWS Automation unless they are for automation workstations when the command request interface applies. Unpredictable and undesirable results may occur if these workstation names are used for other workstations.

For the request interface, you must define TWS workstations. There is a naming convention for these workstations that the names must begin with NV and are four characters long.

Command Request Interface

The TWS user may optionally define an *automation* workstation that represents the target NetView domain where the command is to be executed.

The workstation destination name addresses the NetView where the command should be processed.

Command Interface

Because the command interface is a normal batch job as submitted by TWS, there are no extra TWS definitions. You may optionally create a batch job submission workstation that is used to submit the batch jobs to run commands against SA z/OS. The advantage in doing this is that the command processor that the batch job runs can automatically stop the workstation if the SA z/OS Agent or NetView Agent is not up or the PPI interface is not responding. When the SA z/OS Agent or NetView Agent starts up it will automatically restart the previously stopped workstation.

Currently only one PPI receiver non-MVS subsystem is provided in the sample add-on policy database. However, you may start more by specifying a different PPI receiver name for each one to be started. Note, when using multiple command receivers, each must run on a unique dedicated autotask. Batch jobs have a parameter that can be used to select the appropriate PPI receiver to execute the commands against. By specifying multiple TWS workstations, one per PPI receiver, TWS can schedule batch jobs to the appropriate receiver and thus get a better throughput of requests. However, only one of the workstations will be restarted when the SA z/OS Agent or NetView Agent starts up. The user can use TWS variable substitution to resolve the name of the PPI receiver from the name of the Workstation the job is assigned to.

Defining Tivoli Workload Scheduler to SA z/OS

The Tivoli Workload Scheduler Installation manual insists that each TWS subsystem in a sysplex is uniquely named. Detailed instructions for defining this configuration can be found in *IBM Tivoli System Automation for z/OS Defining Automation Policy*.

SA z/OS supplies a sample policy, *TWS, that provides best practice definitions for running TWS Automation.

Diagrams of the sample policies are provided as PDF files that are located in the USS installation path. The default for this path is: /usr/lpp/ing/doc/policies/.

Tivoli Workload Scheduler consists of a number of MVS™ and non-MVS subsystems that need to be defined to SA z/OS. These are:

1. The TWS Controller
2. The TWS Tracker
3. The TWS Server
4. The TWS Data Store
5. The SA z/OS Batch Job Command Receiver
6. The SA z/OS Request Receiver
7. The SA z/OS Status Observer

The TWS Controller

TWS Controller subsystems may be defined to start simultaneously across multiple systems in a sysplex. Controller subsystems may be defined in a sysplex group, or in a system group that is attached to multiple systems. Controller subsystems require Tracker subsystems to manage the batch job stream.

If multiple controllers are running on the same system they must share the same Controller Details (OCS) definitions. If more than one OCS policy object is linked to the same system, the last is used.

Note that a sysplex move group has been defined in the *TWS add-on policy that is used for backup rather than a hot standby as normal in TWS Automation.

The TWS Tracker

TWS Tracker subsystems may be defined to start simultaneously across multiple systems in a sysplex.

The TWS Server

TWS Server subsystems are usually automatically started by the Controller subsystem that has taken on the ACTIVE role.

The TWS Data Store

The data store is a separate address space. It collects structured information (steps and data sets) and, optionally, unstructured information (SYSOUT) for all submitted jobs. Typically, a data store is installed for each JES spool in a system. In a simple JES configuration this would mean one data store for each tracker.

The TWS DS is started as a normal SA z/OS subsystem. TWS Data Store contain SYSOUT information collected from JES on behalf of the active Controller.

The SA z/OS Batch Job Command Receiver

These subsystems are non-MVS NetView subsystems that run in an SA z/OS Agent or NetView Agent. They provide PPI communications for servicing Batch Job Commands. This allows a batch job to execute a NetView or SA z/OS command and to get the output of the command back at the batch job.

The SA z/OS Request Receiver

These subsystems are non-MVS subsystems that run in the SA z/OS NetView Agent. They provide PPI communications to the TWS Controller that allow the TWS Controller to inject requests or commands to SA z/OS. This allows TWS to control the status of resources being managed by SA z/OS.

The SA z/OS Status Observer

SA z/OS TWS Automation provides a facility that echoes the status of SA z/OS resources in TWS Special Resources. This facility allows you to define TWS operations that will wait until SA z/OS resources reach a desired state. Currently only two desired states are allowed. The UP state is when the automation manager sets the resource to the AVAILABLE state. The DOWN state is when the automation manager sets the resource to the UNAVAILABLE state.

Support for Multiple TWS Controllers

TWS Automation offers support if you have two or more active controllers in the same sysplex. This may be with either multiple active controllers on different systems or on the same system.

When there are multiple active controllers on different systems, all components (that is, those receiving requests or commands via the various interfaces and the User interfaces) work for all the active controllers. In particular, the status observer sends special resource updates to all controllers.

There are two scenarios:

1. The systems can be artificially partitioned with a controller assigned to each partition. For example, a sysplex A, might have 4 systems, S1, S2, S3, S4. Systems S1 and S2 might have their batch jobs controlled by controller TWSA; S3 and S4 by controller TWSB. There are no trackers from TWSA on the S3 and S4 systems, and likewise for TWSB on S1 and S2. Thus from the TWS standpoint, these are two independent sets of systems. In this case, SA z/OS report events to both controllers.
2. The systems are as above but there are trackers for all controllers on all systems. In this case SA z/OS need only report the events on one system.

When there are multiple controllers on the same system, SA z/OS reports events to the controllers in such a way as to ensure that all events are reported to all controllers.

Note that if multiple controllers are running on the same system they must share the same Controller Details (OCS) definitions. If more than one OCS policy object is linked to the same system, the last is used.

Communication Flow

Figure 1 shows the communication flow between TWS and SA z/OS with multiple TWS controllers. The shadow under the controller and tracker represents multiple controllers and trackers.

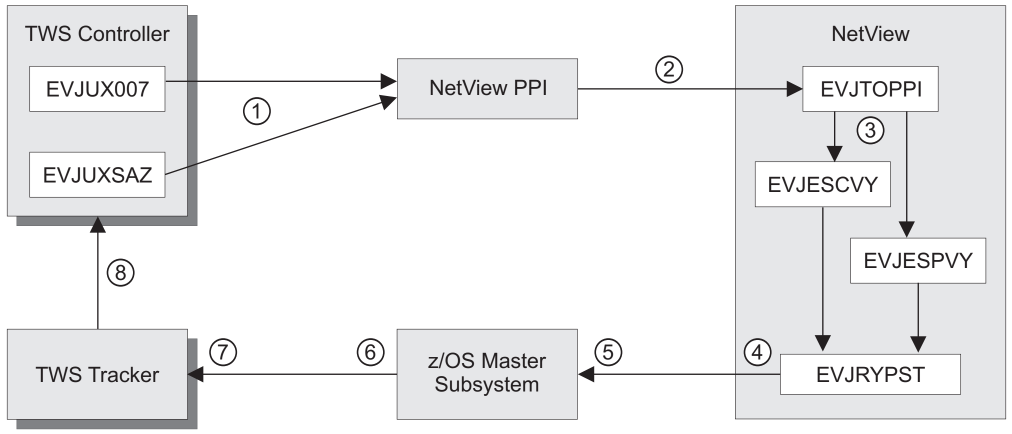


Figure 1. Multiple TWS Controller Communication Flow

The flow is as follows:

1. A TWS controller calls one of the TWS exits (EQQUX007 or EQQUXSAZ) to send a command or a request to SA z/OS for action. These exits can support multiple controllers by running from multiple address spaces in the same z/OS image.
2. The exits pass their information via NetView PPI to the EVJTOPPI receiver task.

   They create unique 8-character PPI sender IDs that consist of the 3-character exit name (either 007 or SAZ) followed by a 4-character address space ID, separated by the # symbol, for example, 007#003D. This prevents a possible clash of PPI receiver IDs should both sets of exits be executing in parallel.
3. EVJTOPPI validates the PPI buffer that it has received and sends the command to the appropriate routine: EVJESPVY for conventional request types or EVJESCVY for command request types.
4. After processing the request, the OPCAPOST or EVJRYPST routine (for conventional requests or command requests, respectively) posts the success or failure of the request or command to TWS.

   Note that because this is posted to all trackers that are running on the SA z/OS that processes the request, the job name for the operation is passed as an additional qualifier for the request to OPCAPOST or EVJRYPST. It is the responsibility of the installation to ensure that the TWS operation is uniquely identified. SA z/OS uses the following attributes to identify the TWS operation:
   - Application name
   - Workstation name
   - Operation number
   - Job name (if present)

- IA time

5. EVJRYPST formats an EQQUSIN buffer and sends this to the z/OS Master Subsystem.
6. The z/OS Master Subsystem sends a copy of this buffer to every z/OS subsystem that has registered for the data.
7. The tracker (and possibly also the controller) will have registered for the TWS buffer data that EQQUSIN created. These address spaces receive a copy of the buffer.
8. If the tracker receives the data, it sends it on to its owning controller, which might be on another system. The controller now marks the operation complete or in error.

Thus all POSTs to TWS go to all trackers on the system that the command or request was eventually executed on. It does not matter how many controllers or trackers are running on a system, they will all receive the POST.

TWS topology requires a tracker to be running on every system that SA z/OS might dispatch a command or request to. There must therefore be a tracker for each of the different controllers. So if you have two controllers TWCA and TWCB you will need a tracker from each to be present on the system. If the trackers are TWTA and TWTB and there are two systems SYS1 and SYS2, then both TWTA and TWTB must be running on both systems. The controllers can run anywhere, even on foreign systems.

Concurrent Requests and Commands

SA z/OS handles concurrent requests and commands differently:

- If two or more *requests* arrive for the same subsystem from different controllers, the first request is processed but the second and subsequent requests fail until the first request has completed processing. There is no attempt to queue the requests; each request is processed as soon as it is received. An existing outstanding request will prevent new requests from being processed and will post a completion code of U005.
- If two or more *commands* arrive for the same subsystem from different controllers, SA z/OS attempts to execute them simultaneously. This may result in one of the commands being timed out due to not reaching its desired status for the subsystem. It is possible that both commands require the same desired status, and in such cases both commands will probably succeed.

  It is not possible to serialize commands for the same resource because the command text is free form. In addition there may be intersections that result in command conflicts. For example, if one command stops a parent and another starts a child. The system relies on the timeout function to resolve these situations and will as a result report one of the commands in error.

Note that creating TWS operations with the same workstation, ADID, operation number and job name might result in unpredictable effects. This is because when SA z/OS posts the result of the operation, all operations with the same ID on all active controllers will also be posted.To avoid this, you should ensure that the ADID (Application Identification) is unique.

The ACF $$$OPCA DOMAINID entry, which is defined in the OPC Workstation Domains policy object in the customization dialog panels, must specify workstations for all active controllers. An installation can either use the same workstation names for different active controllers or it can specify different names that resolve to the same set of domain IDs. For example, active controller 1 might use NV11 for domain ID IPSFM, and active controller 2 might use NV21 for the same domain ID. This is however limited because the first two characters of the workstation name must be NV.

SA z/OS Status Updates to TWS Special Resources

The TWS status observer is represented by a dummy subsystem that is responsible for registering and deregistering the resources that are specified for status tracking in the $$$OPCA SRSTAT fragment.

It is possible that one or more of the controllers may not receive special resource updates. This is because the TWS observer must run on at least one system and the trackers for the two controllers may not intersect. That is, there is no system where all the trackers for the multiple controllers are running. In this case, you will need to run *multiple* TWS observers.

TWS observers thus allow multiple registrations and the subsystem should be added to a server group. You should use SA z/OS server groups and preferences to ensure that the multiple instances of the TWS observer run on the appropriate systems. Each TWS observer subsystem should have a HasParent relationship to the tracker on the same system as the TWS observer runs on.

This allows you to specify the most flexible configuration. Should a tracker fail, the observer will be moved to the most appropriate system that satisfies its HasParent relationship and has the highest preference.

A server or move group can be used to ensure that there is one observer per set of trackers.

NetView Interface to TWS Automation

The program-to-program interface (PPI), a high-performance interface, provides synchronization and bidirectional command and message flow between NetView and other applications. TWS provides additional application programming interfaces (APIs), which allow it to be updated by other programs.

The implementation of these interfaces in TWS Automation provides the following capabilities:

- Automation of TWS startup and termination
- Interception of TWS alerts for analysis by the operator
- Expansion of the Status Display Facility to provide information about TWS errors
- Implementation of a two-way interface between TWS and NetView with SA z/OS:
  - TWS defines and controls interactive applications. Support is provided to start and stop subsystems that are defined to the SA z/OS application.
  - Database tasks can run in both interactive and batch systems with full synchronization between the activities.
  - SA z/OS operators can access TWS calendars and other information as well as update TWS information using the INGOPC or INGTWS command.
- Two user extensions:
  - OPCACOMP allows the start up and shut down of subsystems independently of automation status changes.

– UX*xxxxxx* allows automation of activities not associated with a specific subsystem.

TWS Automation provides commands and panels that allow a NetView operator to make inquiries and issue requests to TWS without actually logging on to TWS.

TWS Automation Special Resources

TWS Automation is able to globally control the creation and setting of TWS special resources based on the status of SA z/OS-monitored subsystems. This will allow TWS to resolve job scheduling dependencies based on the status of SA z/OS managed resources.

The TWS special resources created or set by this function are:

`ING.`*`res_sys`*`.`*`res_type`*`.`*`res_name`*`.UP`, and
`ING.`*`res_sys`*`.`*`res_type`*`.`*`res_name`*`.DOWN`

where:

res_sys	is the MVS sysid of the system where the subsystem status change was detected. If the resource is a SYSPLEX application group, the value SYSPLEX is used.
res_type	is one of APL, APG, SYS, SYG, GRP, or MTR.
res_name	is the name of the resource.
UP	is a literal that defines the resource as being available only when the resource has an observed status of AVAILABLE and a desired status of AVAILABLE.
DOWN	is a literal that defines the resource as being available only when the observed status of the resource is one of the following Automation Manager statuses: SOFTDOWN, HARDDOWN, STANDBY, UNKNOWN, SYSGONE, and when its desired status is UNAVAILABLE.

Possible Uses of TWS Automation

In data centers, certain groups assume responsibility for the daily operation of the systems. Frequently, these groups are split into these two areas that perform the following tasks:

- Controlling online systems
- Processing all batch work

User requests for hours of service form the basis for online planning decisions. The time available to process the jobs required for online systems for the next day, as well as requests for other batch work, determines batch processing.

A system using TWS executes the current plan (CP), which contains the information for batch processing. A help desk, hotline, or service-contact point merge user-change requests into the overall schedule. While processing control executes batch processing, operations or master terminal operators control the online systems, thus adding to the confusion. Changes to online availability are frequently manual in nature. For example, instructions to change online availability often consist of slips of paper or phone calls to the operator.

TWS Automation allows changes that influence both batch and online systems through simple TWS dialogs. Because TWS manages both batch and online systems, these changes are needed only in one place. Because the processes are automated with SA z/OS and TWS, no interoperator communications are required. In fact, in a highly automated environment, no operator intervention or awareness of these user-requested changes is necessary.

TWS Automation can automate some of the more complex operator procedures and thereby provide several new functions. The following topics give some examples and scenarios that demonstrate these functions.

Changing Online Hours of Availability

Several possible methods exist for changing the hours of availability of online services. To illustrate these methods, consider the example of a service such as IMS™. Assume that the scheduled hours of availability for the IMS online service are 7 a.m. to 6 p.m. In NetView, under control of the current plan, TWS Automation performs the timed start and stop events.

- The help desk gets a request from a user group to extend the IMS hours of availability, for today only, from the original plan of 6 p.m. to 8 p.m. (extended service period).
- The help desk ensures that this extended service is acceptable within the service level agreement for this user group.
- The help desk now makes a change to the TWS current plan to reflect this extended IMS period.
- When the revised scheduled time is reached, now two hours later than usual, TWS executes the operation, requesting that TWS Automation stop IMS.
- TWS Automation requests that SA z/OS application stops IMS.
- Once SA z/OS has successfully stopped IMS, TWS Automation returns an operation-ended status to TWS, fulfilling TWS's dependencies on the online IMS.
- Jobs dependent on the termination of IMS are now released for execution.

No restructuring of the batch processing is necessary if the request is within planned service bounds.

Cycling Individual Online Databases

TWS Automation allows TWS to interact not only with the SA z/OS functions, but also with the MVS and MTO consoles, which enables the scheduling of interrelated sequences of events. For example, it is possible to cycle individual databases rather than the complete online system. Figure 2 on page 13 shows how this scheduling results in minimum disruptions to online applications.

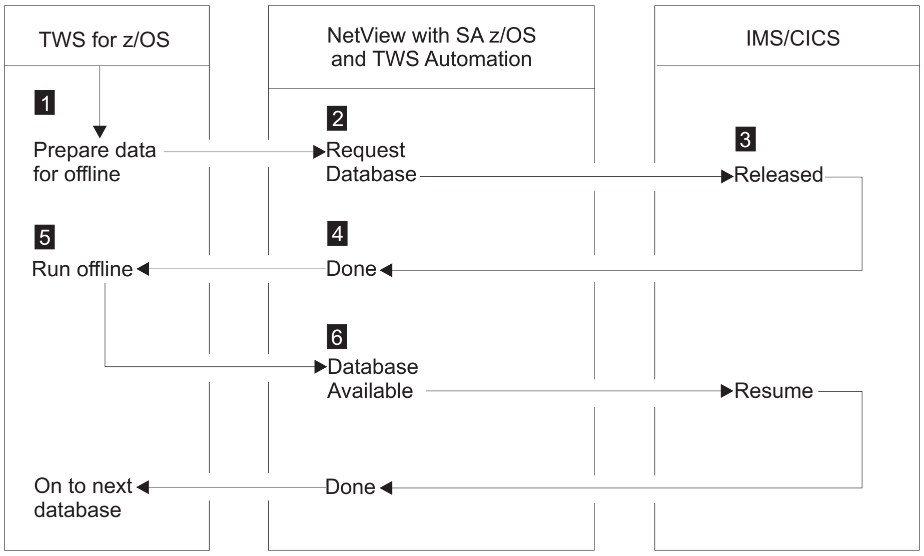


Figure 2. Example of Cycling Individual Online Databases

In Figure 2, the online databases are structured so that you can vary specific ones offline, without an impact to the system, as in the case of databases structured on a geographic or application basis. This process flows as follows:

1. Based on the current plan, TWS begins the READY TO START DATABASE UPDATE job.
2. A request is sent to TWS Automation to issue the command required to vary the subject database offline to allow for batch processing.
3. The request is issued through the MTO interface.
4. TWS Automation ensures that the database is offline.
5. TWS Automation posts the operation as completed in TWS.
6. With the operation completed, TWS dependency starts the batch processing for this database.
7. When the batch process is completed, TWS once again triggers TWS Automation, and the proper MTO command is issued to vary the database online to IMS.

When individual databases accomplish this type of process, the database/data communications (DB/DC) system is always up, and certain small portions of the data is unavailable for short periods. In some cases, you can restructure the databases to further shorten periods of data unavailability.

TWS Automation does not directly support the preceding example, which requires some user-written modules. (See “Interaction with CICS Automation” on page 129 and “Interaction with IMS automation” on page 130 for some examples of this type of user-written module.) TWS Automation transports the request to the appropriate system and prepares the information for the user code. TWS Automation then returns the resulting status to the operation in TWS. TWS Automation also ensures that the actions requested are serialized with other requests to that specific target subsystem and that the status of the subsystem is such that it can accept the requests.

Scheduling Time for Testing

Another example is an automated mechanism that prepares a logically partitioned mode (LPAR) on a process resource/system management (PR/SM™) complex for testing periods.

In this example, a system programmer or application developer makes a request through the help desk for testing. The help desk checks that the resources for the test period are available and invokes a prepared TWS-controlled application, updating the information required to set up the time and duration of the test. No other action is needed.

At the proper time, TWS begins execution. It sends the requests to the target system control facility (processor operations) application to set up the LPAR for the test period, and to IML and IPL the PR/SM partition. If the requestor of the test period prepares the test system, so it is ready and waiting at the start of the test period, there is no waiting for an operator to set up the test environment or to structure the system as required by the testing.

Distributing and Updating Data Across Multiple Systems

As centralization of operations and support progresses, preparing data at a central site and then distributing it to other systems becomes necessary. Controlled execution of batch utilities is often required to update the target systems.

Installation of system maintenance provides an example of this type of distribution. Program temporary fixes (PTFs) are installed and tested at a central site. The PTFs are then shipped to target systems and applied with a system modification program (SMP/E). Frequently, a system programmer performs this by logging on to the target system and executing the job streams manually.

Another example is the creation of office system files on a central system, such as electronic telephone directories. These files are then distributed to the target systems.

TWS can control network job entry (NJE) jobs for the distribution of data, and thus controls the execution of the jobs on the target systems, to apply the data, using dependency control, if required.

TWS Automation extends this TWS capability and allows necessary cycling of the target system application once the maintenance is applied successfully. You can schedule this in such a way as to minimize any impact on the end-user community. The following is a typical scenario:

1. A PTF is installed, tested, and found acceptable. This PTF is then applied to all copies of TSO in a multisystem environment.
2. The application is defined to TWS. In most cases, the application is simply updated because it is already defined.
3. TWS presents the batch jobs that control the SMP/E process to the systems programmer for modifications, if required.
4. TWS schedules the transmission of the jobs to the target systems using NJE. The scheduling can use a time when network traffic is low.
5. Once the jobs are in the target system, TWS dependency control is used to schedule the SMP/E job execution.
6. TWS ensures that the SMP/E jobs run correctly. If TWS encounters problems, the TWS application provides backout procedures.

7. After installing the PTFs, TWS selects the appropriate time to issue a request to TWS Automation to restart TSO.
8. Because TWS fully controls the process for this PTF update, you can inquire at any time to see the progress of the operations. If errors or problems occur, TWS Automation informs the SA z/OS notification operator.

Complex Application Recovery

As computer applications become more critical to the daily operation of your enterprise, disaster recovery takes on an added significance. Usually, installations have the necessary equipment and facilities for disaster recovery, but the operational processes are so complicated that the chance of a successful backup in a short period, lasting from many minutes to no more than a few hours, is highly unlikely.

TWS Automation allows full or partial automation of this type of activity between systems and sites. In some cases, changing the NV*xx*-to-NetView domain ID relationship is adequate to transfer the control of the work load to a different system. However, the change may require some manual intervention for synchronization. Chapter 15, "Resynchronization and Recovery Considerations," on page 133 discusses several scenarios and the process of synchronization.

Although not all steps are required each time, most recoveries consist of three major operational steps that are executed sequentially and provide the following functions:

Step 1: Preparing the recovery system
This may require stopping some or all the applications on the recovery systems, unloading data from disk-storage devices to tape, and reconfiguring the recovery system.

Step 2: Starting the critical applications on the recovery system.
This can include the following:
- Loading databases and applications from tape-to-disk devices
- Starting the recovery system
- Updating data from checkpoint data, logs, or other sources
- Starting critical applications.

Step 3: Returning to the original production system
This is a reversal of the recovery process. These procedures are as complex as the original recovery process, but are scheduled and do not have the urgency of the original recovery.

In the following example, a series of applications need starting on a system after the failure of the original system or possibly even the site. Assume that the installation has prepared properly for this type of problem. This implies tested procedures, current levels of the affected applications and operating environment, and data at the backup site. To simplify this example, assume that the database at the recovery site is adequate for a contingency recovery situation.
- Prior to the need, a series of interdependent recovery applications are defined to TWS, but not scheduled.
- The decision to recover the critical applications at the backup site is made. The scheduler uses normal TWS panels to modify the current plan to schedule the first backup application.
- Before recovery, several factors, which can result in modifications to procedures and JCL, need considering. These modifications are then presented to operators

at manual workstations with instructions in the operator instruction files of TWS. They are also presented to systems programmers at JCL workstations.

- The work load on the recovery system is stopped by scheduling a request to SA z/OS to stop all subsystems other than JES.
- Once the subsystems are stopped, a series of jobs are scheduled to transfer data from disk-to-tape to accommodate the requirements of the critical applications that are recovered.
- Depending on the situation, the same system is reused or restructured, and then followed by an IML and IPL of the recovery system. If this is the case, the focal point implementation option of SA z/OS is used to partially or completely automate this phase of the recovery. Regardless of the specifics, the result is an operating system platform ready to accept the recovery environment.
- TWS schedules a series of JES jobs that restore the databases from backup.
- TWS triggers NetView to issue the appropriate commands to start the subsystem.
- In some cases, NetView requires access to MTO functions to issue specific procedures before the DB/DC system can resume transaction processing. If that is the case, user-provided modules are required to fully automate the recovery.

At this point, the recovery is completed. Normal operating procedures should apply to the environment. Because a recovery situation creates an environment where resources are scarce, the actual applications that are offered are frequently a subset of the normal applications. To accommodate this environment, TWS and TWS Automation may need to change the scheduling of some of the applications controlled by TWS.

After recovery occurs and you resolve the problems that forced the original backup, the applications should be moved back to the original system. The scenario for this move is similar to the one above except that this move is planned instead of forced. This allows you to move specific applications one at a time, as opposed to the all-at-once scenario that a critical situation requires. The fact that some of the applications are moved to an already working system makes the takeback more complex than the original recovery.

Give special consideration to any synchronization procedure in an TWS Automation environment. For more information on the synchronization process, see Chapter 15, “Resynchronization and Recovery Considerations,” on page 133.

Part 2. Operator's Guide

This part describes the actions that an operator can perform with TWS Automation commands.

Subtopics:

© Copyright IBM Corp. 1990, 2010

Chapter 2. Managing the TWS Current Plan

You can use the INGOPC command (or INGTWS, which is a synonym) to list or modify any current plan Application, Operation, Special Resource or Workstation, and to list any current plan Calendar. This function is described in the *IBM Tivoli System Automation for z/OS Operator's Commands*.

Selecting the TWS Controller to Access

The INGOPC command allows you to select the TWS Controller to access via the TWS API.

The positional parameter specifies the SA z/OS resource name of the TWS Controller. If the TWS Controller is in a sysplex and may have active and standby Controllers, you may specify an SA z/OS Application Group that contains the set of TWS Controller resources. The INGOPC command will automatically select the active Controller.

Using Multiple Resource Definitions

The Resource Parameter can take multiple arguments contained in brackets and separated by commas, for example:

```
INGOPC (CTL1/APL/SYS1,CTL2/APL/SYS2)
```

In this case both CTL1 and CTL2 TWS Controller subsystems will be scanned to see which is the Active Controller. The Active Controller is the subsystem that is in the AVAILABLE Automation Manager state and that is marked ACTIVE by the automation manager.

Using Wildcards

The Resource Parameter can take wildcards as defined in the INGLIST command, for example:

```
INGOPC CTL*/APL/*
```

In this case both CTL1 and CTL2 would be scanned as in the example for multiple resource definitions, however, the user specification is shorter.

Using Application Groups

The Resource Parameter may be SA z/OS Application Groups, for example:

```
INGOPC CTLR/APG
```

or:

```
INGOPC CTLG/APG/SYS1
```

Both SYSPLEX and SYSTEM type application groups are allowed. The members of the application groups are found and checked to see if they are TWS type applications. Only the TWS type applications with a subtype of CONTROLLER or TRACKER are checked.

Indirectly Selecting a Controller

In most cases the resource specification will resolve to a single TWS Controller subsystem. However, there are occasions when the Controller subsystem is not present on any system that SA z/OS has access to. In these cases, there must be at least one Tracker on the system or sysplex that SA z/OS is managing. This Tracker must have the "TWS Control" policy with the "Controller ID" and the "API LU Name" policy items specified.

If the INGOPC command cannot resolve the resources that you specify as an active TWS Controller, a final attempt is made to see if any of the resources are a TWS Tracker. If a single OPC Tracker is found in an AVAILABLE state and has an LUNAME policy entry, then this tracker will be used. If multiple controllers or trackers (or both) are found, and INGOPC is running in full screen mode, then a selection list is displayed; when OUTMODE=LINE is specified an error message is displayed.

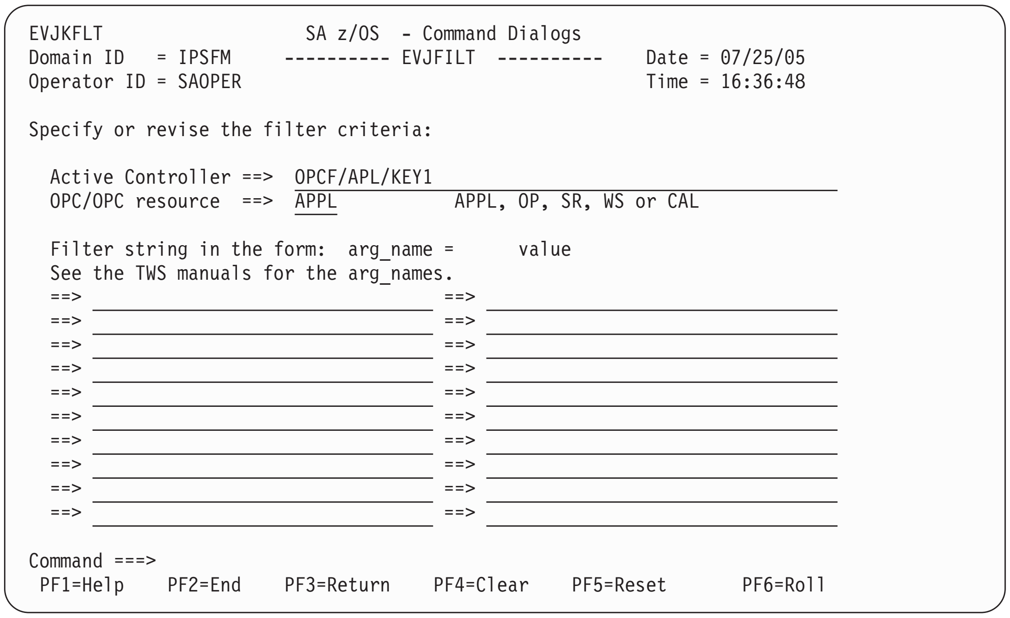
The Tracker selected will be used to refer to a remote Controller via the definitions in the "TWS Control" policy as specified above.

Displaying the Current Plan

To display the Current® Plan, use the INGOPC command with REQ=LIST and specify the type of TWS resource required from:

- APPL for applications
- OP for operations
- SR for special resources
- WS for workstations
- CAL for calendars

On each of the panels that display the current plan, pressing PF5 displays a filter selection panel similar to Figure 3.



```
EVJKFLT                  SA z/OS  - Command Dialogs
Domain ID   = IPSFM     ---------- EVJFILT  ----------    Date = 07/25/05
Operator ID = SAOPER                                      Time = 16:36:48

Specify or revise the filter criteria:

  Active Controller ==>  OPCF/APL/KEY1
  OPC/OPC resource  ==>  APPL           APPL, OP, SR, WS or CAL

  Filter string in the form:  arg_name =      value
  See the TWS manuals for the arg_names.
  ==> ________________________________ ==> _________________________________
  ==> ________________________________ ==> _________________________________
  ==> ________________________________ ==> _________________________________
  ==> ________________________________ ==> _________________________________
  ==> ________________________________ ==> _________________________________
  ==> ________________________________ ==> _________________________________
  ==> ________________________________ ==> _________________________________
  ==> ________________________________ ==> _________________________________
  ==> ________________________________ ==> _________________________________
  ==> ________________________________ ==> _________________________________

Command ===>
 PF1=Help    PF2=End    PF3=Return    PF4=Clear    PF5=Reset       PF6=Roll
```

Figure 3. INGOPC Filter Sample Panel

Specify filter strings in the following format:

field-name op contents

Where:

- *field-name* is a valid field name as specified by the MODIFY command arguments in *Tivoli Workload Scheduler for z/OS Programming Interfaces*.
- *op* can be one of the following:

  = ^= < <= > >=
- *contents* are the desired values to be matched by the *op* operator. The trailing wildcard character '*' may be used for *op*.

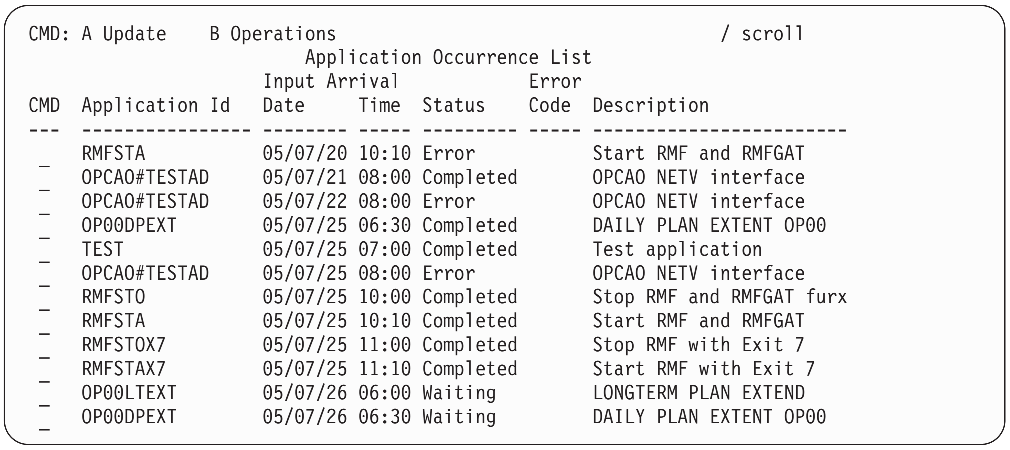
The operands must be separated by a blank.

Displaying TWS Applications

If you specify TYPE=APPL, this will select TWS Applications for display. To reduce the number of applications that are listed, you can use the optional parameters of AD= and IA= to specify the application ID and the Input Arrival time of the application.

In OUTMODE=LINE, a set of messages will be returned that contains all the data for the selected applications.

The fullscreen interface is as shown in Figure 4.



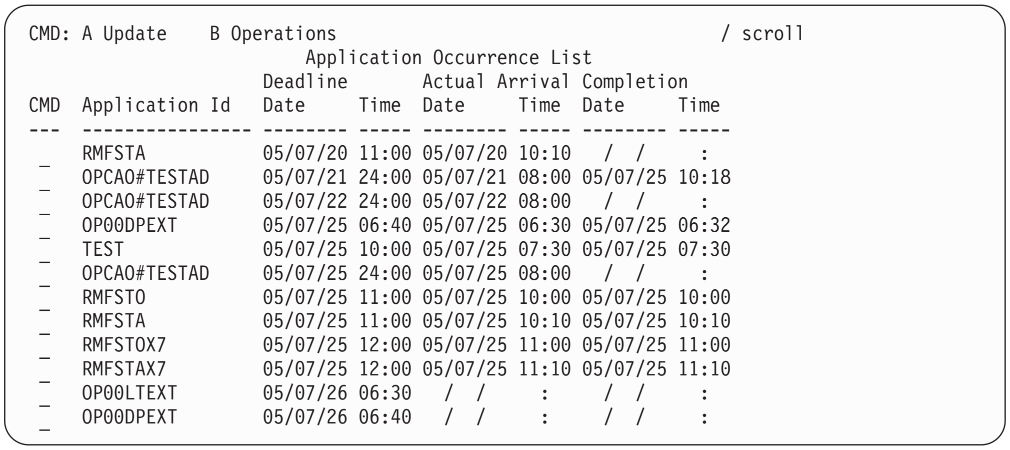
```
CMD: A Update    B Operations                                    / scroll
                            Application Occurrence List
                        Input Arrival            Error
CMD  Application Id   Date     Time  Status    Code  Description
---  ---------------- -------- ----- --------- ----- ------------------------
 _   RMFSTA           05/07/20 10:10 Error           Start RMF and RMFGAT
 _   OPCAO#TESTAD     05/07/21 08:00 Completed       OPCAO NETV interface
 _   OPCAO#TESTAD     05/07/22 08:00 Error           OPCAO NETV interface
 _   OP00DPEXT        05/07/25 06:30 Completed       DAILY PLAN EXTENT OP00
 _   TEST             05/07/25 07:00 Completed       Test application
 _   OPCAO#TESTAD     05/07/25 08:00 Error           OPCAO NETV interface
 _   RMFSTO           05/07/25 10:00 Completed       Stop RMF and RMFGAT furx
 _   RMFSTA           05/07/25 10:10 Completed       Start RMF and RMFGAT
 _   RMFSTOX7         05/07/25 11:00 Completed       Stop RMF with Exit 7
 _   RMFSTAX7         05/07/25 11:10 Completed       Start RMF with Exit 7
 _   OP00LTEXT        05/07/26 06:00 Waiting         LONGTERM PLAN EXTEND
 _   OP00DPEXT        05/07/26 06:30 Waiting         DAILY PLAN EXTENT OP00
```

Figure 4. TWS Applications Interface Panel

You can scroll left and right with the PF11 (Next) and PF10 (Previous) keys. If more data is available than is displayed on the screen, use the PF8 (Down) or PF7 (Up) keys to scroll the data up or down.

For details about the different column values, see the online help or the description of the INGOPC command in *IBM Tivoli System Automation for z/OS Operator's Commands*.

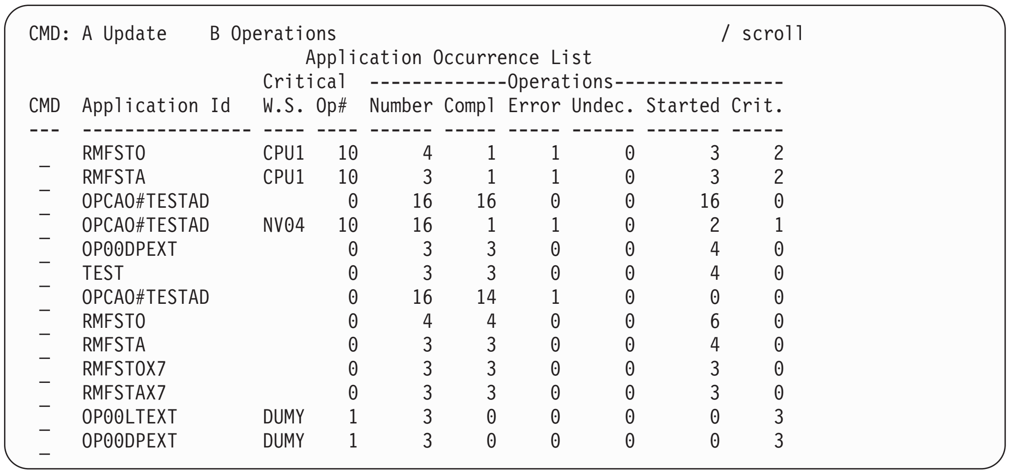
Figure 5 on page 22 is displayed if you press the PF11 key.



```
CMD: A Update    B Operations                                    / scroll
                           Application Occurrence List
                       Deadline       Actual Arrival Completion
CMD  Application Id    Date     Time  Date     Time  Date     Time
---  ----------------  -------- ----- -------- ----- -------- -----
 _   RMFSTA            05/07/20 11:00 05/07/20 10:10   /  /     :
 _   OPCAO#TESTAD      05/07/21 24:00 05/07/21 08:00 05/07/25 10:18
 _   OPCAO#TESTAD      05/07/22 24:00 05/07/22 08:00   /  /     :
 _   OP00DPEXT         05/07/25 06:40 05/07/25 06:30 05/07/25 06:32
 _   TEST              05/07/25 10:00 05/07/25 07:30 05/07/25 07:30
 _   OPCAO#TESTAD      05/07/25 24:00 05/07/25 08:00   /  /     :
 _   RMFSTO            05/07/25 11:00 05/07/25 10:00 05/07/25 10:00
 _   RMFSTA            05/07/25 11:00 05/07/25 10:10 05/07/25 10:10
 _   RMFSTOX7          05/07/25 12:00 05/07/25 11:00 05/07/25 11:00
 _   RMFSTAX7          05/07/25 12:00 05/07/25 11:10 05/07/25 11:10
 _   OP00LTEXT         05/07/26 06:30   /  /     :     /  /     :
 _   OP00DPEXT         05/07/26 06:40   /  /     :     /  /     :
```

Figure 5. TWS Applications Interface Panel, Screen 2

Figure 6 is displayed if you press the PF11 key.



```
CMD: A Update    B Operations                                    / scroll
                           Application Occurrence List
                       Critical  -------------Operations----------------
CMD  Application Id    W.S. Op#  Number Compl Error Undec. Started Crit.
---  ----------------  ---- ---- ------ ----- ----- ------ ------- -----
 _   RMFSTO            CPU1   10      4     1     1      0       3     2
 _   RMFSTA            CPU1   10      3     1     1      0       3     2
 _   OPCAO#TESTAD              0     16    16     0      0      16     0
 _   OPCAO#TESTAD      NV04   10     16     1     1      0       2     1
 _   OP00DPEXT                 0      3     3     0      0       4     0
 _   TEST                      0      3     3     0      0       4     0
 _   OPCAO#TESTAD              0     16    14     1      0       0     0
 _   RMFSTO                    0      4     4     0      0       6     0
 _   RMFSTA                    0      3     3     0      0       4     0
 _   RMFSTOX7                  0      3     3     0      0       3     0
 _   RMFSTAX7                  0      3     3     0      0       3     0
 _   OP00LTEXT         DUMY    1      3     0     0      0       0     3
 _   OP00DPEXT         DUMY    1      3     0     0      0       0     3
```

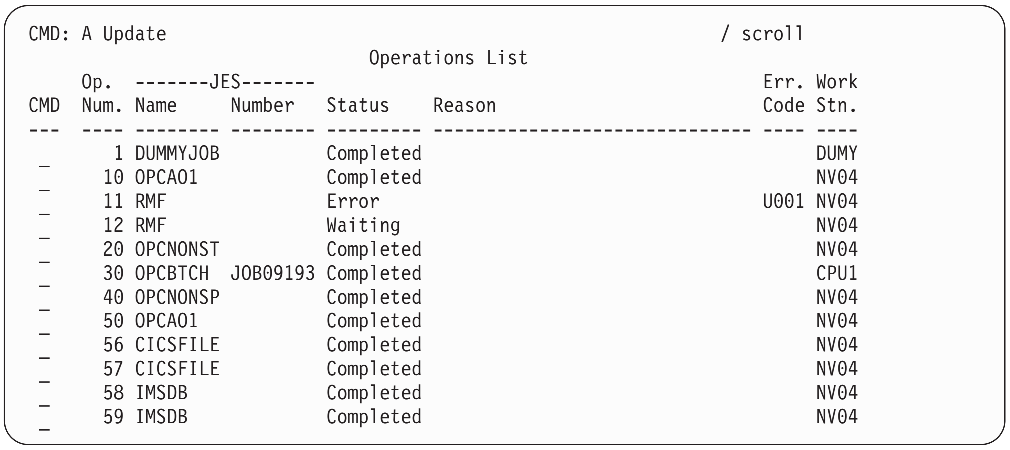
Figure 6. TWS Applications Interface Panel, Screen 3

Displaying TWS Operations

If you specify TYPE=OP, this will select TWS Operations for display. To reduce the number of operations that are listed, you can use the optional parameters of AD=, IA= and OPNO= to specify the application ID, the Input Arrival time of the application and the operation number.

In OUTMODE=LINE, a set of messages will be returned that contains all the data for the selected operations.

The fullscreen interface is shown in Figure 7 on page 23.



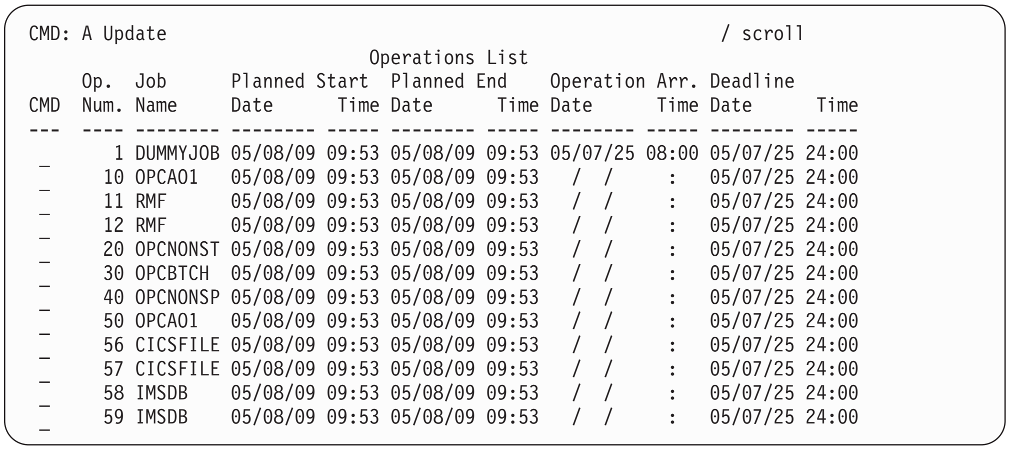
```
CMD: A Update                                                      / scroll
                                Operations List
     Op.  -------JES-------                                            Err. Work
CMD  Num. Name     Number   Status    Reason                           Code Stn.
---  ---- -------- -------- --------- ------------------------------- ---- ----
 _      1 DUMMYJOB          Completed                                       DUMY
 _     10 OPCAO1            Completed                                       NV04
 _     11 RMF               Error                                      U001 NV04
 _     12 RMF               Waiting                                         NV04
 _     20 OPCNONST          Completed                                       NV04
 _     30 OPCBTCH  JOB09193 Completed                                       CPU1
 _     40 OPCNONSP          Completed                                       NV04
 _     50 OPCAO1            Completed                                       NV04
 _     56 CICSFILE          Completed                                       NV04
 _     57 CICSFILE          Completed                                       NV04
 _     58 IMSDB             Completed                                       NV04
 _     59 IMSDB             Completed                                       NV04
```

Figure 7. TWS Operations Interface Panel

You can scroll left and right with the PF11 (Next) and PF10 (Previous) keys. If more data is available than is displayed on the screen, use the PF8 (Down) or PF7 (Up) keys to scroll the data up or down.

For details about the different column values, see the online help or the description of the INGOPC command in *IBM Tivoli System Automation for z/OS Operator's Commands*.

Figure 8 is displayed if you press the PF11 key.



```
CMD: A Update                                                      / scroll
                                Operations List
     Op.  Job      Planned Start  Planned End    Operation Arr. Deadline
CMD  Num. Name     Date      Time Date      Time Date      Time Date      Time
---  ---- -------- -------- ----- -------- ----- -------- ----- -------- -----
 _      1 DUMMYJOB 05/08/09 09:53 05/08/09 09:53 05/07/25 08:00 05/07/25 24:00
 _     10 OPCAO1   05/08/09 09:53 05/08/09 09:53   /  /     :   05/07/25 24:00
 _     11 RMF      05/08/09 09:53 05/08/09 09:53   /  /     :   05/07/25 24:00
 _     12 RMF      05/08/09 09:53 05/08/09 09:53   /  /     :   05/07/25 24:00
 _     20 OPCNONST 05/08/09 09:53 05/08/09 09:53   /  /     :   05/07/25 24:00
 _     30 OPCBTCH  05/08/09 09:53 05/08/09 09:53   /  /     :   05/07/25 24:00
 _     40 OPCNONSP 05/08/09 09:53 05/08/09 09:53   /  /     :   05/07/25 24:00
 _     50 OPCAO1   05/08/09 09:53 05/08/09 09:53   /  /     :   05/07/25 24:00
 _     56 CICSFILE 05/08/09 09:53 05/08/09 09:53   /  /     :   05/07/25 24:00
 _     57 CICSFILE 05/08/09 09:53 05/08/09 09:53   /  /     :   05/07/25 24:00
 _     58 IMSDB    05/08/09 09:53 05/08/09 09:53   /  /     :   05/07/25 24:00
 _     59 IMSDB    05/08/09 09:53 05/08/09 09:53   /  /     :   05/07/25 24:00
```

Figure 8. TWS Operations Interface Panel, Screen 2

Figure 9 on page 24 is displayed if you press the PF11 key.

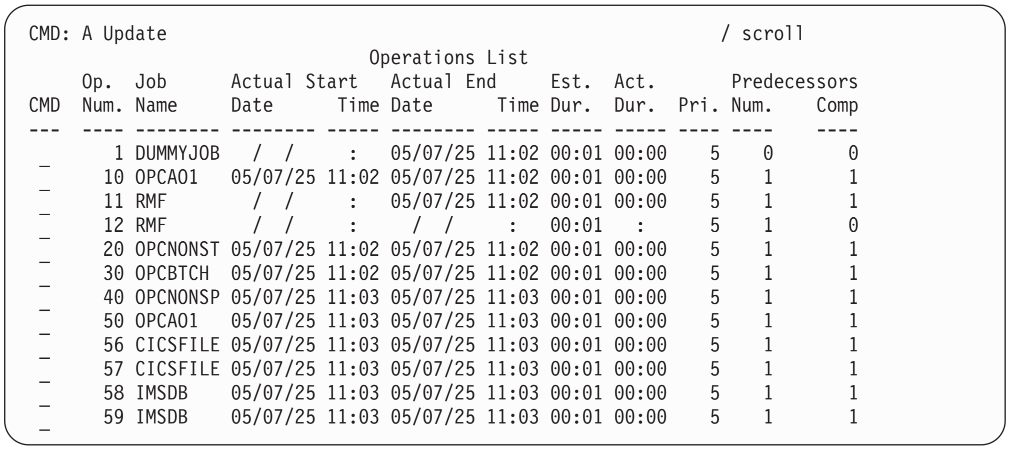

```
CMD: A Update                                                    / scroll
                                Operations List
     Op.  Job      Actual Start   Actual End     Est.  Act.       Predecessors
CMD  Num. Name     Date       Time Date      Time Dur.  Dur.  Pri. Num.    Comp
---  ---- -------- -------- ----- -------- ----- ----- ----- ---- ----    ----
 _      1 DUMMYJOB   /  /     :   05/07/25 11:02 00:01 00:00    5    0       0
 _     10 OPCAO1   05/07/25 11:02 05/07/25 11:02 00:01 00:00    5    1       1
 _     11 RMF        /  /     :   05/07/25 11:02 00:01 00:00    5    1       1
 _     12 RMF        /  /     :     /  /     :   00:01   :      5    1       0
 _     20 OPCNONST 05/07/25 11:02 05/07/25 11:02 00:01 00:00    5    1       1
 _     30 OPCBTCH  05/07/25 11:02 05/07/25 11:02 00:01 00:00    5    1       1
 _     40 OPCNONSP 05/07/25 11:03 05/07/25 11:03 00:01 00:00    5    1       1
 _     50 OPCAO1   05/07/25 11:03 05/07/25 11:03 00:01 00:00    5    1       1
 _     56 CICSFILE 05/07/25 11:03 05/07/25 11:03 00:01 00:00    5    1       1
 _     57 CICSFILE 05/07/25 11:03 05/07/25 11:03 00:01 00:00    5    1       1
 _     58 IMSDB    05/07/25 11:03 05/07/25 11:03 00:01 00:00    5    1       1
 _     59 IMSDB    05/07/25 11:03 05/07/25 11:03 00:01 00:00    5    1       1
```

Figure 9. TWS Operations Interface Panel, Screen 3

Figure 10 is displayed if you press the PF11 key.

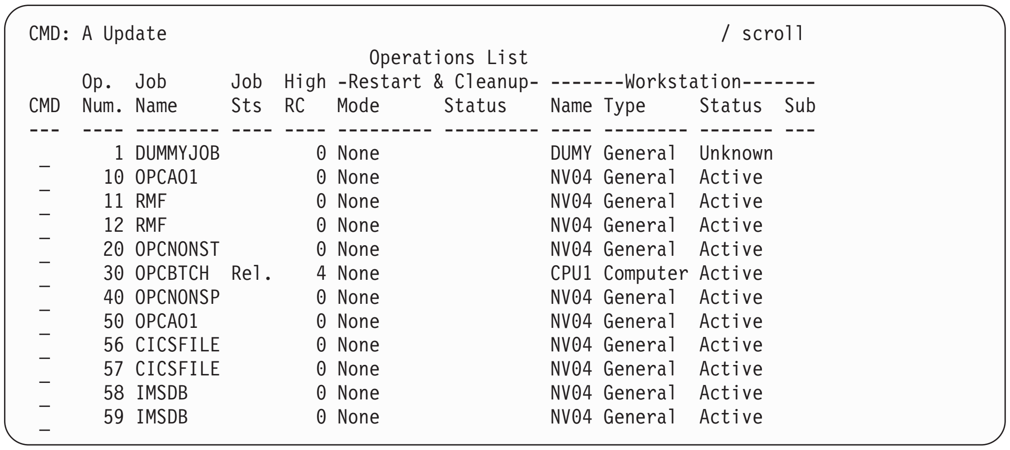

```
CMD: A Update                                                    / scroll
                                Operations List
     Op.  Job      Job  High -Restart & Cleanup- -------Workstation-------
CMD  Num. Name     Sts  RC   Mode      Status    Name Type     Status  Sub
---  ---- -------- ---- ---- --------- --------- ---- -------- ------- ---
 _      1 DUMMYJOB         0 None                DUMY General  Unknown
 _     10 OPCAO1           0 None                NV04 General  Active
 _     11 RMF              0 None                NV04 General  Active
 _     12 RMF              0 None                NV04 General  Active
 _     20 OPCNONST         0 None                NV04 General  Active
 _     30 OPCBTCH  Rel.    4 None                CPU1 Computer Active
 _     40 OPCNONSP         0 None                NV04 General  Active
 _     50 OPCAO1           0 None                NV04 General  Active
 _     56 CICSFILE         0 None                NV04 General  Active
 _     57 CICSFILE         0 None                NV04 General  Active
 _     58 IMSDB            0 None                NV04 General  Active
 _     59 IMSDB            0 None                NV04 General  Active
```

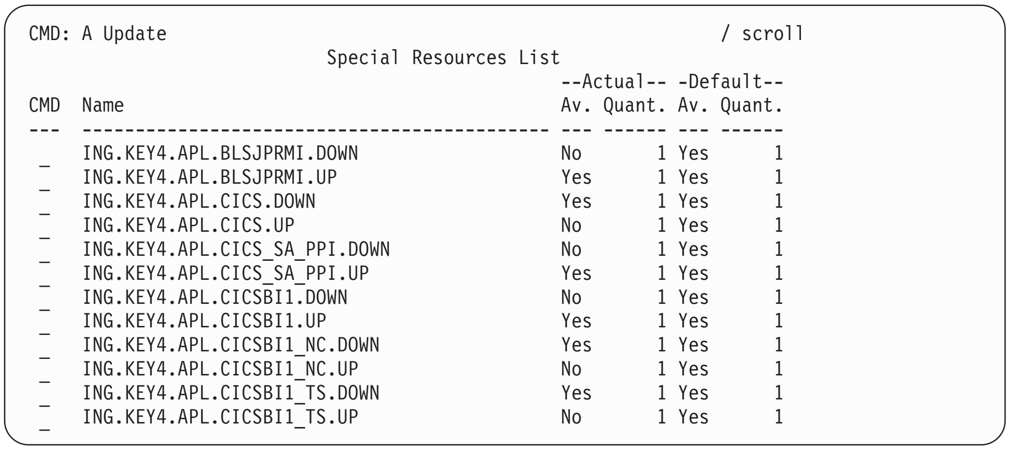
Figure 10. TWS Operations Interface Panel, Screen 4

Displaying TWS Special Resources

If you specify TYPE=SR, this will select TWS Special Resources for display. To reduce the number of special resources that are listed, you can use the optional parameter of SRNAME= to specify the special resource name.

In OUTMODE=LINE, a set of messages will be returned that contains all the data for the selected special resources.

The fullscreen interface is shown in Figure 11 on page 25.



```
CMD: A Update                                                    / scroll
                        Special Resources List
                                                  --Actual-- -Default--
CMD  Name                                         Av. Quant. Av. Quant.
---  -------------------------------------------- --- ------ --- ------
 _   ING.KEY4.APL.BLSJPRMI.DOWN                   No       1 Yes      1
 _   ING.KEY4.APL.BLSJPRMI.UP                     Yes      1 Yes      1
 _   ING.KEY4.APL.CICS.DOWN                       Yes      1 Yes      1
 _   ING.KEY4.APL.CICS.UP                         No       1 Yes      1
 _   ING.KEY4.APL.CICS_SA_PPI.DOWN                No       1 Yes      1
 _   ING.KEY4.APL.CICS_SA_PPI.UP                  Yes      1 Yes      1
 _   ING.KEY4.APL.CICSBI1.DOWN                    No       1 Yes      1
 _   ING.KEY4.APL.CICSBI1.UP                      Yes      1 Yes      1
 _   ING.KEY4.APL.CICSBI1_NC.DOWN                 Yes      1 Yes      1
 _   ING.KEY4.APL.CICSBI1_NC.UP                   No       1 Yes      1
 _   ING.KEY4.APL.CICSBI1_TS.DOWN                 Yes      1 Yes      1
 _   ING.KEY4.APL.CICSBI1_TS.UP                   No       1 Yes      1
```

Figure 11. TWS Special Resources Interface Panel

If more data is available than is displayed on the screen, use the PF8 (Down) or PF7 (Up) keys to scroll the data up or down.

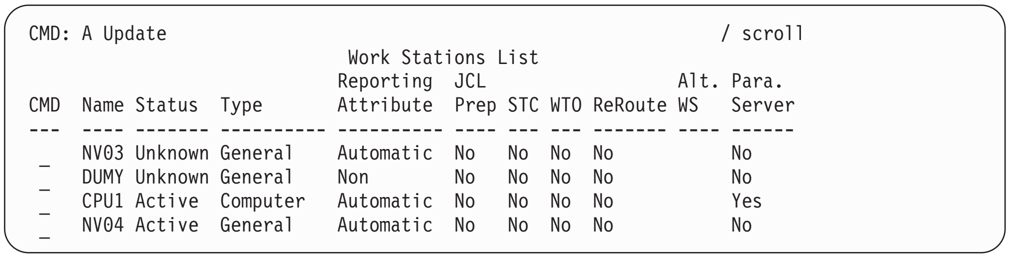
For details about the different column values, see the online help or the description of the INGOPC command in *IBM Tivoli System Automation for z/OS Operator's Commands*.

Displaying TWS Workstations

If you specify TYPE=WS, this will select TWS Workstations for display. To reduce the number of workstations that are listed, you can use the optional parameter of WSNAME= to specify the workstation name.

In OUTMODE=LINE, a set of messages will be returned that contains all the data for the selected workstations.

The fullscreen interface is shown in Figure 12.



```
CMD: A Update                                                    / scroll
                         Work Stations List
                        Reporting  JCL                     Alt. Para.
CMD  Name Status  Type       Attribute  Prep STC WTO ReRoute WS   Server
---  ---- ------- ---------- ---------- ---- --- --- ------- ---- ------
 _   NV03 Unknown General    Automatic  No   No  No  No           No
 _   DUMY Unknown General    Non        No   No  No  No           No
 _   CPU1 Active  Computer   Automatic  No   No  No  No           Yes
 _   NV04 Active  General    Automatic  No   No  No  No           No
```

Figure 12. TWS Workstations Interface Panel

You can scroll left and right with the PF11 (Next) and PF10 (Previous) keys. If more data is available than is displayed on the screen, use the PF8 (Down) or PF7 (Up) keys to scroll the data up or down.

For details about the different column values, see the online help or the description of the INGOPC command in *IBM Tivoli System Automation for z/OS Operator's Commands*.

Figure 13 on page 26 is displayed if you press the PF11 key.

```
CMD: A Update                                                  / scroll
                              Work Stations List
       --Comp. Ops.-- --Int. Ops.--- -Started- --Ready-- -Waiting-
CMD  Name Num. eDur aDur Num. eDur aDur Num. eDur Num. eDur Num. eDur
---  ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ----
 _   NV03    0    0    0    0    0    0    0    0    0    0    0    0
 _   DUMY   12   11    0    0    0    0    0    0   26   20  115  115
 _   CPU1   13    0    1    0    0    0    2    0    0    0  102    8
 _   NV04   19    1    2    0    0    0    0    0    5    0   97    6
```

Figure 13. TWS Workstations Interface Panel, Screen 2

Displaying TWS Calendars

If you specify TYPE=CAL, this will select TWS Calendars for display. To reduce the number of calendars that are listed, you can use the optional parameter of CALENDAR= to specify the calendar name.

In OUTMODE=LINE, a set of messages will be returned that contains all the data for the selected calendars.

The fullscreen interface is shown in Figure 14.

```
CMD: No Commands allowed                                       / scroll
                                 Calendar List

CMD  Name             Days Shift Description
---  ---------------- ---- ----- ------------------------------
     APC                 8 0000  general APC calendar
     DEFAULT             8 0000  general APC calendar
```

Figure 14. TWS Calendar Interface Panel

If more data is available than is displayed on the screen, use the PF8 (Down) or PF7 (Up) keys to scroll the data up or down.

For details about the different column values, see the online help or the description of the INGOPC command in *IBM Tivoli System Automation for z/OS Operator's Commands*.

Specifying Additional LIST Criteria

Additional search criteria can be specified with INGOPC REQ=LIST by using the TWSPARM= parameter. TWSPARM= is only valid with REQ=LIST and OUTMODE=LINE and can be used to specify any valid combination of TWS/OPC LIST arguments and associated values.

The *TWS for z/OS Programming Interfaces* manual describes the TWS/OPC LIST arguments in detail. Arguments must match the current plan segment that is being searched; for example, TYPE=OP searches the CPOP segment (current plan operations) so only CPOP arguments are valid with TYPE=OP.

You can specify additional search criteria in two ways:

1. Firstly, via the command parameter TWSPARM=. TWS/OPC LIST arguments are specified as keyword value pairs separated by an equals sign (=). Pairs of data are separated by the semicolon character (;). For example:
   ```
   INGOPC ... TWSPARM=(PRIORITY=3;STATUS=A)
   ```

2. Secondly, via the default safe as passed to the INGOPC command. TWS/OPC LIST arguments are specified as keyword value pairs separated by an equals sign (=). Each pair is contained as a single line of a multiline message in the default safe. For example:

   ```
   twsparms.0 = 2
   twsparms.1 = 'JOBCRT = Y'
   twsparms.2 = 'STATUS = A'
   'PIPE STEM twsparms. | COLLECT | SAFE * '
   'PIPE NETV INGOPC ......'
   ```

If TWS/OPC LIST arguments are specified via the default safe and via the TWSPARM command parameter in the same invocation of INGOPC then the values specified via TWSPARM= will take precedence.

For example:

```
INGOPC opc_controller,REQ=LIST,TYPE=OP,TWSPARM=(JOBCRT=Y),
                     OUTMODE=LINE
```

will display all operations with the critical job flag set to Y.

```
INGOPC opc_controller,REQ=LIST,TYPE=OP,OUTMODE=LINE,
                     TWSPARM=(JOBCRT=Y;OPNO=1)
```

will display all operations with the critical job flag set to Y and the operation number equal to 1.

```
INGOPC opc_controller,REQ=LIST,TYPE=APPL,TWSPARM=(JOBCRT=Y),
                     OUTMODE=LINE
```

is invalid because the JOBCRT argument is not valid with TYPE=APPL (segment type CPOC).

Modifying the Current Plan

To modify the Current Plan, use the INGOPC command with REQ=MOD and specify the type of TWS resource required from:

- APPL for Applications
- OP for Operations
- SR for Special Resources
- WS for Workstations
- CAL for Calendars

Alternatively you can use the modify line commands in the INGOPC display panels.

Line Mode Modifications

You can use the INGOPC command to modify TWS Current Plan resources in line mode. First, you must specify the TWS resource, and then specify the data that is to be modified.

You specify the TWS resource with, selection criteria parameters that are different for each TWS resource type, as shown in Table 2 on page 28:

Table 2. TWS Resource Type Selection Criteria Parameters

TWS Resource Type	Selection Criteria Parameters
APPL	**AD=** The Application Description of the applications occurrence in the current plan. **IA=** The Applications Input Arrival Time of the applications occurrence in the current plan.
OP	**AD=** The Application Description of the application that the operation belongs to in the current plan. **IA=** The Applications Input Arrival Time of the application that the operation belongs to in the current plan. **OPNO=** The Operation Number of the operation in the current plan.
SR	**SRNAME=** The Special Resource Name in the current plan of the required special resource.
WS	**WSNAME=** The Work Station Name in the current plan of the required workstation.

You can specify the data to be modified for a given TWS resource in two ways:

- Firstly, via the command parameter UPDATE=. The data are specified as keyword value pairs separated by an equals sign (=). Pairs of data are separated by the semi-colon character (;). For example:
  ```
  INGOPC ... UPDATE=(PRIORITY=3;STATUS=A)
  ```
- Secondly, via the default safe as passed to the INGOPC command. The data are specified as keyword value pairs separated by " = " (the blanks either side of the equals sign are required). Each pair is contained as a separate message in the default safe. For example:
  ```
  updateStem.0 = 2
  updateStem.1 = 'PRIORITY = 3'
  updateStem.2 = 'STATUS = A'
  'PIPE STEM updateStem. | COLLECT | SAFE * '
  'PIPE NETV INGOPC ......'
  ```

The valid keywords are derived from the TWS-related manual, *Tivoli Workload Scheduler for z/OS Programming Interfaces* (SH19-4545-00). Any keyword as specified in the "Modify Request", "Arguments" section may be used. Table 3 matches the INGOPC TYPE= parameter value to the TWS resource types.

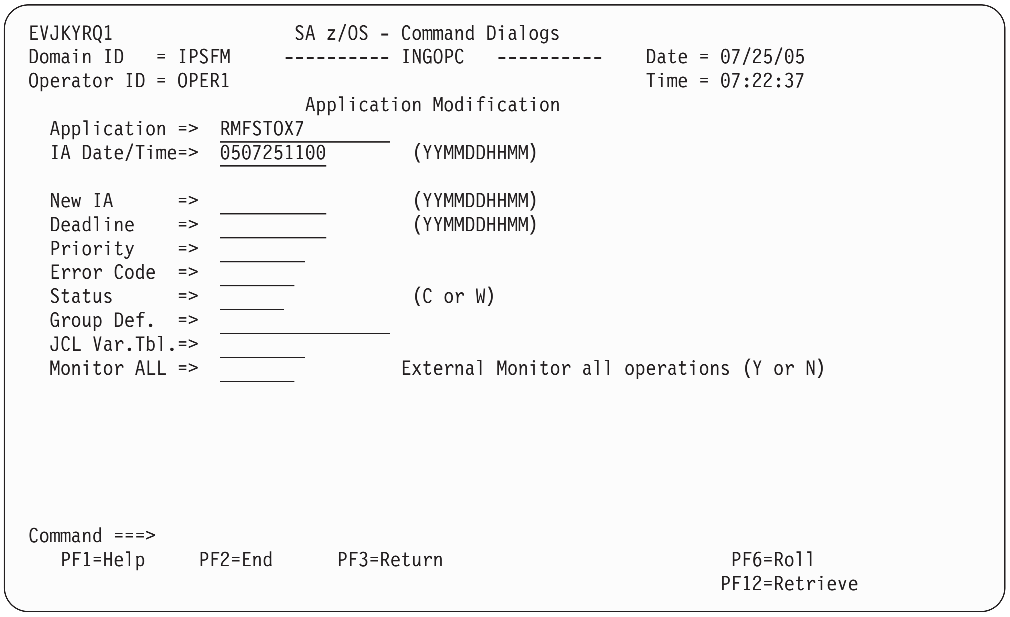
Table 3. INGOPC TYPE= Parameters Matched to TWS Resource Types.

TYPE=	TWS Current Plan Resource	TWS Manual Section
APPL	CPOC	Modify CPOC Arguments
OP	CPOP	Modify CPOP Arguments
SR	CSR	Modify CSR Arguments
WS	CPWS	Modify CPWS Arguments

Modifications via Panel Interaction

TWS Applications

From a list of applications (TYPE=APPL) use the "A" (update) command code against an application. The panel shown in Figure 15 is displayed.



```
EVJKYRQ1                 SA z/OS - Command Dialogs
Domain ID   = IPSFM     ---------- INGOPC   ----------    Date = 07/25/05
Operator ID = OPER1                                       Time = 07:22:37
                          Application Modification
  Application =>  RMFSTOX7
  IA Date/Time=>  0507251100        (YYMMDDHHMM)

  New IA      =>  __________        (YYMMDDHHMM)
  Deadline    =>  __________        (YYMMDDHHMM)
  Priority    =>  ________
  Error Code  =>  _______
  Status      =>  ______            (C or W)
  Group Def.  =>  ________________
  JCL Var.Tbl.=>  ________
  Monitor ALL =>  _______          External Monitor all operations (Y or N)







Command ===>
   PF1=Help     PF2=End      PF3=Return                           PF6=Roll
                                                                 PF12=Retrieve
```

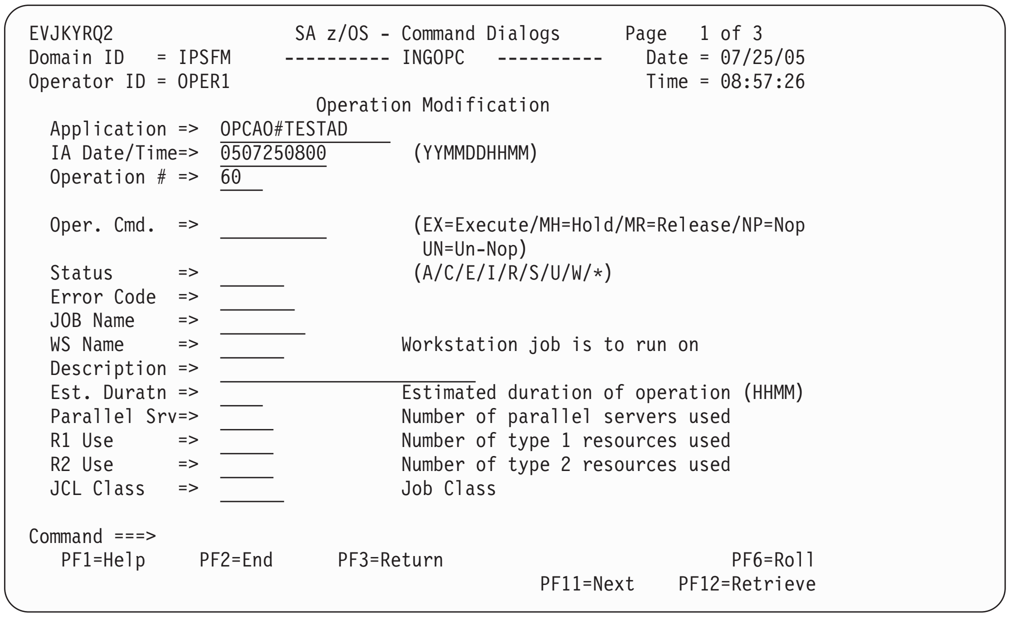
Figure 15. TWS Applications Modification Panel

Fill in the fields to achieve the desired result and press the Enter key.

For details about the different fields see the online help.

TWS Operations

From a list of operations (TYPE=OP) use the "A" (update) command code against an operation. The panel shown in Figure 16 on page 30 is displayed.

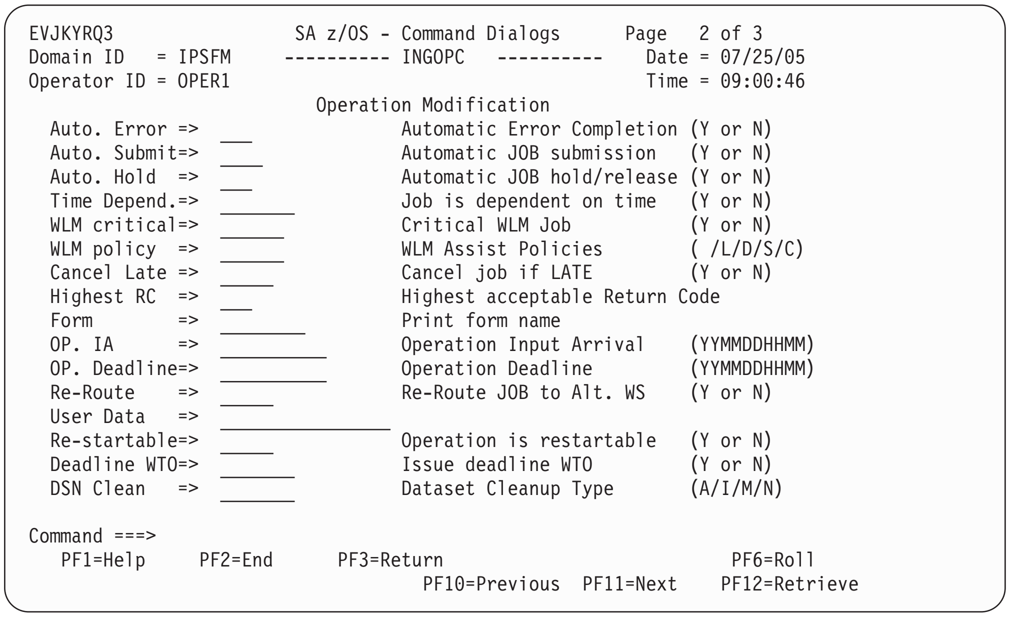


```
EVJKYRQ2                 SA z/OS - Command Dialogs      Page   1 of 3
Domain ID   = IPSFM     ---------- INGOPC   ----------    Date = 07/25/05
Operator ID = OPER1                                       Time = 08:57:26
                           Operation Modification
  Application =>  OPCAO#TESTAD____
  IA Date/Time=>  0507250800        (YYMMDDHHMM)
  Operation # =>  60__

  Oper. Cmd.  =>  __________        (EX=Execute/MH=Hold/MR=Release/NP=Nop
                                     UN=Un-Nop)
  Status      =>  ______            (A/C/E/I/R/S/U/W/*)
  Error Code  =>  _______
  JOB Name    =>  ________
  WS Name     =>  ______           Workstation job is to run on
  Description =>  ________________________
  Est. Duratn =>  ____             Estimated duration of operation (HHMM)
  Parallel Srv=>  _____            Number of parallel servers used
  R1 Use      =>  _____            Number of type 1 resources used
  R2 Use      =>  _____            Number of type 2 resources used
  JCL Class   =>  ______           Job Class

Command ===>
   PF1=Help     PF2=End      PF3=Return                           PF6=Roll
                                              PF11=Next    PF12=Retrieve
```

Figure 16. TWS Operations Modification Panel

For details about the different fields see the online help.

Fill in the fields to achieve the desired result, then press the Enter key or press PF11 key to scroll to the next page. If you press the PF11 key, the panel shown in Figure 17 is displayed.

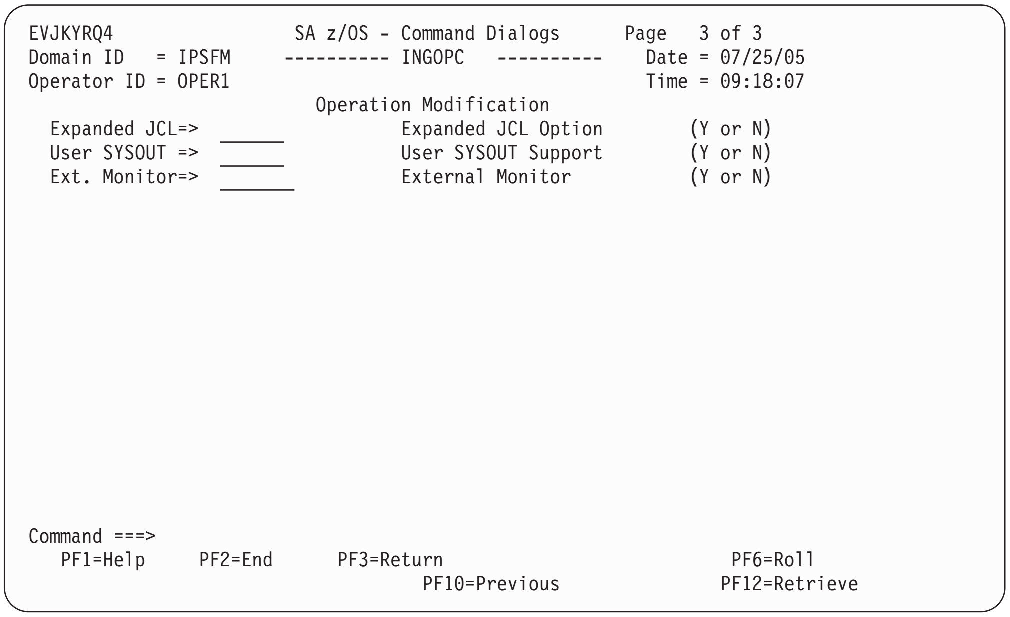


```
EVJKYRQ3                 SA z/OS - Command Dialogs      Page   2 of 3
Domain ID   = IPSFM     ---------- INGOPC   ----------    Date = 07/25/05
Operator ID = OPER1                                       Time = 09:00:46
                           Operation Modification
  Auto. Error =>  ___              Automatic Error Completion (Y or N)
  Auto. Submit=>  ____             Automatic JOB submission   (Y or N)
  Auto. Hold  =>  ___              Automatic JOB hold/release (Y or N)
  Time Depend.=>  _______          Job is dependent on time   (Y or N)
  WLM critical=>  ______           Critical WLM Job           (Y or N)
  WLM policy  =>  ______           WLM Assist Policies        ( /L/D/S/C)
  Cancel Late =>  _____            Cancel job if LATE         (Y or N)
  Highest RC  =>  ___              Highest acceptable Return Code
  Form        =>  ________         Print form name
  OP. IA      =>  __________       Operation Input Arrival    (YYMMDDHHMM)
  OP. Deadline=>  __________       Operation Deadline         (YYMMDDHHMM)
  Re-Route    =>  _____            Re-Route JOB to Alt. WS    (Y or N)
  User Data   =>  ________________
  Re-startable=>  _____            Operation is restartable   (Y or N)
  Deadline WTO=>  _______          Issue deadline WTO         (Y or N)
  DSN Clean   =>  _______          Dataset Cleanup Type       (A/I/M/N)

Command ===>
   PF1=Help     PF2=End      PF3=Return                           PF6=Roll
                                       PF10=Previous  PF11=Next    PF12=Retrieve
```

Figure 17. TWS Operations Modification Panel, Screen 2

For details about the different fields see the online help.

Fill in the fields to achieve the desired result, then press the Enter key or press PF11 key to scroll to the next page. If you press the PF11 key, the panel shown in Figure 18 on page 31 is displayed.



```
EVJKYRQ4                 SA z/OS - Command Dialogs      Page   3 of 3
Domain ID   = IPSFM     ---------- INGOPC   ----------    Date = 07/25/05
Operator ID = OPER1                                       Time = 09:18:07
                           Operation Modification
  Expanded JCL=>  ______          Expanded JCL Option        (Y or N)
  User SYSOUT =>  ______          User SYSOUT Support        (Y or N)
  Ext. Monitor=>  _______         External Monitor           (Y or N)















Command ===>
   PF1=Help     PF2=End      PF3=Return                           PF6=Roll
                                     PF10=Previous              PF12=Retrieve
```

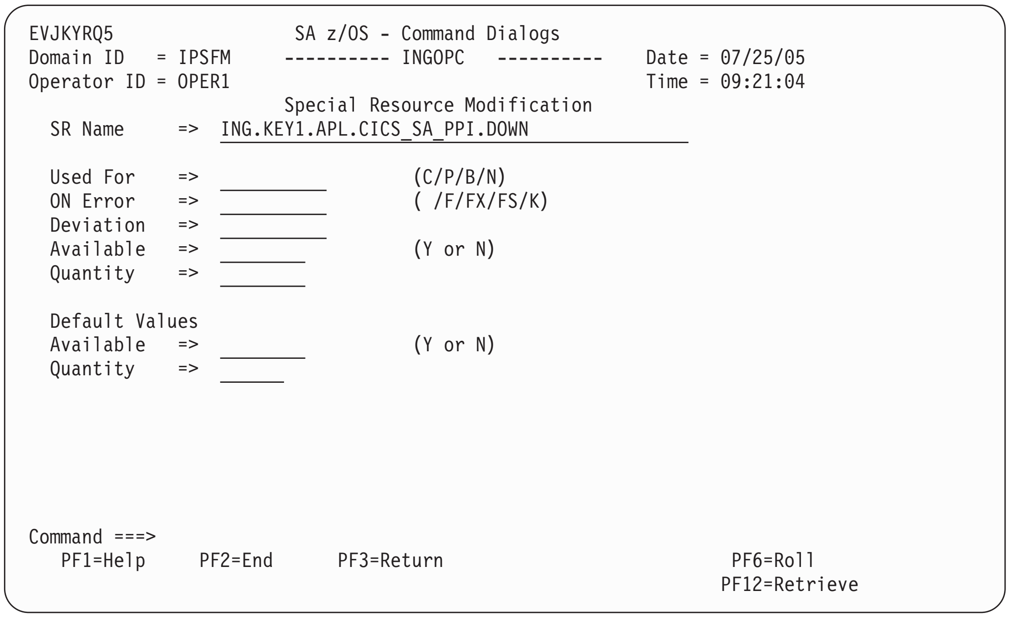
Figure 18. TWS Operations Modification Panel, Screen 3

Fill in the fields to achieve the desired result, then press the Enter key.

For details about the different fields see the online help.

TWS Special Resources

From a list of special resources (TYPE=SR) use the "A" (update) line command against a special resource. The panel shown in Figure 19 is displayed.



```
EVJKYRQ5                 SA z/OS - Command Dialogs
Domain ID   = IPSFM     ---------- INGOPC   ----------    Date = 07/25/05
Operator ID = OPER1                                       Time = 09:21:04
                        Special Resource Modification
  SR Name     =>  ING.KEY1.APL.CICS_SA_PPI.DOWN

  Used For    =>  __________       (C/P/B/N)
  ON Error    =>  __________       ( /F/FX/FS/K)
  Deviation   =>  __________
  Available   =>  ________         (Y or N)
  Quantity    =>  ________

  Default Values
  Available   =>  ________         (Y or N)
  Quantity    =>  ______







Command ===>
   PF1=Help     PF2=End      PF3=Return                           PF6=Roll
                                                                PF12=Retrieve
```

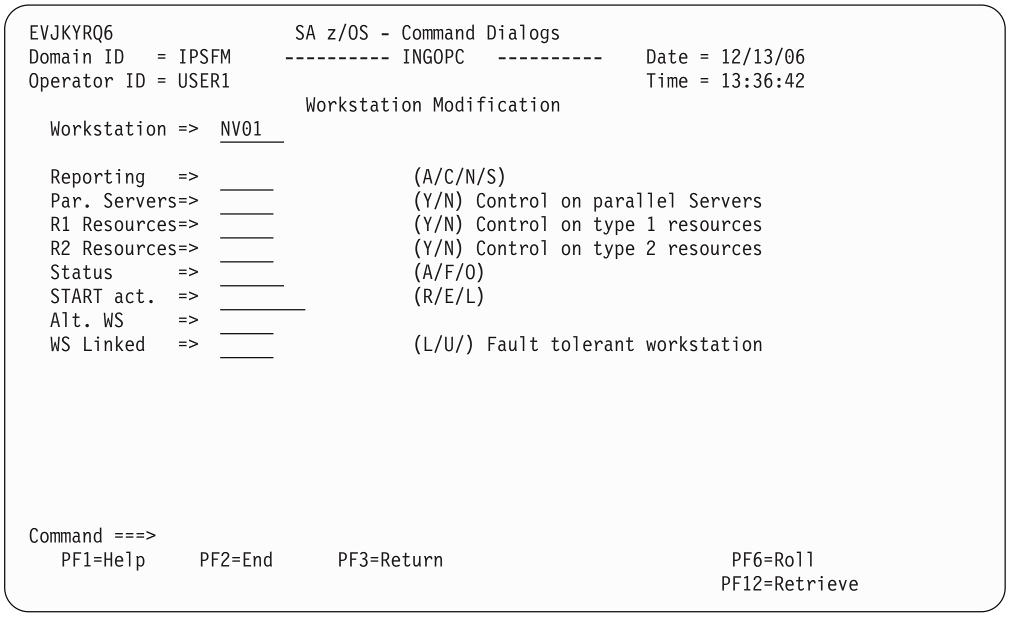
Figure 19. TWS Special Resources Modification Panel

Fill in the fields to achieve the desired result and press the Enter key.

For details about the different fields see the online help.

TWS Workstations

From a list of workstations (TYPE=WS) use the A (update) line command against a workstation. The panel shown in Figure 20 is displayed.



```
EVJKYRQ6                  SA z/OS - Command Dialogs
Domain ID   = IPSFM     ---------- INGOPC   ----------    Date = 12/13/06
Operator ID = USER1                                       Time = 13:36:42
                           Workstation Modification
  Workstation =>  NV01

  Reporting   =>  _____             (A/C/N/S)
  Par. Servers=>  _____             (Y/N) Control on parallel Servers
  R1 Resources=>  _____             (Y/N) Control on type 1 resources
  R2 Resources=>  _____             (Y/N) Control on type 2 resources
  Status      =>  ______            (A/F/O)
  START act.  =>  ________          (R/E/L)
  Alt. WS     =>  _____
  WS Linked   =>  _____             (L/U/) Fault tolerant workstation




Command ===>
   PF1=Help     PF2=End      PF3=Return                          PF6=Roll
                                                                PF12=Retrieve
```

Figure 20. TWS Workstations Modification Panel

Fill in the fields to achieve the desired result and press the Enter key.

For details about the different fields see the online help.

Chapter 3. Monitoring using SDF

The Status Display Facility uses color to represent the various subsystem resource statuses such as error, warning, action, or informational states. Typically, a subsystem shown in green on a Status Display Facility status panel indicates that it is up, whereas red indicates a stopped or problem state.

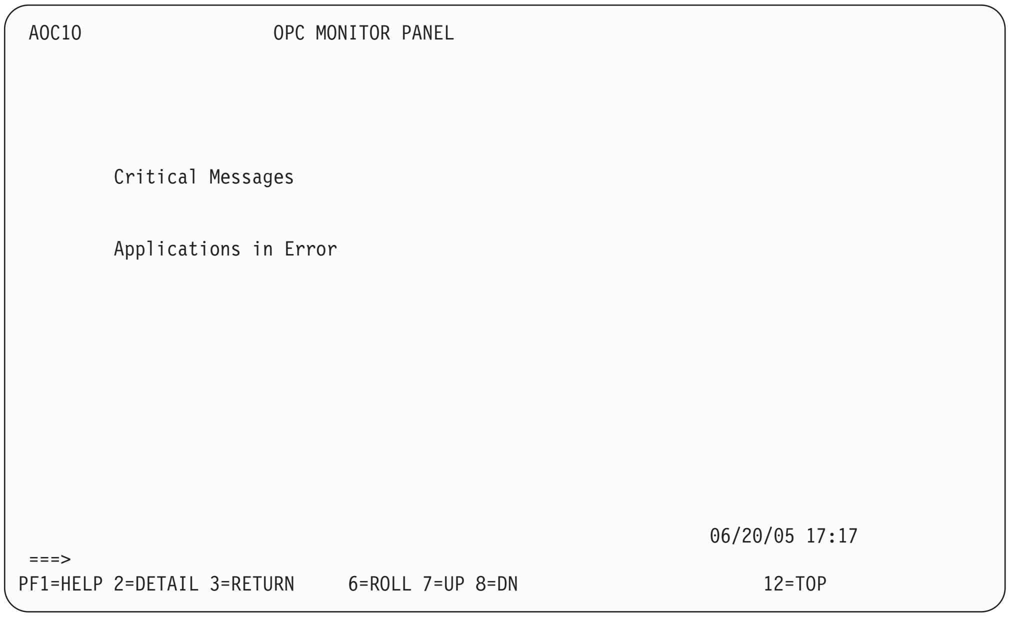
The Status Display Facility status display panels can be tailored to present the status of system components in a hierarchical manner. The hierarchical display of status information is implemented using tree structures. A tree structure always starts with the system name as the root component. The "leaves" of the tree are the monitored resources.

Color can be propagated up or down the leaves of the tree structure based on the order of dependencies. The effect of propagation is to consolidate, at the root component, the status of all the monitored resources in that system. In this way, the color of the root component reflects the most important or critical status in a computer operations center. If all the monitored resources are green, the root component (the system) will be green.

TWS Automation provides additional Status Display Facility panels that monitor the events that occur in the following areas for all TWS regions defined to TWS Automation:

Applications in Error
: Shows the TWS applications that have encountered an error.

To use the TWS Automation Status Display Facility panels, enter SDF at a NetView panel command line.



```
 AOC10                OPC MONITOR PANEL



        Critical Messages


        Applications in Error








                                                        06/20/05 17:17
 ===>
PF1=HELP 2=DETAIL 3=RETURN     6=ROLL 7=UP 8=DN                       12=TOP
```

Figure 21. The TWS Monitor Panel

© Copyright IBM Corp. 1990, 2010

Chapter 4. NMC Display Support

SA z/OS TWS Automation will display the status of TWS applications or operations in error. In addition, the status of TSO users may optionally be displayed.

NMC Resource Definitions

TWS Naming Convention

For TWS status monitoring the anchor "TWS" enables the status of TWS operations to be accessed without the need to know the name of the currently active TWS Controller. TWS resources are represented by objects with a minor name of:

```
jobname
```

TSO Naming Convention

For TSO user monitoring the anchor "TSO" likewise enables the status of all TSO users to be accessed from one view. TSO users are represented by objects with a minor name of:

```
systemId_tsoUserName
```

where:

systemId	is the name of the system that the TSO user is logged onto.
tsoUserName	is the name of the TSO user.

NMC BuildViews for TWS objects

To show the TWS objects on a view off the main tree, use the buildviews statements shown in Figure 22:

```
VIEW=K1.OPC,
  ANNOTATION='OPC Monitoring in Sysplex K1'

WILDCARD=(?,*)

NONSNA=K1.OPC/ANCH*,
  QUERYFIELD=MYNAME
```

Figure 22. Sample TWS BuildViews Statements

To show the TSO objects on a view off the main tree, use the buildviews statements shown in Figure 23:

```
VIEW=K1.TSO,
  ANNOTATION='TSO Monitoring in Sysplex K1'

WILDCARD=(?,*)

NONSNA=K1.TSO/ANCH*,
  QUERYFIELD=MYNAME
```

Figure 23. Sample TSO BuildViews Statements

© Copyright IBM Corp. 1990, 2010

Chapter 5. Managing the PPI Receivers

This chapter describes how to start up and shut down the NetView PPI receivers. There are two types of receiver subsystems:

- The *Request* receiver is responsible for handling status updates, requests, and command requests from the TWS exit.
- The *Command* receiver is responsible for handling commands that are issued from a batch program.

If requests for automation services from TWS need to be stopped for some reason, then the TWS Request Receiver in the appropriate SA z/OS NetView Agent should be stopped. Requests typically start or stop a SA z/OS resource and are TWS operations that are assigned to a workstation. For *conventional* requests, the general workstation must have the name `NV**`, but for *command* requests the automation workstation can have any name and use the workstation destination name to define the NetView domain.

If batch interface commands are to be stopped for some reason, then the TWS Command Receiver in the appropriate SA z/OS Agent or NetView Agent should be stopped. Batch commands are typically used to gather information for further processing.

Use the following procedures to start or stop the desired PPI receivers. Because there may be more than one Command or Request receiver involved, you should check with your System Programmer to determine the correct subsystems to start or stop.

Subtopics:
- "Starting and Stopping the Request Receiver"
- "Starting and Stopping the Command Receivers" on page 38

Starting and Stopping the Request Receiver

The Request receiver is controlled by SA z/OS. It is defined to SA z/OS as a NON-MVS subsystem. If the system programmer has taken the defaults when customizing TWS Automation, then the name of the Request receiver will be TWSREQR. If this is not the case, then you must find out the SA z/OS subsystem name of the Request receiver from the system programmers.

To start the Request receiver, Issue the INGREQ REQ=START command against the appropriate subsystem, for example:

```
INGREQ TWSREQR/APL/SYS1 REQ=START
```

You should not have to start the Request receiver in normal circumstances because it should automatically start when the SA z/OS Agent registers with the Automation Manager.

To stop the Request receiver, Issue the INGREQ REQ=STOP command against the appropriate subsystem, for example:

```
INGREQ TWSREQR/APL/SYS1 REQ=STOP
```

© Copyright IBM Corp. 1990, 2010

Starting and Stopping the Command Receivers

The Command receivers are controlled by SA z/OS. They are defined to SA z/OS as NON-MVS subsystems. If the system programmer has taken the defaults when customizing TWS Automation, then there will be only one Command receiver and its name will be TWSCMDR. However, there can be multiple Command receivers and you need to find the names of them from the system programmers.

To start the Command receiver, Issue the INGREQ REQ=START command against the appropriate subsystem, for example:

```
INGREQ TWSCMDR/APL/SYS1 REQ=START
```

You should not have to start the Command receiver in normal circumstances because it should automatically start when the SA z/OS Agent registers with the Automation Manager.

To stop the Command receiver, Issue the INGREQ REQ=STOP command against the appropriate subsystem, for example:

```
INGREQ TWSCMDR/APL/SYS1 REQ=STOP
```

Chapter 6. TWS Automation Operator Commands

The following commands are used with TWS Automation:

Table 4. TWS Automation Commands

Command	**Description**
INGOPC	Queries or modifies a TWS controller. See "INGOPC" on page 40.
OPC/OPCA	Displays the TWS Automation Main Menu (EVJKYOPC) that lists the most commonly used commands. See "TWS Automation Main Menu" on page 49.
OPCAPOST	Posts an operation to TWS from SA z/OS. See "OPCAPOST—Posting a TWS Operation from SA z/OS" on page 50.
OPCAQRY	Displays the status of TWS Automation operations. See "OPCAQRY—Display Status of Operations" on page 51.
SRSTAT	Sets the status of TWS special resources. See "SRSTAT—Setting TWS Special Resource Status" on page 55.

© Copyright IBM Corp. 1990, 2010

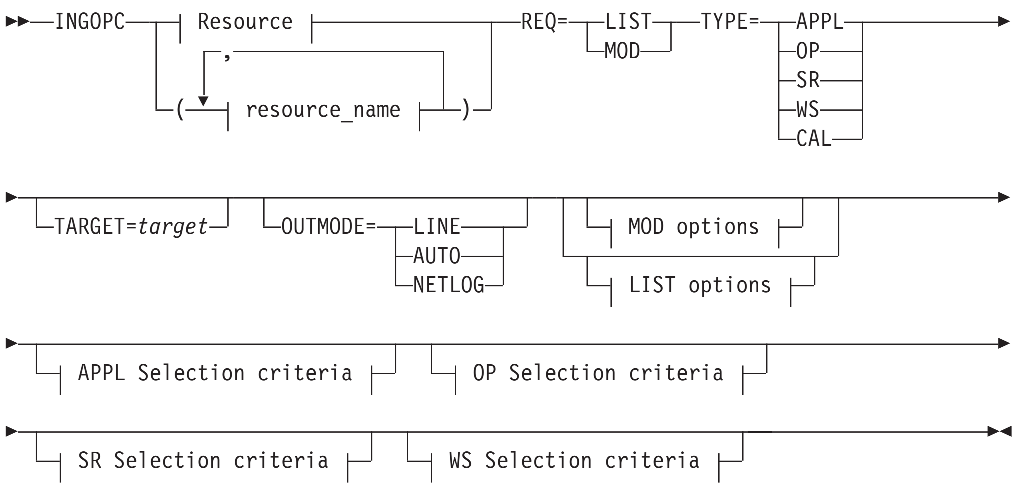
INGOPC

Purpose

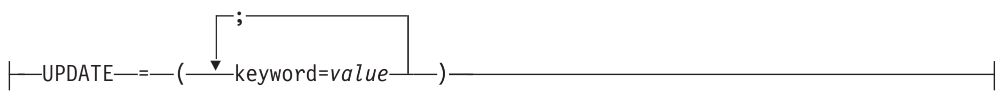
The INGOPC command lets you:

- Display Application, Operation, Special Resource, Work Station and Calendar information from the Current Plan.
- Modify Application, Operation, Special Resource, Work Station information in the Current Plan.
- Issue a request against any controller defined to SA z/OS in a sysplex.
- Issue a request against a foreign controller where the local tracker is defined to SA z/OS.
- Output the INGOPC command in either fullscreen or pipeable line mode.

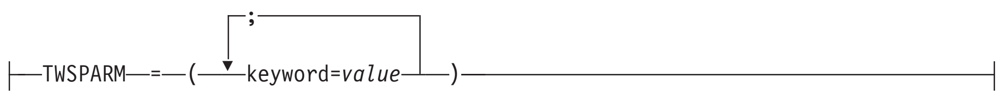
Format



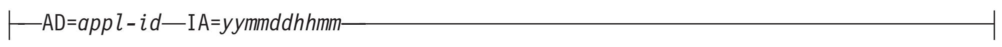
MOD options:



LIST options:



APPL Selection criteria:



OP Selection criteria:

```
├──AD=appl-id──IA=yymmddhhmm──OPNO=nnnn──JOBNAME=jobname──STATUS=status──────────►

►─ERRCODE=error_code──GROUP=group──OWNER=owner──PRIORITY=priority────────────────►

►─WSNAME=workstation─────────────────────────────────────────────────────────────┤
```

SR Selection options:

```
├──SRNAME──=──special_resource──────────────────────────────────────────────────┤
```

WS Selection options:

```
├──WSNAME=workstation───────────────────────────────────────────────────────────┤
```

Parameters

resource
The resource specifies the OPC controller that is to be queried or modified. Multiple specifications are allowed as well as system and sysplex application groups. Wildcards % and * are supported.

The command attempts to resolve the specification to a single appropriate target resource. In all cases the groups are resolved to their members and wildcards are resolved to specific sets of resources.

The resulting list of resources is scanned to check whether there is a single active controller. If a single active controller is found then it is used. If no active controller is found then the list is scanned to check whether there is a single active tracker (only active trackers with LUNAME policy entries are checked because only these trackers can be used to communicate with TWS via the PIF interface). If a single active tracker is found then it is used.

If no viable resources are found then an error message is displayed. If multiple viable resources are found, and INGOPC is running in full screen mode, then a selection panel will be displayed; when OUTMODE=LINE is specified an error message is displayed.

If an active controller could be found a command is dispatched to the appropriate system in the sysplex to execute the OPC API on the same system as the active controller. If a tracker was found, the LUNAME parameter of the trackers OPCCNTL entry may be used to specify a remote controller. In this case the command is dispatched to the system where the tracker is running and the APPC API is used to connect to the remote controller from that system.

REQ
Specifies the request to be issued to the OPC subsystem. It can be one of the following:

LIST	Lists OPC Current Plan resources.
MOD	Modifies OPC Current Plan resources.

TYPE
Specifies the type of Current Plan resource to be listed or modified. It can be one of the following:

APPL	Specifies the Current Plan Application Description resource.

OP Specifies the Current Plan Operation resource.

SR Specifies the Current Plan Special resource.

WS Specifies the Current Plan Workstation resource.

CAL Specifies the Current Plan Calendar resource.

UPDATE
Specifies the fields that are to be updated and the new contents of the field. Multiple fields can be specified separated by a semi-colon ";". The names of the fields are the same names as specified in the MODIFY command arguments in *TWS for z/OS Programming Interfaces*.

An alternative to specifying all the fields to be updated using the UPDATE= parameter is to specify the fields and their contents in the default SAFE. Specify one field per message with the format of <fieldname><blank>=<blank><contents>The blanks between the fieldname and the = symbol and the = symbol and the contents are required.

TWSPARM
Specifies additional fields to be used to locate a resource during LIST processing. Multiple fields may be specified separated by a semi-colon ";". The names of the fields are the same names as specified in the LIST command arguments described in the *TWS for z/OS Programming Interfaces* manual. The field names must match the TWS segment being searched. For example, CPOC fields are valid for TYPE=APPL and CPOP fields are valid for TYPE=OP, etc. A TWS EQQ* message will be issued if the field specification is incorrect.

An alternative to specifying additional LIST fields using the TWSPARM= parameter is to specify the fields and their values in the default SAFE. Specify one field per message using the format <fieldname><blank(s)>=<blank(s)><contents>. The blanks are optional.

If the same field names are specified in both the TWSPARM and the default SAFE, the TWSPARM values will be used.

Additional LIST fields can only be specified with OUTMODE=LINE.

AD
Specifies the Application Description selection criteria. For LIST requests, this may contain the trailing "*" wildcard character. For MOD requests, this must be the exact name of the application description to be updated.

IA Specifies the input arrival date/time of the application. The format is as specified by the system programmer when installing and customizing OPC. The default format is YYYYMMDDHHMM.

OPNO
Specifies the operation number selection criteria. This is the operation number of an operation in an application description.

JOBNAME
Specifies the OPC jobname. This field is used to qualify requests of type OP and is optional for all requests.

STATUS
Specifies the OPC status. This field is used to qualify requests of type OP and is optional for all requests.

ERRCODE
Specifies the OPC error code. This field is used to qualify requests of type OP and is optional for all requests.

GROUP
Specifies the OPC group. This field is used to qualify requests of type OP and is optional for all requests.

OWNER
Specifies the OPC owner. This field is used to qualify requests of type OP and is optional for all requests.

PRIORITY
Specifies the OPC priority. This field is used to qualify requests of type OP and is optional for all requests.

SRNAME
Specifies the Special Resource selection criteria.

| For LIST requests, this may contain the trailing "*" wildcard character. For
| MOD requests, this must be the exact name of the special resource.

If the special resource name contains special characters then it must be enclosed in single quotation marks.

WSNAME
Specifies the workstation name selection criteria. Specifies the workstation name selection criteria but may also be used to qualify TYPE=OP requests.

For LIST requests, this may contain the trailing "*" wildcard character. For MOD requests, this must be the exact name of the workstation.

TARGET
Specifies the name of the system (system name or domain ID) that the command should be routed to. The TWS controller specified in the resource field must be active on this system or the command will return no data. This is only necessary when the resource is not part of the local sysplex.

For information on the TARGET parameter, refer to *IBM Tivoli System Automation for z/OS Operator's Commands*.

OUTMODE
For information on the OUTMODE parameter, refer to *IBM Tivoli System Automation for z/OS Operator's Commands*.

Restrictions

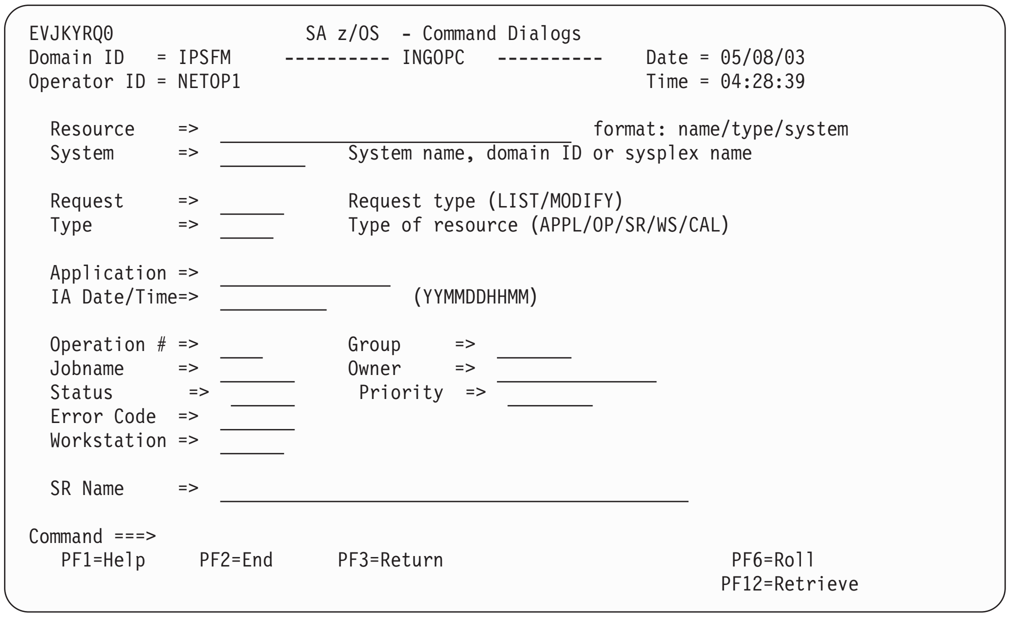
To use the INGOPC command system operations must be initialized.

Usage

The INGOPC command operates sysplex-wide. For an overview refer to "Overview of Commands that Operate Sysplex-Wide" in *IBM Tivoli System Automation for z/OS Operator's Commands*.

Examples

If you type INGOPC a panel similar to Figure 24 on page 44 is displayed.



```
EVJKYRQ0                 SA z/OS  - Command Dialogs
Domain ID   = IPSFM     ---------- INGOPC   ----------    Date = 05/08/03
Operator ID = NETOP1                                      Time = 04:28:39

  Resource    =>  _________________________________  format: name/type/system
  System      =>  ________    System name, domain ID or sysplex name

  Request     =>  ______      Request type (LIST/MODIFY)
  Type        =>  _____       Type of resource (APPL/OP/SR/WS/CAL)

  Application =>  ________________
  IA Date/Time=>  __________        (YYMMDDHHMM)

  Operation # =>  ____        Group     =>  _______
  Jobname     =>  _______     Owner     =>  _______________
  Status       =>  ______      Priority  =>  ________
  Error Code  =>  _______
  Workstation =>  ______

  SR Name     =>  ____________________________________________

Command ===>
   PF1=Help     PF2=End      PF3=Return                          PF6=Roll
                                                                PF12=Retrieve
```

Figure 24. INGOPC Command Dialog Panel

- The **Resource** field shows the name of the OPC active controller subsystem to be used for issuing the requests. The format is name/type[/system]. Wildcards are supported.
- The **System** field shows the name of the system (system name, domain ID, or sysplex name) that the command should be routed to. Specifying this is only necessary if the resources do not reside on the local sysplex.
- The **Request** field shows the request to be carried out. It can be LIST or MODIFY.
- The **Type** field shows the type of OPC Current Plan resource to be specified.
- The **Application** field specifies the OPC application id, This field is used to qualify requests of type APPL or OP and is optional for LIST requests but is required for MODIFY requests.
- The **IA Date/Time** field specifies the OPC input arrival time. This field is used to qualify requests of type APPL or OP and is optional for LIST requests but is required for MODIFY requests.
- The **Operation #** field specifies the OPC operation number. This field is used to qualify requests of type APPL or OP and is optional for LIST requests but is required for MODIFY requests.
- The **Jobname** field specifies the OPC jobname associated with an operation. This field is used to qualify requests of type OP and is optional for all requests.
- The **Status** field specifies the OPC status associated with an operation. This field is used to qualify requests of type OP and is optional for all requests.
- The **Error Code** field specifies the OPC error code associated with an operation. This field is used to qualify requests of type OP and is optional for all requests.
- The **Group** field specifies the OPC group associated with an operation. This field is used to qualify requests of type OP and is optional for all requests.
- The **Owner** field specifies the OPC owner associated with an operation. This field is used to qualify requests of type OP and is optional for all requests.
- The **Priority** field specifies the OPC priority associated with an operation. This field is used to qualify requests of type OP and is optional for all requests.

- The **Workstation** field specifies the OPC workstation for the operation. This field is used to qualify requests of type OP and type WS and is optional for all LIST requests but is required for type WS MODIFY requests.
- The **SR Name** field specifies the OPC special resource name. This field is used to qualify requests of type SR and is optional for LIST requests but is required for MODIFY requests.

If you specify `INGOPC * REQ=LIST TYPE=APPL` a panel similar to Figure 25 is displayed.

```
INGKYSTO                 SA z/OS  - Command Dialogs  Line  1    of 65
Domain ID   = IPSFM      -------- INGOPC    ---------    Date = 04/10/02
Operator ID = AFRANCK         Sysplex = KEY1PLEX         Time = 16:19:30
CMD: A Update    B Operations                                    / scroll
                             Application Occurrence List
                      Input Arrival            Error
CMD  Application Id   Date     Time  Status    Code  Description
---  ---------------- -------- ----- --------- ----- ------------------------
 _   JKOPCTST1        02/01/23 00:01 Completed       Test Batch Iface
 _   JKTEST1          02/01/23 00:08 Completed       This is a test
 _   JKOPCTST1        02/01/24 00:01 Completed       Test Batch Iface
 _   JKTEST1          02/01/24 00:08 Completed       This is a test
 _   IEFBR14          02/01/24 08:01 Completed       This is a test
 _   JKOPCTST1        02/01/25 00:01 Starting        Test Batch Iface
 _   JKTEST1          02/01/25 00:08 Completed       This is a test
 _   IEFBR14          02/01/25 08:01 Completed       This is a test
 _   JKOPCTST1        02/01/26 00:01 Error           Test Batch Iface
 _   JKTEST1          02/01/26 00:08 Completed       This is a test
 _   IEFBR14          02/01/26 08:01 Completed       This is a test
 _   JKOPCTST1        02/01/27 00:01 Error           Test Batch Iface
 _   JKTEST1          02/01/27 00:08 Completed       This is a test

Command ===>
 PF1=Help    PF2=End      PF3=Return                PF5=Filters   PF6=Roll
```

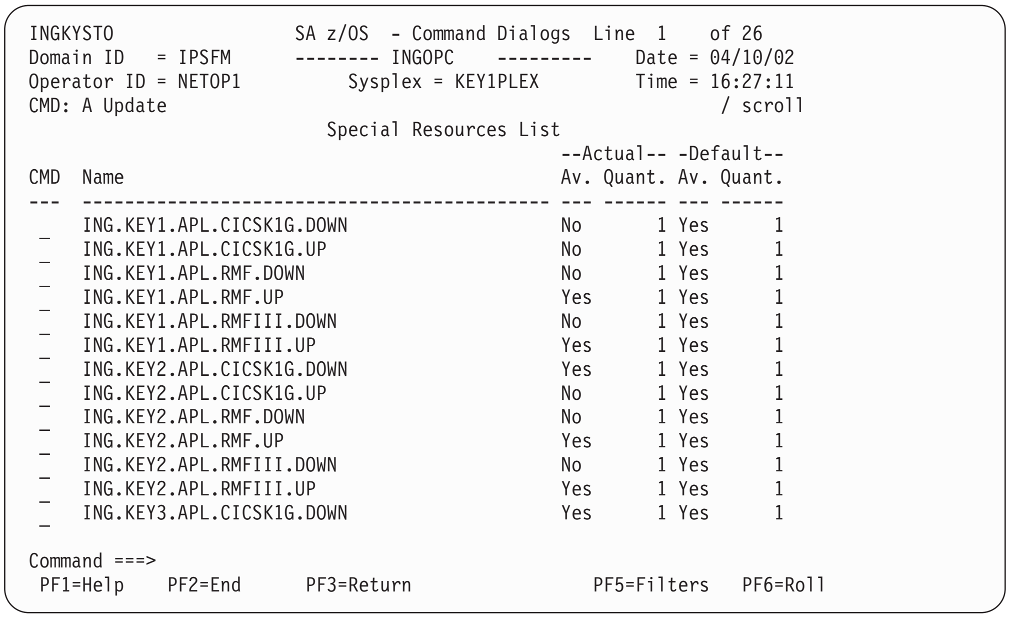
Figure 25. INGOPC REQ=LIST TYPE=APPL Sample Panel

If you specify `INGOPC * REQ=LIST TYPE=OP` a panel similar to Figure 26 is displayed.

```
INGKYSTO                 SA z/OS  - Command Dialogs  Line  1    of 3
Domain ID   = IPSFM      -------- INGOPC    ---------    Date = 04/10/02
Operator ID = NETOP1          Sysplex = KEY1PLEX         Time = 16:23:08
CMD: A Update                                                    / scroll
                                   Operations List
     Op.  -------JES-------                                         Err. Work
CMD  Num. Name     Number   Status    Reason                        Code Stn.
---  ---- -------- -------- --------- ----------------------------- ---- ----
 _      1 EVJSJ001          Completed                                    NV01
 _      1 JKTST1   JOB06429 Completed                                    N001
 _      1 IEFBR14  JOB00325 Completed                                    CPU1
 _      1 EVJSJ001 JOB07522 Completed                                    N001
 _      1 JKTST1   JOB07521 Completed                                    N001
 _      1 IEFBR14  JOB07523 Completed                                    CPU1
 _      1 EVJSJ001 JOB00787 Interrupt                                    N001
 _      1 JKTST1   JOB07524 Completed                                    N001
 _      1 IEFBR14  JOB07526 Completed                                    CPU1
 _      1 EVJSJ001 JOB07528 Error                                   JCL  N001
 _      1 JKTST1   JOB07527 Completed                                    N001
 _      1 IEFBR14  JOB07529 Completed                                    CPU1
 _      1 EVJSJ001 JOB07533 Error                                   JCL  N001

Command ===>
 PF1=Help    PF2=End      PF3=Return                PF5=Filters   PF6=Roll
```

Figure 26. INGOPC REQ=LIST TYPE=OP Sample Panel

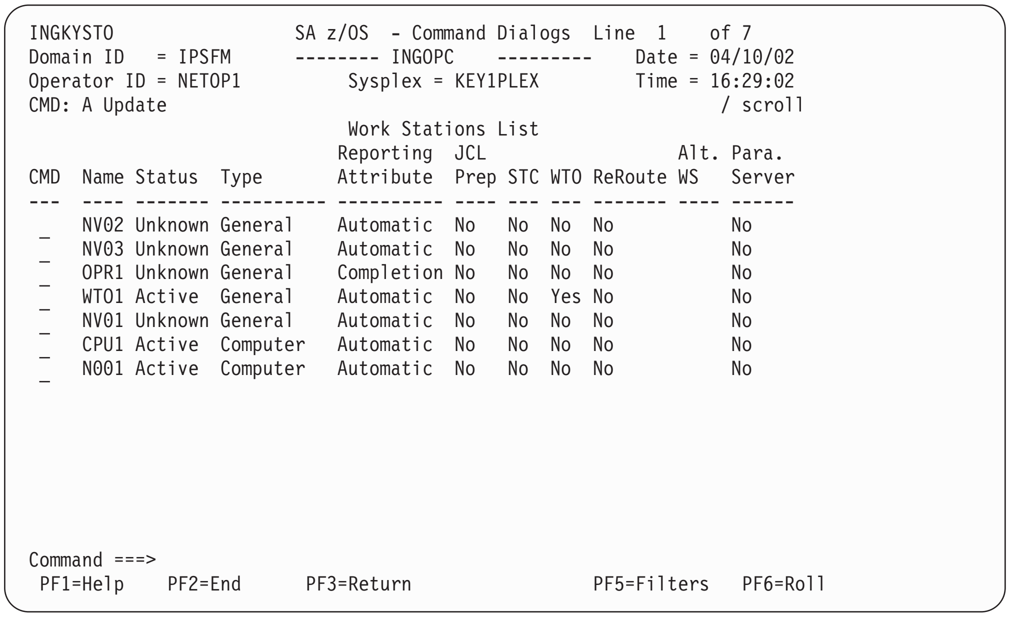
If you specify `INGOPC * REQ=LIST TYPE=SR` a panel similar to Figure 27 is displayed.



```
INGKYSTO                 SA z/OS  - Command Dialogs  Line  1    of 26
Domain ID   = IPSFM      -------- INGOPC    ---------    Date = 04/10/02
Operator ID = NETOP1          Sysplex = KEY1PLEX         Time = 16:27:11
CMD: A Update                                                    / scroll
                            Special Resources List
                                              --Actual-- -Default--
CMD  Name                                     Av. Quant. Av. Quant.
---  ---------------------------------------- --- ------ --- ------
 _   ING.KEY1.APL.CICSK1G.DOWN                No       1 Yes      1
 _   ING.KEY1.APL.CICSK1G.UP                  No       1 Yes      1
 _   ING.KEY1.APL.RMF.DOWN                    No       1 Yes      1
 _   ING.KEY1.APL.RMF.UP                      Yes      1 Yes      1
 _   ING.KEY1.APL.RMFIII.DOWN                 No       1 Yes      1
 _   ING.KEY1.APL.RMFIII.UP                   Yes      1 Yes      1
 _   ING.KEY2.APL.CICSK1G.DOWN                Yes      1 Yes      1
 _   ING.KEY2.APL.CICSK1G.UP                  No       1 Yes      1
 _   ING.KEY2.APL.RMF.DOWN                    No       1 Yes      1
 _   ING.KEY2.APL.RMF.UP                      Yes      1 Yes      1
 _   ING.KEY2.APL.RMFIII.DOWN                 No       1 Yes      1
 _   ING.KEY2.APL.RMFIII.UP                   Yes      1 Yes      1
 _   ING.KEY3.APL.CICSK1G.DOWN                Yes      1 Yes      1

Command ===>
 PF1=Help    PF2=End      PF3=Return                PF5=Filters   PF6=Roll
```

Figure 27. INGOPC REQ=LIST TYPE=SR Sample Panel

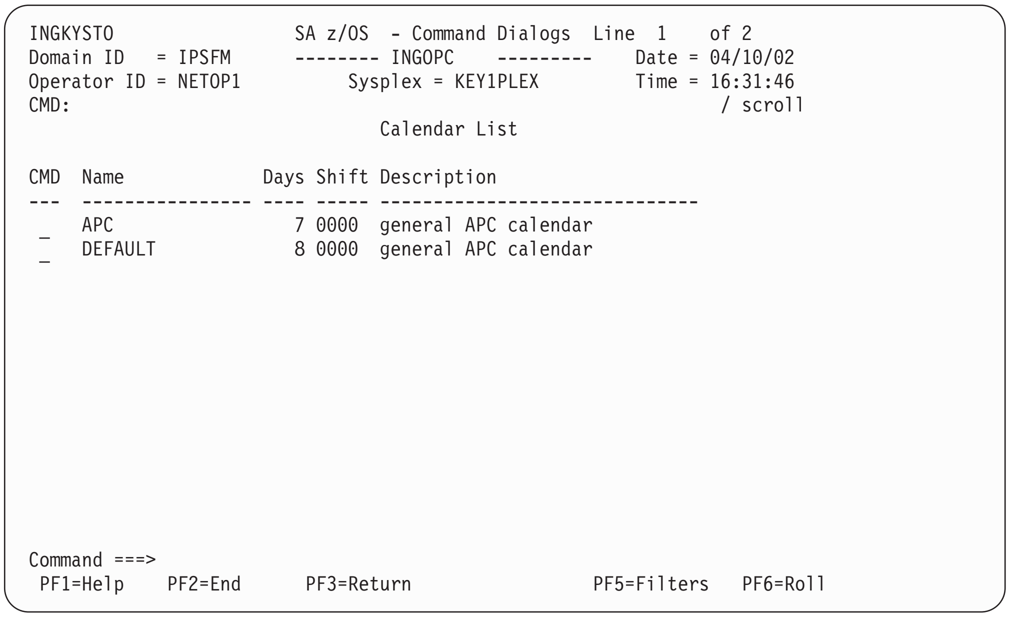
If you specify `INGOPC * REQ=LIST TYPE=WS` a panel similar to Figure 28 is displayed.



```
INGKYSTO                 SA z/OS  - Command Dialogs  Line  1    of 7
Domain ID   = IPSFM      -------- INGOPC    ---------    Date = 04/10/02
Operator ID = NETOP1          Sysplex = KEY1PLEX         Time = 16:29:02
CMD: A Update                                                    / scroll
                              Work Stations List
                             Reporting  JCL                  Alt. Para.
CMD  Name Status  Type       Attribute  Prep STC WTO ReRoute WS   Server
---  ---- ------- ---------- ---------- ---- --- --- ------- ---- ------
 _   NV02 Unknown General    Automatic  No   No  No  No           No
 _   NV03 Unknown General    Automatic  No   No  No  No           No
 _   OPR1 Unknown General    Completion No   No  No  No           No
 _   WTO1 Active  General    Automatic  No   No  Yes No           No
 _   NV01 Unknown General    Automatic  No   No  No  No           No
 _   CPU1 Active  Computer   Automatic  No   No  No  No           No
 _   N001 Active  Computer   Automatic  No   No  No  No           No







Command ===>
 PF1=Help    PF2=End      PF3=Return                PF5=Filters   PF6=Roll
```

Figure 28. INGOPC REQ=LIST TYPE=WS Sample Panel

If you specify `INGOPC * REQ=LIST TYPE=CAL` a panel similar to Figure 29 on page 47 is displayed.

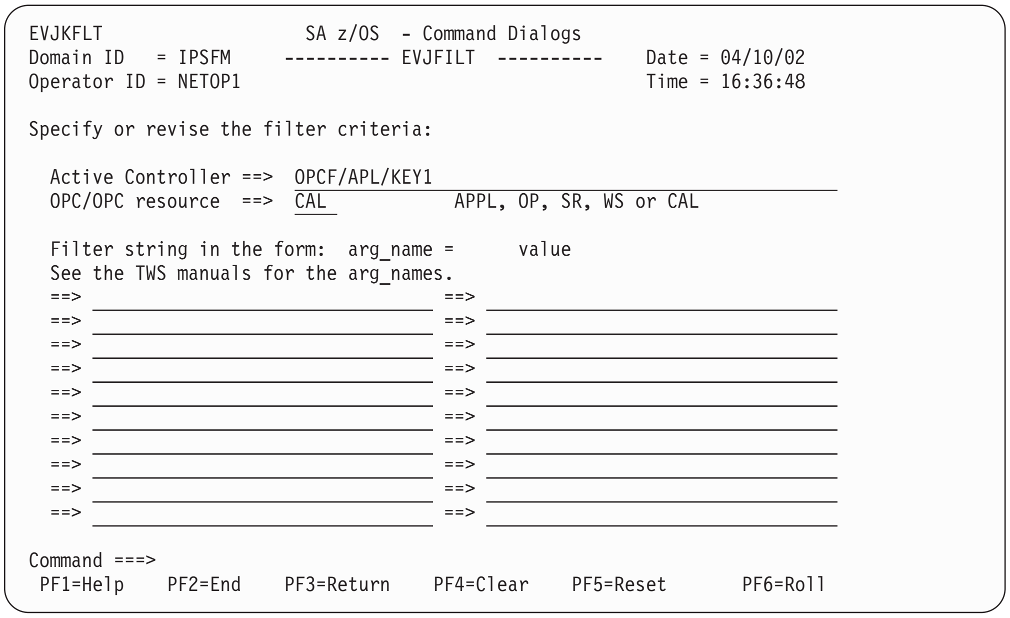


```
INGKYSTO                  SA z/OS  - Command Dialogs  Line  1    of 2
Domain ID   = IPSFM       -------- INGOPC    ---------    Date = 04/10/02
Operator ID = NETOP1           Sysplex = KEY1PLEX         Time = 16:31:46
CMD:                                                              / scroll
                                 Calendar List

CMD  Name             Days Shift Description
---  ---------------- ---- ----- ------------------------------
 _   APC                 7 0000  general APC calendar
 _   DEFAULT             8 0000  general APC calendar













Command ===>
 PF1=Help    PF2=End      PF3=Return                PF5=Filters   PF6=Roll
```

Figure 29. INGOPC REQ=LIST TYPE=CAL Sample Panel

Press PF10 and PF11 to display more information for each resource type. Issuing the command code A Update command against a resource in the CMD field displays a panel that lets you modify the resource. The Application Description list supports the B Operations command code. Issuing this command code against an application resource displays a list of operations for that resource. SORT/FIND/RFIND commands are supported. Refer to "Deciding the Format of the Command Output (Fullscreen only)" in *IBM Tivoli System Automation for z/OS Operator's Commands* for further information.

Pressing PF5 displays a filter selection panel similar to Figure 30 is displayed.



```
EVJKFLT                    SA z/OS  - Command Dialogs
Domain ID   = IPSFM     ---------- EVJFILT  ----------    Date = 04/10/02
Operator ID = NETOP1                                      Time = 16:36:48

Specify or revise the filter criteria:

  Active Controller ==>  OPCF/APL/KEY1
  OPC/OPC resource  ==>  CAL           APPL, OP, SR, WS or CAL

  Filter string in the form:  arg_name =      value
  See the TWS manuals for the arg_names.
  ==> ________________________________  ==> _________________________________
  ==> ________________________________  ==> _________________________________
  ==> ________________________________  ==> _________________________________
  ==> ________________________________  ==> _________________________________
  ==> ________________________________  ==> _________________________________
  ==> ________________________________  ==> _________________________________
  ==> ________________________________  ==> _________________________________
  ==> ________________________________  ==> _________________________________
  ==> ________________________________  ==> _________________________________
  ==> ________________________________  ==> _________________________________

Command ===>
 PF1=Help    PF2=End    PF3=Return    PF4=Clear    PF5=Reset       PF6=Roll
```

Figure 30. INGOPC Filter Sample Panel

Specify filter strings in the format *field-name op contents* where:

- *field-name* is a valid field name as specified by the MODIFY command arguments in *TWS for z/OS Programming Interfaces*.
- *op* can have the following values:

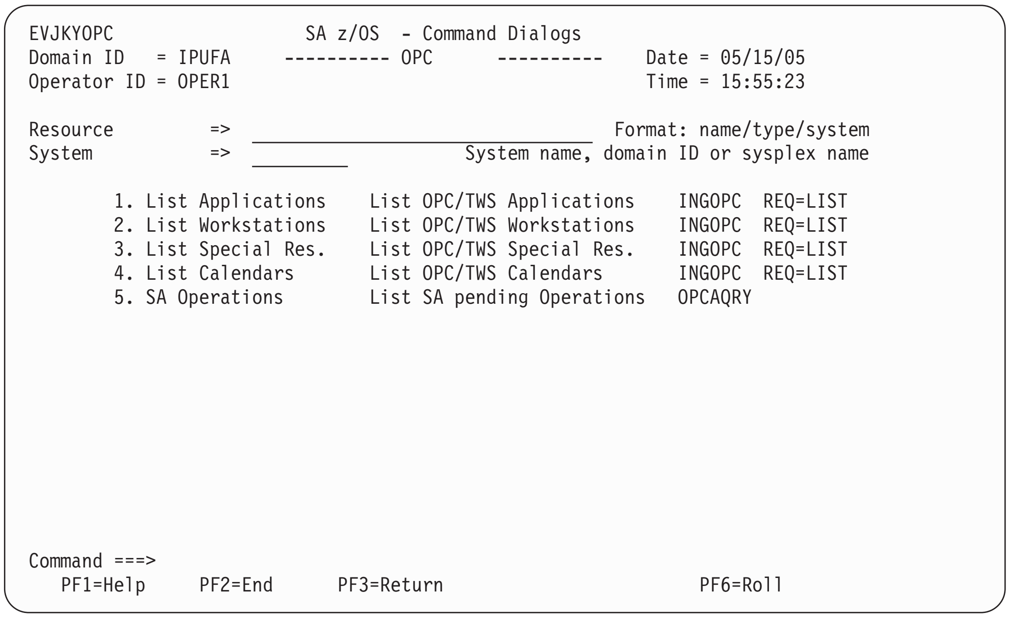
```
    =     ^=     <     <=     >     >=
```

- *contents* are the desired values to be matched by the *op*. The trailing wildcard character '*' may be used for *op*.

The operands must be separated by a blank.

TWS Automation Main Menu

Type **OPCA** on the command line. After you press Enter, TWS Automation displays the TWS Main Menu, as shown in Figure 31.



```
EVJKYOPC                  SA z/OS  - Command Dialogs
Domain ID   = IPUFA     ---------- OPC      ----------    Date = 05/15/05
Operator ID = OPER1                                       Time = 15:55:23

Resource        =>  ________________________________  Format: name/type/system
System          =>  _________            System name, domain ID or sysplex name

        1. List Applications    List OPC/TWS Applications    INGOPC  REQ=LIST
        2. List Workstations    List OPC/TWS Workstations    INGOPC  REQ=LIST
        3. List Special Res.    List OPC/TWS Special Res.    INGOPC  REQ=LIST
        4. List Calendars       List OPC/TWS Calendars       INGOPC  REQ=LIST
        5. SA Operations        List SA pending Operations   OPCAQRY











Command ===>
   PF1=Help     PF2=End      PF3=Return                        PF6=Roll
```

Figure 31. TWS Main Menu

To define one or more OPC Subsystems to be processed, type in the name of a resource. This should be in the format name/type/system.

To display a list of all known OPC resources type a question mark (?) for the resource and specify the required system. The **System** field allows you to specify a TARGET parameter where you would like the required OPC function to be performed (system, domain or sysplex name). To limit the resource list to a specific system, use the */APL/system notation.

Otherwise, type the number of one of the listed functions and press Enter.

OPCAPOST—Posting a TWS Operation from SA z/OS

Purpose

This command is used by SA z/OS to inform TWS of status changes. This is accomplished by the OPCAPOST command processor, which is normally used internally in TWS Automation. Although you can issue OPCAPOST as an operator command, operators should use INGOPC TYPE=OP REQ=MODIFY, if possible. INGOPC provides a fullscreen interface to TWS and dynamically acknowledges the action unlike OPCAPOST.

If you determine that you must use the OPCAPOST command, refer to "OPCAPOST" on page 117.

OPCAQRY—Display Status of Operations

Purpose

The OPCAQRY command displays the status of TWS Automation operations, including all commands that are received via the request interface.

Syntax

```
►►──OPCAQRY──┬───────────┬──┬────────────┬──┬──────────────────────────┬──────────────►
             └─subsystem─┘  └─REQ=DETAIL─┘  └─TARGET=──┬─system_name──┬─┘
                                                       ├─domain_ID────┤
                                                       ├─sysplex_name─┤
                                                       └─*ALL─────────┘

►──┬──────────────────────┬──────────────────────────────────────────────────────────►◄
   └─OUTMODE=──┬─LINE───┬─┘
               ├─AUTO───┤
               └─NETLOG─┘
```

Parameters

subsystem
: The name of the subsystem. Unless you specify REQ=DETAIL, more than one subsystem name as well as a wildcard can be specified. The wildcard can be, for example, SAP*, *SAP or *SAP*

REQ=DETAIL
: Displays TWS-related information for the specified subsystem. The resource name is mandatory when REQ=DETAIL is specified.

TARGET
: For information on the TARGET parameter, refer to *IBM Tivoli System Automation for z/OS Operator's Commands*.

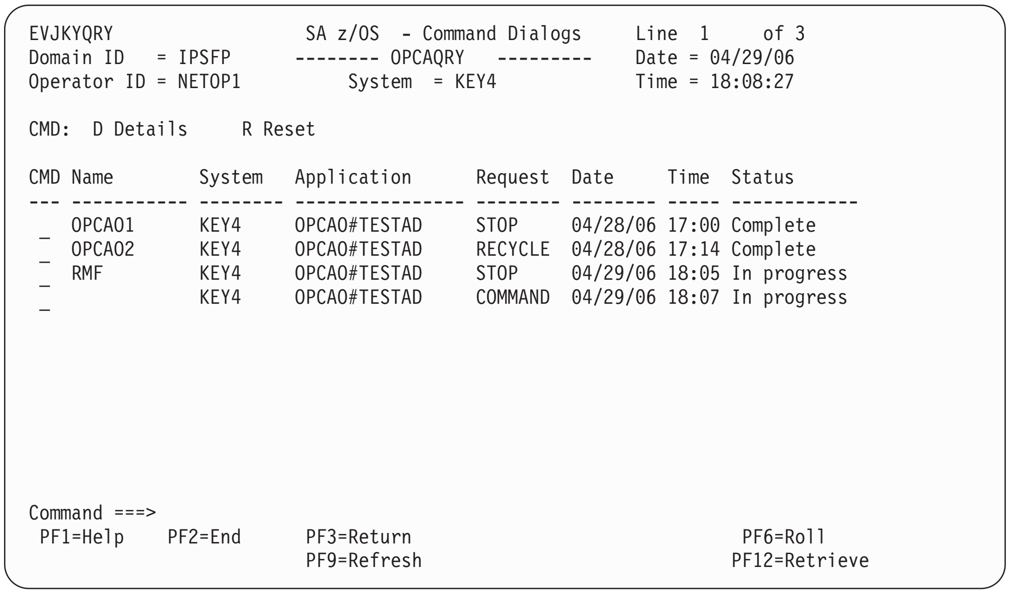
OUTMODE
: For information on the OUTMODE parameter, refer to *IBM Tivoli System Automation for z/OS Operator's Commands*.

Restrictions

The OPCAQRY command can only be used when SA z/OS is initialized.

Examples

If you enter the OPCAQRY command without the REQ=DETAIL parameter, a panel similar to Figure 32 on page 52 is displayed. The panel shows information about TWS-controlled subsystems that match the filter criteria.



```
EVJKYQRY                 SA z/OS  - Command Dialogs     Line  1     of 3
Domain ID   = IPSFP     -------- OPCAQRY   ---------    Date = 04/29/06
Operator ID = NETOP1          System  = KEY4            Time = 18:08:27

CMD:  D Details     R Reset

CMD Name        System   Application      Request  Date     Time  Status
--- ----------- -------- ---------------- -------- -------- ----- ------------
 _  OPCAO1      KEY4     OPCAO#TESTAD     STOP     04/28/06 17:00 Complete
 _  OPCAO2      KEY4     OPCAO#TESTAD     RECYCLE  04/28/06 17:14 Complete
 _  RMF         KEY4     OPCAO#TESTAD     STOP     04/29/06 18:05 In progress
 _              KEY4     OPCAO#TESTAD     COMMAND  04/29/06 18:07 In progress









Command ===>
 PF1=Help    PF2=End      PF3=Return                              PF6=Roll
                          PF9=Refresh                            PF12=Retrieve
```

Figure 32. OPCAQRY Command Dialog Panel

- The **CMD** field allows you to specify command codes to invoke another command dialog. The following command codes are available:

  D Shows the TWS operation details for the subsystem.

  R Resets the timer and completion flags to a null value, and unlocks a specific subsystem after a user error has been detected and corrected. By resetting the timer and completion flags, SA z/OS again accepts requests from TWS.

  For a command entry, the entry is removed by deleting the CGlobals that are used to keep track of the command processing.

- The **Status** field shows the status of the request or command in SA z/OS:
  - For a request, the status can be:

    Complete
    The operation completed successfully. This is considered to be a normal status.

    Incomplete
    This indicates that the operation did not achieve the expected status set by the system programmer in the OPCA code entry.

    Timeout
    This indicates that the operation is marked in error because it did not complete within the time limit set by the system programmer in the OPCA code entry.

    In progress
    The request has been received and processing has been started.

    No request
    No request was made
  - For a command, the status can be:

    In progress
    The command has been received and processing has been started.

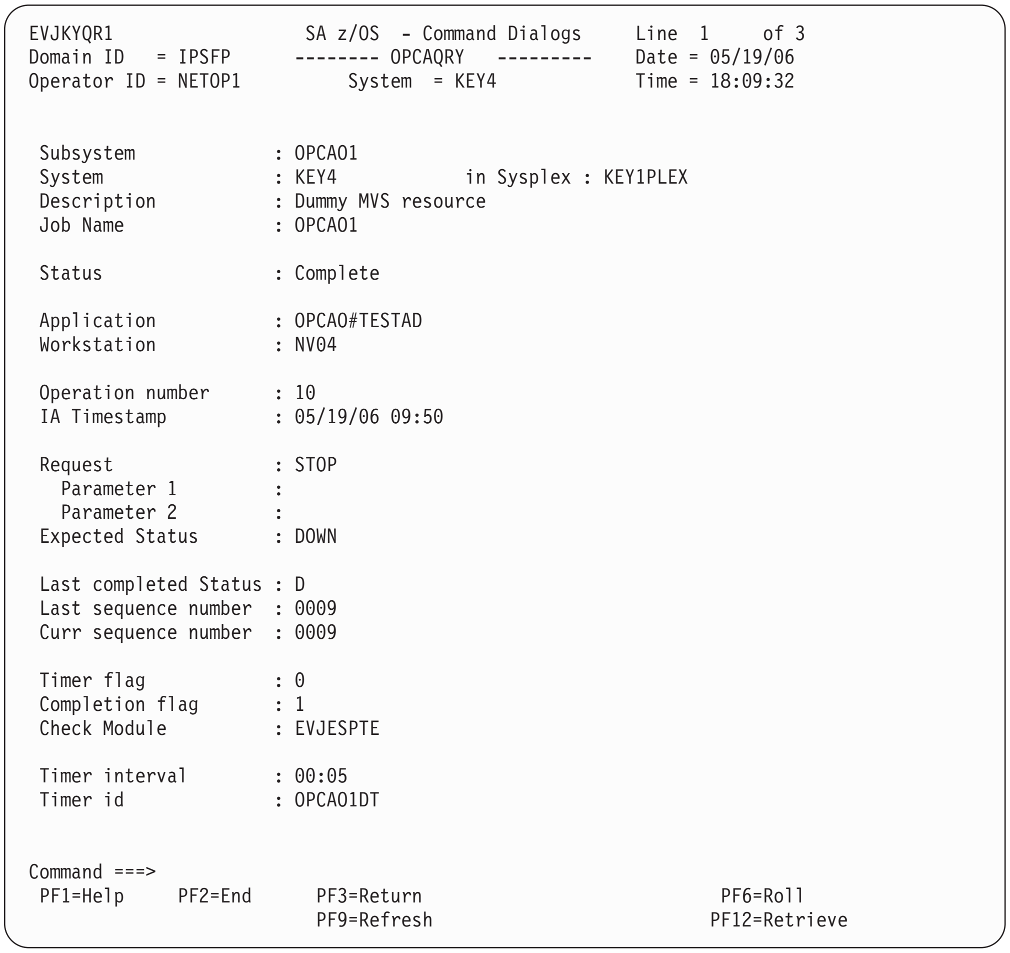
    In error
    The command completed but failed.

Complete
: The command completed successfully. This is considered to be a normal status.

Timeout
: The command did not finish processing within the time that was specified in the completion information for the command.

Waiting
: The command finished processing but is now waiting for completion.

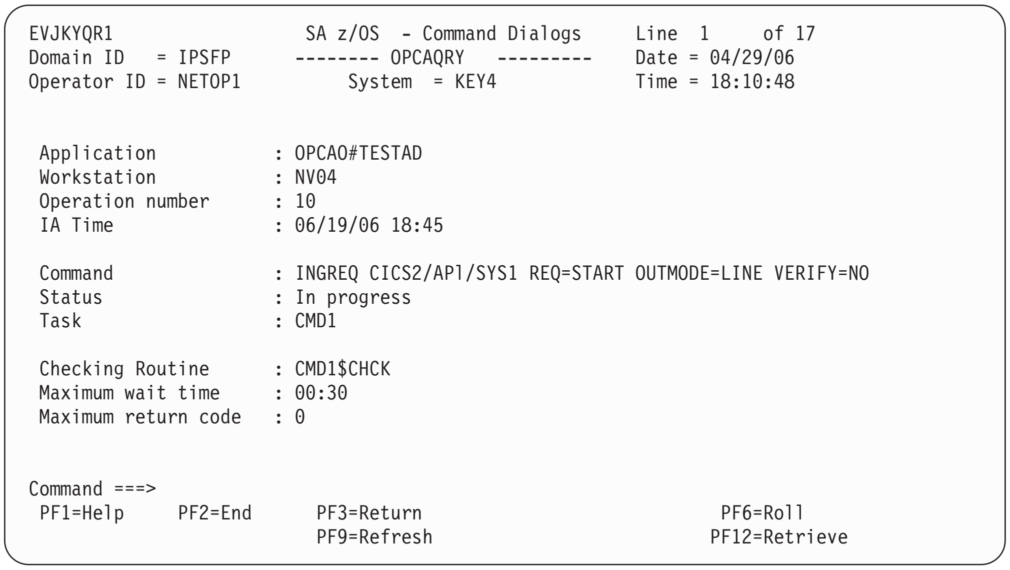
If you enter command code D for a subsystem or specify the REQ=DETAIL option, a panel similar to Figure 33 is displayed.



```
EVJKYQR1                  SA z/OS  - Command Dialogs     Line  1     of 3
Domain ID   = IPSFP      -------- OPCAQRY   ---------    Date = 05/19/06
Operator ID = NETOP1          System  = KEY4             Time = 18:09:32


 Subsystem            : OPCAO1
 System               : KEY4            in Sysplex : KEY1PLEX
 Description          : Dummy MVS resource
 Job Name             : OPCAO1

 Status               : Complete

 Application          : OPCAO#TESTAD
 Workstation          : NV04

 Operation number     : 10
 IA Timestamp         : 05/19/06 09:50

 Request              : STOP
   Parameter 1        :
   Parameter 2        :
 Expected Status      : DOWN

 Last completed Status : D
 Last sequence number  : 0009
 Curr sequence number  : 0009

 Timer flag           : 0
 Completion flag      : 1
 Check Module         : EVJESPTE

 Timer interval       : 00:05
 Timer id             : OPCAO1DT


Command ===>
 PF1=Help     PF2=End      PF3=Return                         PF6=Roll
                           PF9=Refresh                       PF12=Retrieve
```

Figure 33. OPCAQRY Command Dialog Panel Showing Details for a Subsystem

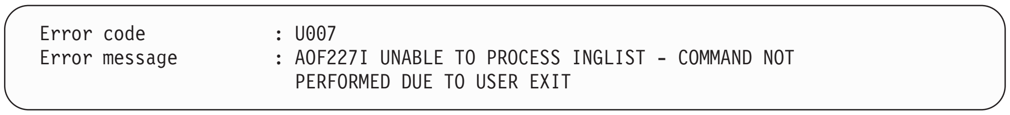
If details are requested for a command, a panel similar to Figure 34 on page 54 is displayed.



```
EVJKYQR1                  SA z/OS  - Command Dialogs     Line  1     of 17
Domain ID   = IPSFP      -------- OPCAQRY   ---------    Date = 04/29/06
Operator ID = NETOP1          System  = KEY4             Time = 18:10:48


 Application           : OPCAO#TESTAD
 Workstation           : NV04
 Operation number      : 10
 IA Time               : 06/19/06 18:45

 Command               : INGREQ CICS2/APl/SYS1 REQ=START OUTMODE=LINE VERIFY=NO
 Status                : In progress
 Task                  : CMD1

 Checking Routine      : CMD1$CHCK
 Maximum wait time     : 00:30
 Maximum return code   : 0


Command ===>
 PF1=Help     PF2=End      PF3=Return                          PF6=Roll
                           PF9=Refresh                        PF12=Retrieve
```

Figure 34. OPCAQRY Showing Details for a Command

If the command is in an error condition, details similar to those in Figure 35 are displayed.



```
 Error code            : U007
 Error message         : AOF227I UNABLE TO PROCESS INGLIST - COMMAND NOT
                         PERFORMED DUE TO USER EXIT
```

Figure 35. OPCAQRY Details for a Command in Error

SRSTAT—Setting TWS Special Resource Status

Purpose

This command lets you update the status of the specified TWS special resource to the value given in the parameters. The status is returned via messages.

Syntax

SRSTAT *srname*,**SUBSYS=***subsys*,**AVAIL=Y|N**

Parameters

srname
Special resource name — up to 44 characters.

Note: The special resource name must be enclosed in single quotes if it contains any spaces or commas.

subsys
The MVS subsystem ID of the TWS tracker — 4 characters.

AVAIL=Y|N
Availability indicator.

Return Codes

SRSTAT has the following return codes:

0	Okay.
1	Bad parameters.
2	Invalid special resource name.
3	Invalid subsystem ID.
4	Invalid availability status.

Example

```
SRSTAT EOD.CICSPRD1.TRANS,SUBSYS=OPCT,AVAIL=Y
```

In this example, end-of-day transactions are required to finish before production work can begin. SRSTAT is executed when the transactions are complete. The special resource name EOD.CICSPRD1.TRANS is used to trigger OPC/ESA applications that are able to run when the transactions are finished. A number of applications are added to the current plan.

The variable used for *subsys*, OPCT, is the name of the tracker subsystem. The tracker subsystem name is only required with this command.

Part 3. Programmer's Reference

This part describes the information needed by system programmers to install and customize the TWS Product Automation of System Automation for z/OS.

Subtopics:

© Copyright IBM Corp. 1990, 2010

Chapter 7. Installing TWS Automation

This chapter describes the steps to follow when installing TWS Automation.

Enabling and Disabling TWS Automation

SA z/OS TWS Automation may be disabled and enabled by specification of subsystems with an application type of OPC.

To disable SA z/OS TWS Automation, do not specify any TWS applications in the Policy Database for the system that is to have TWS Automation disabled. Disabling occurs on a system-by-system basis, so by not linking TWS type applications to a system, automatically disables TWS Automation.

Disabling TWS Automation causes the message traps in the SA z/OS NetView to also be disabled. This will speed message processing for those systems that do not participate in TWS functions. Disabling TWS Automation does not prevent execution of the INGOPC command. As long as at least one system in the sysplex contains a Controller or Tracker, the INGOPC command will work.

If you disable SA z/OS TWS Automation for any reason, be sure to unlink the TWS Command Receiver NON-MVS subsystem and the TWS Request Receiver NON-MVS subsystem from systems that TWS Automation is disabled for.

To enable SA z/OS TWS Automation, specify the TWS applications for the Controllers and Trackers in the Policy Database of all systems that have either Controllers and Trackers running on them. Include any Trackers that belong to foreign Controllers; that is, Controllers not present anywhere in the sysplex that the Trackers belong to.

Defining System Automation Policy

Several automation policy items are required for correct operation of TWS Automation. These policy items are:

- The Automation Operators that are required for function enablement.
- The required non-MVS subsystem that is the PPI request receiver. This subsystem provides support to pass requests and command requests from TWS to SA z/OS.
- The optional non-MVS subsystem that defines the PPI batch command receiver.
- The definition of TWS Controller, Tracker, Server, and Data Store.
- The definition of workstation names to be automatically activated on SA z/OS Agent startup (command interface).
- Definition of the status observer subsystem to ensure that SA z/OS status changes are reflected in TWS special resource statuses.

Detailed instructions for defining this configuration can be found in *IBM Tivoli System Automation for z/OS Defining Automation Policy* and in the *TWS add-on policy database.

Table 5 on page 60 shows the definitions that are required for TWS Automation or the different interfaces that you can use with TWS.

Table 5. Policy Definition Requirements for TWS Automation

Definition	Request Interface	Command Request Interface	Command Interface
Optional Workstations	–	–	✓
TWS Request Server	✓	✓	–
TWS Command Server	–	–	✓
Workstation Domain Entries	✓	✓ (possibly)	–
Controller Details	Always required for TWS Automation		
System Details	Always required for TWS Automation		
Special Resources Policy	Always required for TWS Automation		
Subsystem Messages/User Data	✓	–	–

Define SA z/OS Automation Operators

The automation operators that are required for correct operation of SA z/OS TWS Automation are listed in Table 6.

Table 6. Automation Operators

Automation Operator	Description	Required for	Messages
AOFTWS*nn*	TWS command request execution operator	The command request interface	none
OPCAMSTR	Main Automation Operator	TWS Automation	EVJ*
OPCAOPR2	TWS Request execution operator	TWS Automation	none
OPCACMDR	TWS Batch Command Execution operator	The batch command interface	none

These automation operator definitions can be found in the *TWS add-on sample PDB definitions under the Auto Operators policy with the name "TWS_AUTO_OPS".

Automated Operator Tasks

TWS is an SA z/OS-controlled subsystem. Normal definitions in the SA z/OS policy database can describe TWS. In addition, SA z/OS defines an automated operator task (called an *automated function* by SA z/OS) for the TWS Controller in the system containing the Controller, as well as one for the TWS Tracker in each system. These automated operator tasks perform the TWS-requested functions in the SA z/OS application.

TWS Automation requires actions in a specific order. Changes in this order can result in unpredictable and undesirable results. To ensure that a proper sequence of processing is maintained, you must complete the actions in a single-thread fashion. In TWS, this is the responsibility of the user and is achieved through dependency control or critical resource specifications.

NetView maintains this control by ensuring that actions are executed sequentially through the use of automated operator tasks. Specify only one automated operator task for the TWS Controller functions and only one for the Tracker functions. Stipulating any additional automated operator tasks for TWS Automation results in loss of synchronization. This, in turn, can create an uncontrolled environment, requiring a substantial amount of operator/system programmer effort to recover, and additional loss of synchronization until a single automated operator task for the Tracker and Controller functions is reinstated. When TWS Automation detects any violations, it checks for out-of-sequence requests and stops processing for a specific application through an error code to TWS Automation.

However, separate automated operator tasks for Controller and Tracker are required. Running TWS Automation on the Controller system with a single automated operator task specified for both Controller and Tracker functions results in a lockout condition. Consider this especially on backup systems, which do not normally run Controller functions. If you specify only one automated operator task for both systems, each task runs properly until they become an active backup system and lock.

For the automated operator task OPCAMSTR, the operator ID must be AUTOPCP. For the automated operator task OPCAOPR2, you may specify whatever operator ID meets your installation standards. However, do not change the TWS Automation operator task names OPCAMSTR and OPCAOPR2.

For *automation* workstations you must also specify the automated operator tasks AOFTWS*xx*. Generally, you should define one automated operator task for each workstation, however, if a workstation is associated with multiple servers that work together in parallel, then you will need to specify enough automated operator tasks to cover these.

RMTCMD Security Considerations

NetView RMTCMD is used to communicate with remote domains (that is, gateway connected domains outside the system or sysplex where the TWS Controller is running). RMTCMD will be used if TWS Automation is controlling applications on a remote domain and recovery is required for the TWS Automation-controlled operations after the remote system or gateway has failed.

The operator IDs for OPCAMSTR and OPCAOPR2 (AUTOPCP and AUTOPCE or user specified operator ID) must have the appropriate NetView or RACF® authority to use RMTCMD on the local system (that is, the system running the TWS Controller). NetView or RACF definitions may also be required on the remote systems. For more details about the NetView or RACF security implications when using RMTCMD please refer to the *Tivoli NetView Security Reference*.

Define Optional Workstations

This is required for the command interface.

The batch jobs that execute NetView and SA z/OS commands may be submitted by any TWS Computer/Automated Workstation. However, if the NetView PPI receiver is not operational at the time of execution of the batch job, the batch job may optionally set the workstation that submitted it to an inoperative state. If this function is to be used, which is the default for the batch job, then it may be prudent to create additional batch job submission workstations to which these NetView-related batch jobs are assigned. This will allow the rest of the batch job stream to be submitted, whilst holding the NetView-related jobs until NetView starts.

The Workstations that are automatically re-enabled at SA z/OS startup are those defined on the WORKSTATION User message policy for either the Trackers or Controllers defined to SA z/OS. An example of this is shown in Figure 36, which shows the definition of workstation N001, where:

- CODE1 represents the name of the sysplex that the Batch Command Server non-MVS subsystem is running on.
- CODE2 represents the name of the system that the Batch Command Server non-MVS subsystem is running on.
- CODE3 is not used.

This allows the same Controller or Tracker subsystem definition to be run on different systems or sysplexes and the name of the workstation can be different on each sysplex or system combination.

```
Code 1          Code 2          Code 3          Value Returned
*______________ *______________ *______________ N001______________________________
```

Figure 36. Defining Workstation User Message Policy

Non-MVS Subsystem Definition for the TWS Request Server

This is required for the request interface and the command request interface.

This non-MVS subsystem is required to allow requests from Tivoli Workload Scheduler to SA z/OS.

See the *TWS add-on sample PDB that contains the subsystem definition TWSREQR. This subsystem definition contains the policy definition for the TWS Request Server.

For details of relationships see the *TWS add-on policy database and its documentation.

Non-MVS Subsystem Definition for the TWS Command Server

This is required for the command interface.

This non-MVS subsystem is required to allow commands from the batch command interface to SA z/OS.

See the *TWS sample add-on PDB that contains the subsystem definition TWSCMDR. This subsystem definition contains the policy definition for the TWS Command Server.

For details of relationships see the *TWS add-on policy database and its documentation.

Define Workstation Domain Entries

This is required for the request interface and possibly also the command request interface.

Customize the WORKSTATION DOMAINS (ODM entry type) policy objects in the SA z/OS policy database and connect them to all systems where the TWS controller may run.

These policy objects map TWS workstations to NetView domain IDs to show where the requests should be routed to be executed.

Define Controller Details

This is always required for TWS Automation.

Customize the CONTROLLER DETAILS (OCS entry type) policy objects in the SA z/OS policy database and connect them to all systems where the TWS controller may run and all systems where applications will be automated by TWS Automation (that is, Tracker only systems).

These objects specify the location of a controller and can be associated with a set of TWS special resources. See *IBM Tivoli System Automation for z/OS Defining Automation Policy* for more information.

Note: If multiple controllers are running on the same system they must share the same Controller Details (OCS) definitions. If more than one OCS policy object is linked to the same system, the last is used.

Define System Details

This is always required for TWS Automation.

Customize the OPC SYSTEM DETAILS policy objects (OEN entry type) in the SA z/OS policy database and connect them to systems where the TWS controller may run and to all systems where applications will be automated by TWS Automation (that is, Tracker only systems).

These objects contain control information for TWS Automation, such as the checking of subsystem status of START and STOP requests, the retention of critical messages, the operation reset delay, and PPI name of the Batch Interface Server. See *IBM Tivoli System Automation for z/OS Defining Automation Policy* for more details.

Define Special Resources Policy

This is always required for TWS Automation.

If required, define TWS special resources as TWS SPECIAL RESOURCES policy objects (OSR entry type) in the SA z/OS policy database and link them to CONTROLLER DETAILS objects. See *IBM Tivoli System Automation for z/OS Defining Automation Policy* for more information.

Define or Modify Subsystem Messages/User Data

This is required for the request interface.

You must define each subsystem you wish to automate from TWS to SA z/OS. In many cases, you will also have to define OPCA and OPCACMD entries in the MESSAGES/USER DATA policy item for these subsystems (see “Executing TWS Requests with TWS Automation” on page 82).

Note: You do not have to define any OPCA and OPCACMD entries if you are using only *automation* workstations.

Defining the SA z/OS Status Observer

SA z/OS TWS Automation provides a facility that echoes the status of SA z/OS resources in TWS Special Resources. This facility allows you to define TWS operations that will wait until SA z/OS resources reach a desired state. Currently only two desired states are allowed:

- The UP state is when the automation manager sets the resource to the AVAILABLE state.
- The DOWN state is when the automation manager sets the resource to the UNAVAILABLE state.

The Status observer is implemented as an automation agent function. This function registers with the automation manager and receives status changes. It then translates the status changes to the appropriate TWS special resources and issues an MVS subsystem broadcast to all TWS Controllers and Trackers on the system that the agent is running on.

The SA z/OS Status Observer is defined as a non-MVS SA z/OS subsystem. This subsystem should run on the system that contains the TWS Controller or alternatively a TWS Tracker.

For details of relationships see the *TWS add-on policy database and its documentation.

Chapter 8. Submitting NetView Commands from a Batch Job

This chapter describes how to execute NetView commands from a Batch job. This is particularly useful for Tivoli Workload Scheduler, but can be used stand alone without either Tivoli Workload Scheduler or Operations Planning for Control.

Sample Batch Job JCL

A sample batch job can be found in the System Automation for z/OS Installation library SINGSAMP. Member EVJSJ001 contains the sample JCL. The batch job must be run on the same system as the SA z/OS Agent that contains the Command receiver specified by the batch job. In most cases there will be a Command receiver running on every SA z/OS Agent. However, customization of the Command receivers can alter the names of the Command receivers and also the number and configuration of the Command receivers. You should check with your system programmers to determine the correct system and command receiver to use for these batch jobs.

Command Statement Syntax

The commands supplied to the batch job in the //SYSIN ddname have the following syntax:

```
►►──┬─line-mode-command──┬─────────┬──┬─────┬─┬──────────────────────────►◄
    │                    └─>DDNAME─┘  └─"-"─┘ │
    └─*comment────────────────────────────────┘
```

1. All blank lines are ignored.
2. All lines starting with an asterisk (*) are comment lines and are printed in the output but otherwise ignored.
3. Comments on the end of commands are not allowed.
4. Comments are not allowed between continuation lines.
5. A command can be continued by appending a dash (-) to the line.
6. Command output normally goes to //SYSTSPRT.
7. Command output may be redirected to other DDNAMEs. The default for this is the right angle bracket (>) symbol.
8. PIPE > stage is prohibited. Use PIPE QSAM instead.
9. Fullscreen commands are not allowed.

Valid Command Types

Any command, clist or REXX program that issues correlated line messages may be used.

This means almost all NetView commands, all SA z/OS commands that support OUTMODE=LINE and any clist or REXX program that either issues SAY messages or PIPES the messages to CONSOLE.

The return code from the command can be used to stop the remaining commands from being executed. See the "MAXRC parameter" on page 69 of the EVJRYCMD procedure definition.

© Copyright IBM Corp. 1990, 2010

Command Continuation

Commands are continued across lines by appending a dash to the end of the command, for example:

```
PIPE NETVIEW LIST STATUS=OPS | -
CONSOLE ONLY
```

Command Output Redirection

Normally command output is printed on the //SYSTSPRT DDNAME. However, the output of commands may be redirected to other DDNAMEs. This is achieved via a redirection symbol. The default is the > symbol, for example:

```
PIPE NETVIEW LIST STATUS=OPS | CONSOLE ONLY >MYOUTPUT
```

This allows subsequent steps in the batch job or other batch jobs to use the output of the command for their own purpose.

You can change the redirection symbol with the REDIRECT parameter of EVJRYCMD if, for example, you use > as a command prefix. Note that the redirection symbol must not be the same as any of the characters that occur in the command. For more details, see “EVJRYCMD Description” on page 68.

The DCB characteristics of the output DDNAME should be as follows:

```
LRECL=132,RECFM=FB
```

Executing a Command on a Different NetView

Almost all SA z/OS commands can specify the TARGET= parameter to force the command to execute on the target system. If a command does not have this facility, for example the NetView LIST command, you can use PIPE labels to send the command to the appropriate NetView, for example:

```
PIPE CC dom01: LIST STATUS | CONSOLE ONLY
```

Or even:

```
PIPE CC dom01/auto1: LIST STATUS=OPS | CONSOLE ONLY
```

JCL for the Batch Command Interface

Figure 37 on page 67 shows the sample from the product sample library (SINGSAMP).

```
//*--------------------------------------------------------------***
//S0       EXEC PGM=IEFBR14
//CONCAT   DD  DSN=TEMP.CONCAT.LIST,
//             DISP=(MOD,DELETE,DELETE),
//             UNIT=SYSDA,SPACE=(1,1),AVGREC=M,
//             LRECL=132,RECFM=FB,STORCLAS=SMS
//*--------------------------------------------------------------***
//*%OPC SCAN
//S1       EXEC PGM=IKJEFT01,DYNAMNBR=30,REGION=4M,
//         PARM='EVJRYCMD &OWSID SERVER=EVJCMDRV HIGHRC=16'
//STEPLIB  DD  DSN=SYS1.NETV.SEKGLNK1,DISP=SHR   NETVIEW LIBRARY
//SYSPROC  DD  DSN=SYS1.SAM.SINGNREX,DISP=SHR    SA LIBRARY
//EQQMLIB  DD  DSN=SYS1.TWS.SEQQMSG0,DISP=SHR    TWS LIBRARY
//OUTPUT   DD  SYSOUT=*
//CONCAT   DD  DSN=TEMP.CONCAT.LIST,
//             DISP=(MOD,CATLG),
//             UNIT=SYSDA,SPACE=(1,1),AVGREC=M,
//             LRECL=132,RECFM=FB,STORCLAS=SMS
//SYSTSPRT DD  SYSOUT=*
//SYSTSIN  DD  DUMMY
//SYSIN    DD  *
* THIS IS A COMMENT
MVS D A,L >OUTPUT
D NET,MAJNODES >CONCAT

PIPE NETV WHO | -
     NLOC /AUT/ | -
     CONS ONLY
INGLIST */APL/* OUTMODE=LINE >CONCAT
/*
```

Figure 37. Sample JCL for the Batch Command Interface

Ensure that the appropriate NetView library is assigned to //STEPLIB. This library should contain the DSIPHONE module.

S0
This step deletes a temporary listing data set that is used for concatenation of output from the command.

%OPC
This statement is used to tell TWS to begin scanning for TWS substitution variables and to replace them when found with their contents.

S1
This step executes a BATCH TSO TMP to run the REXX command that sends commands to a SA z/OS Agent.

EVJRYCMD
This is the command that runs to process batch commands.

&OWSID
This is a TWS substitution variable that represents the name of the workstation that submitted the job.

//STEPLIB
The NetView library SEKGLNK1 (or equivalent) is used to provide the DSIPHONE REXX function. Substitute the correct library name for the appropriate release of NetView that is being used by SA z/OS.

If Module EQQYCOM cannot be found in the system LINK LIST, then concatenate the appropriate TWS load library here.

//SYSPROC
The SA z/OS library that contains the EVJRYCMD REXX procedure.

//EQQMLIB
The TWS for z/OS message library.

//OUTPUT
Optional output DDNAME for command output redirection.

//CONCAT Optional output DDNAME for command output redirection. In this case a data set that will have successive command output concatenated to it.

//SYSTSPRT Required TSO TMP output DDNAME.

//SYSTSIN Required TSO TMP command input DDNAME. Dummied out because the only command to be executed is in the parameter to the TSO TMP.

//SYSIN Commands to be executed in the SA z/OS Agent. For details of how to start or stop a SA z/OS resource, see "Starting or Stopping an SA z/OS Resource" on page 70.

EVJRYCMD Description

Purpose

EVJRYCMD is a REXX procedure that issues commands to a SA z/OS agent and receives the results of those commands.

```
                                        ┌─SERVER=EVJCMDRV─┐ ┌─TIMEOUT=60──────────┐
►►──EVJRYCMD──wsid──┬─────────┬──┴─────────────────┴─┼─────────────────────┼──────────────►
                    └─NOWKSTS─┘  └─SERVER=server───┘  ├─TIMEOUT=seconds─────┤
                                                      └─TIMEOUT=NONE────────┘

  ┌─HIGHRC=0───────────┐  ┌─MAXRC=999──────────┐  ┌─SYSIN=SYSIN───┐
►─┴────────────────────┴──┴────────────────────┴──┴───────────────┴──────────────────────►
  └─HIGHRC=return_code─┘  └─MAXRC=return_code──┘  └─SYSIN=DDname──┘

  ┌─REDIRECT=>─────┐
►─┴────────────────┴──────────────────────────────────────────────────────────────────►◄
  └─REDIRECT=chars─┘
```

Parameters

wsid This is a required parameter.

This parameter specifies the name of the TWS workstation that submitted this batch job. This information is used by the command to disable the workstation in the event that communications between the batch job and the SA z/OS Agent cannot be established. See "the NOWKSTS parameter" to modify this behavior.

This parameter must be specified, even if it is not to be used.

In the case of the NOWKSTS parameter being specified, or the batch job being submitted manually or by a product other than TWS, specify any non-blank character sequence.

NOWKSTS
This parameter is optional.

This parameter modifies the behavior of the command. In the event of a failure in communications to the SA z/OS Agent, this parameter prevents the command from disabling the TWS workstation that is defined with the *wsid* parameter.

If this parameter is not specified, any NetView PPI communications problem will cause the command to issue a TWS WSSTAT command to place the workstation offline.

SERVER

This parameter is optional.

The default for this parameter is EVJCMDRV. This parameter specifies the name of the PPI receiver in the SA z/OS Agent NetView that commands will be sent to.

TIMEOUT

This parameter is optional.

The default for this parameter is 60 seconds.

This parameter specifies the time in seconds that the batch job will wait for a command to execute in the SA z/OS Agent NetView. This timeout is applied separately to each command. If the timeout is set to NONE, no timeout will be applied to the batch job.

Note: It is recommended that the INGREQ timeout (as defined with the FDBK parameter) should be less than the TIMEOUT= parameter for the job.

This is because the INGREQ command's FDBK parameter can be used to specify a WAIT period that will result in the command waiting until the desired status change is complete. For example, if the TIMEOUT parameter is defaulted to 60 seconds, the INGREQ FDBK parameter should be coded as, say, `FDBK=(WAIT, :55)`

HIGHRC

This parameter is optional.

The default for this parameter is 0 (zero).

This parameter specifies the highest acceptable Return Code for the job. Any return codes from commands that are less then or equal to this value will reset the JCL Step return code to zero. Any command return code that is greater than this value will be passed as the JCL Step return code.

Note: The JCL Step return code will be the highest return code of all the command return codes.

MAXRC

This parameter is optional.

The default for this parameter is 999.

This parameter specifies the maximum acceptable return codes from commands issued by the batch job. If a command return code is higher than the value specified, the batch job is aborted and any remaining commands will not be executed.

The return code that is reported to the JCL is determined by the HIGHRC parameter.

SYSIN

This parameter is optional.

The default for this parameter is SYSIN.

This parameter sets the DDNAME of the input file that contains the command to be executed.

REDIRECT

This parameter is optional.

The default for this parameter is >.

This parameter defines the redirection character. Enclose it in quotes or double-quotes if the string contains special characters, such as the equal sign. It must not be the same as any of the characters that occur in the command.

Usage

When the SA z/OS Agent is started, it will automatically issue a WSSTAT command to mark the workstation online. The specifications of which workstations to mark online at agent restart are contained in the WORKSTATION message/user data policy for the tracker or controller. Multiple workstations may be defined. Workstations that are assigned to trackers should have their WORKSTATION policy defined to the same trackers that they are assigned to. See “Define Optional Workstations” on page 62.

Each command is submitted in turn and the results of the command are retrieved. These results are then written to either SYSTSPRT or to the output redirection DDNAME.

Starting or Stopping an SA z/OS Resource

To start or stop a SA z/OS Resource, use the previous section as a basis.

1. In the JCL for job SAMPJOB copy the sample job EVJSJ001 from SINGSAMP and tailor it for your environment.
2. Put the INGREQ command in the //SYSIN and specify the OUTMODE=LINE option for INGREQ.
3. If you are starting a resource (APL or APG) update the Special Resource for the operation 005 to reflect the name of the resource specified in the INGREQ command. For example, for INGREQ TEST/APG/SYS1 use special resource ING.SYS1.APG.TEST.UP.
4. If you are stopping a resource (APL or APG) update the Special Resource for the operation 005 to reflect the name of the resource specified in the INGREQ command. For example, for INGREQ TEST/APG/SYS1 use special resource ING.SYS1.APG.TEST.DOWN.
5. Remember to create the special resource in the TWS Database via option 1.6 in the TWS dialogs.

The operation is now ready for LTP and Current Plan planning functions.

When TWS submits the first operation, the JCL will run the Batch Command Interface and execute the INGREQ command on the SA z/OS Agent. SA z/OS will then process the command and issue the appropriate orders to the Agents to achieve the desired status. When the resource has achieved the desired status, SA z/OS will notify TWS that this has occurred by updating the special resource. This will cause operation number 005 to be marked Complete - especially if you assign it to a General Non-Reporting workstation. Operations and Applications that have a predecessor of operation number 005 will now be able to run.

Chapter 9. Using TWS Special Resources

This chapter describes the TWS Special Resources that are created by SA z/OS and how to use them.

SA z/OS can dynamically create TWS special resources based on the status of SA z/OS resources. These TWS special resources can in turn be used to control the flow of applications and operations in TWS.

TWS Special Resource Definition

If allowed to, SA z/OS creates TWS Special Resources via policy definitions. The definition of the name of these special resources is as described in "TWS Automation Special Resources" on page 11.

Each SA z/OS Resource (application, application group, system, system group, or monitor resource) has *two* TWS special resources. One tracks the state of the SA z/OS resource in the UP case and the other tracks in the DOWN case. Setting the availability of these two TWS special resources is independent of each other. It is possible for the SA z/OS resource to have both the UP and DOWN TWS special resources UNAVAILABLE. This can occur when the SA z/OS resource is starting for example. It is neither UP or DOWN.

It should not normally be possible to have both TWS special resources AVAILABLE at the same time.

Enabling SA z/OS TWS Special Resources

To enable the SA z/OS TWS Special Resource tracking:

1. Ensure that the SA z/OS Resources that are to be monitored are defined in SA z/OS.

   All these policy items are described in *IBM Tivoli System Automation for z/OS Defining Automation Policy* in the section "Defining Automation for TWS Components" in "Product Automation Policy Object":

   a. Set the OPCA PCS Special Resources Policy.

      Use NO if you do not want any Special Resources Set.

      Use ALL if you want ALL SA z/OS Resources echoed as TWS Special Resources (this will create a large number of TWS special resources).

      Use YES if you want a selection of SA z/OS Resources echoed as TWS Special Resources.

   b. If you specified YES above, ensure that the TWS Special Resources policy item is updated with the appropriate resource masks and linked to the appropriate systems via the WHERE USED entry.
2. Ensure that the definitions for the TWS Status Observer are entered into the Automation Policy. The instructions for this are given in "Defining the SA z/OS Status Observer" on page 64.
3. Ensure that TWS option RESOPTS DYNAMICADD(YES) is specified.

© Copyright IBM Corp. 1990, 2010

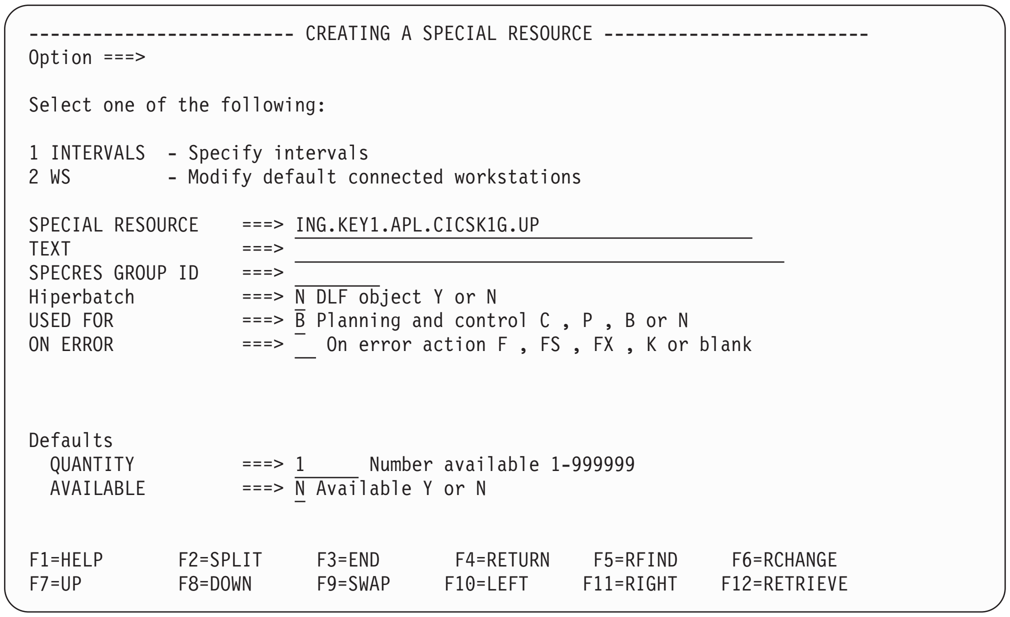
Using SA z/OS TWS Special Resources in an Application

Holding an Operation until an SA z/OS Resource Reaches a Desired State

To use a SA z/OS special resource to hold an operation until the appropriate status is achieved do the following:

1. Create the special resource in the TWS Database.

   Use the TWS ISPF dialog to create the special resource. Set the Availability of the special resource to N, as shown in Figure 38.



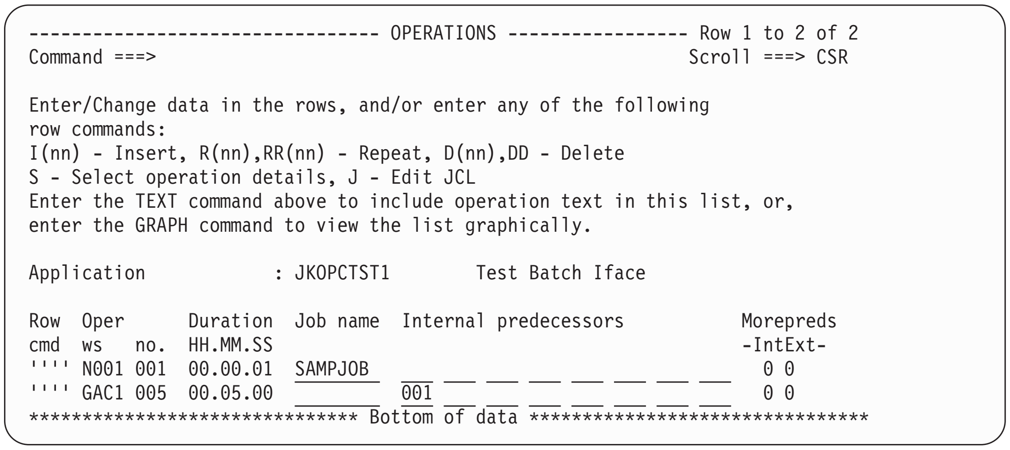
```
------------------------- CREATING A SPECIAL RESOURCE -------------------------
Option ===>

Select one of the following:

1 INTERVALS  - Specify intervals
2 WS         - Modify default connected workstations

SPECIAL RESOURCE    ===> ING.KEY1.APL.CICSK1G.UP
TEXT                ===>
SPECRES GROUP ID    ===>
Hiperbatch          ===> N DLF object Y or N
USED FOR            ===> B Planning and control C , P , B or N
ON ERROR            ===> __ On error action F , FS , FX , K or blank



Defaults
  QUANTITY          ===> 1     Number available 1-999999
  AVAILABLE         ===> N Available Y or N


F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

Figure 38. Creating a Special Resource

2. Create the Operations in the application:

   The first operation should submit the batch job to execute the requested function.

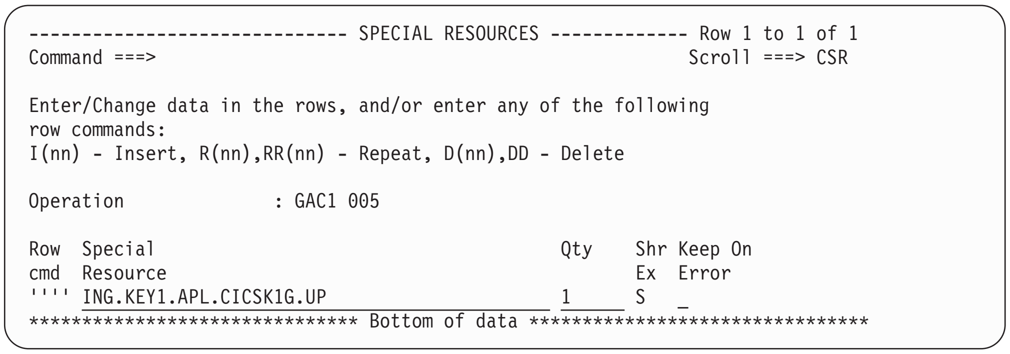
   The second operation runs at a General Completion Workstation and waits for the appropriate Special Resource, as shown in Figure 39.



```
--------------------------------- OPERATIONS ----------------- Row 1 to 2 of 2
Command ===>                                                  Scroll ===> CSR

Enter/Change data in the rows, and/or enter any of the following
row commands:
I(nn) - Insert, R(nn),RR(nn) - Repeat, D(nn),DD - Delete
S - Select operation details, J - Edit JCL
Enter the TEXT command above to include operation text in this list, or,
enter the GRAPH command to view the list graphically.

Application           : JKOPCTST1       Test Batch Iface

Row  Oper      Duration  Job name  Internal predecessors           Morepreds
cmd  ws   no.  HH.MM.SS                                            -IntExt-
'''' N001 001  00.00.01  SAMPJOB   ___ ___ ___ ___ ___ ___ ___ ___   0 0
'''' GAC1 005  00.05.00  ________  001 ___ ___ ___ ___ ___ ___ ___   0 0
******************************* Bottom of data ********************************
```

Figure 39. Creating the Operations

3. The Special resource for operation 005 is as shown in Figure 40.



```
------------------------------ SPECIAL RESOURCES ------------- Row 1 to 1 of 1
Command ===>                                                Scroll ===> CSR

Enter/Change data in the rows, and/or enter any of the following
row commands:
I(nn) - Insert, R(nn),RR(nn) - Repeat, D(nn),DD - Delete

Operation             : GAC1 005

Row  Special                                    Qty    Shr Keep On
cmd  Resource                                          Ex  Error
'''' ING.KEY1.APL.CICSK1G.UP____________________ 1_____ S   _
******************************* Bottom of data ********************************
```

Figure 40. Special Resource for Operation 005

4. Make sure you define the Special Resource to TWS with an initial Availability of N.

In this case Operation Number 005 will wait until SA z/OS sets the special resource ING.KEY1.APL.CICSK1G.UP to Y. This will only happen if the observed and desired status of CICSK1G/APL/KEY1 are both AVAILABLE.

Chapter 10. Automating Applications with TWS Automation

This chapter explains how to set up TWS Automation in TWS and in SA z/OS.

Defining Automated TWS Applications

This section describes what you need to do when you define an application in TWS that you can automate using TWS Automation.

Note: This section contains some specific details about how to define applications and other items to TWS. However, it does not explain basic TWS functions because it assumes that you have a prerequisite knowledge of TWS and will refer to the TWS documentation when necessary.

Defining Information for TWS Automation in TWS

The information that is passed to TWS Automation is entered in standard TWS description fields. This information is used to route requests where they are verified and executed. When the request is completed, a status change for the operation is sent back to TWS. The minimum information that needs transferring to accomplish this is listed in Table 7.

Table 7. TWS Automation Items Defined in TWS

Definition in TWS	Information item	Refer to
General reporting workstations that represent target NetView domains	TWS workstation ID representing the NetView domain ID	"Defining the Target NetView Domains" on page 75
TWS-defined application making requests to TWS Automation	Application name	Application field in Figure 42 on page 77
Job name of the SA z/OS subsystem that is associated with this operation	Job name	Job name field in Figure 42 on page 77
Operations executed within a job	Operation number or numbers	No. field in Figure 42 on page 77
Request and request parameters to be performed for the specific operation	A request, such as STOP, START, or CANCEL and optional parameters	Operation text field in Figure 43 on page 77 and Figure 44 on page 78
Command request to be executed by SA z/OS	Command text and completion information	"Defining a Command with an Automation Workstation" on page 86

Defining the Target NetView Domains

To define a TWS request that TWS Automation can use, a workstation representing the target NetView domain is required. This workstation, which is defined with TWS using the standard TWS dialogs, should be a general, automatic reporting workstation. Its name *must* have the format

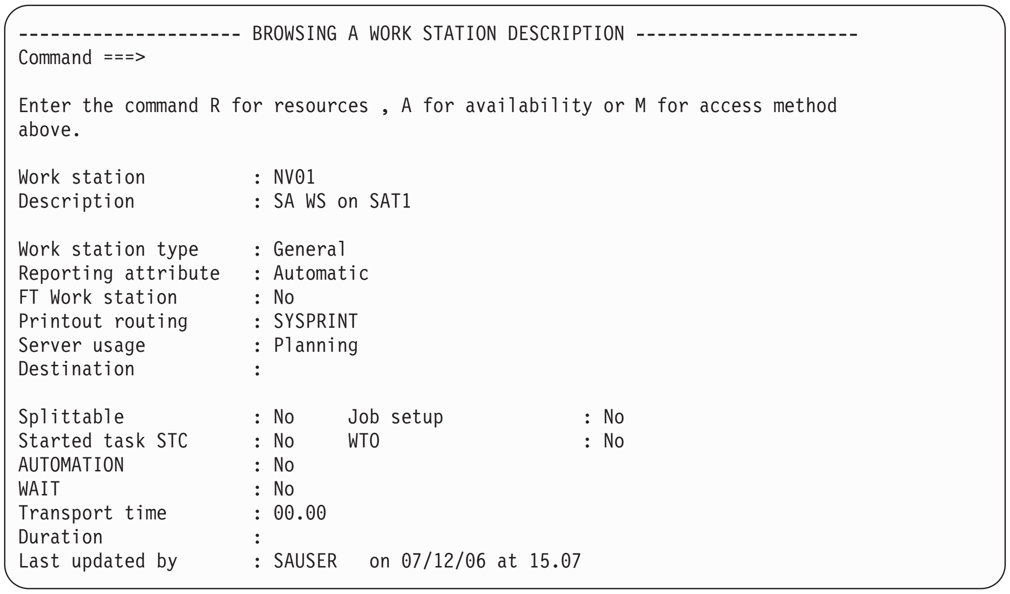
```
NVxx
```

© Copyright IBM Corp. 1990, 2010

Notes:

1. Reserve NV*xx* workstations for TWS Automation. Unpredictable and undesirable results may occur if these workstation names are used for other workstations.
2. If using *automation* workstations, the workstation destination name can contain the NetView domain ID. If it does not, the workstation name is mapped to the NetView domain ID.

Figure 41 shows a typical definition.



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

Enter the command R for resources , A for availability or M for access method
above.

Work station          : NV01
Description           : SA WS on SAT1

Work station type     : General
Reporting attribute   : Automatic
FT Work station       : No
Printout routing      : SYSPRINT
Server usage          : Planning
Destination           :

Splittable            : No     Job setup             : No
Started task STC      : No     WTO                   : No
AUTOMATION            : No
WAIT                  : No
Transport time        : 00.00
Duration              :
Last updated by       : SAUSER   on 07/12/06 at 15.07
```

Figure 41. Sample NVxx Workstation Definition in TWS

Entries in the SA z/OS policy database (WORKSTATION DOMAINS policy object, entry type ODM) translate NV*xx* to an actual NetView domain ID. The automation programmer defines these entries. In this manner, the scheduler defining the TWS applications does not need to know the NetView domain ID names, but rather works with the workstation representation of those names. This allows changes to the relationship of workstations to NetView domain IDs without modifying the TWS definitions.

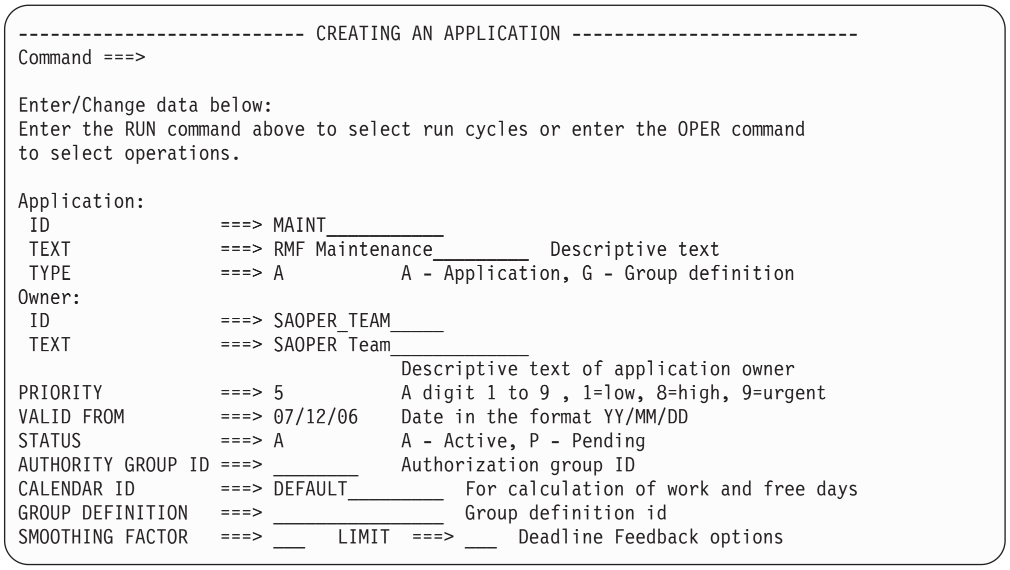
Note: You may use TWS database management dialogs or batch loader jobs to define the NV*xx* workstations.

Defining Applications for TWS Automation in TWS

Standard TWS application description panels are used to define applications that put requests to TWS Automation. The following items are defined:

- Application making the request
- Function requested

Application Making the Request: The application making the request is defined to TWS with operations specified on the NV*xx* workstation, as shown in Figure 42 on page 77.

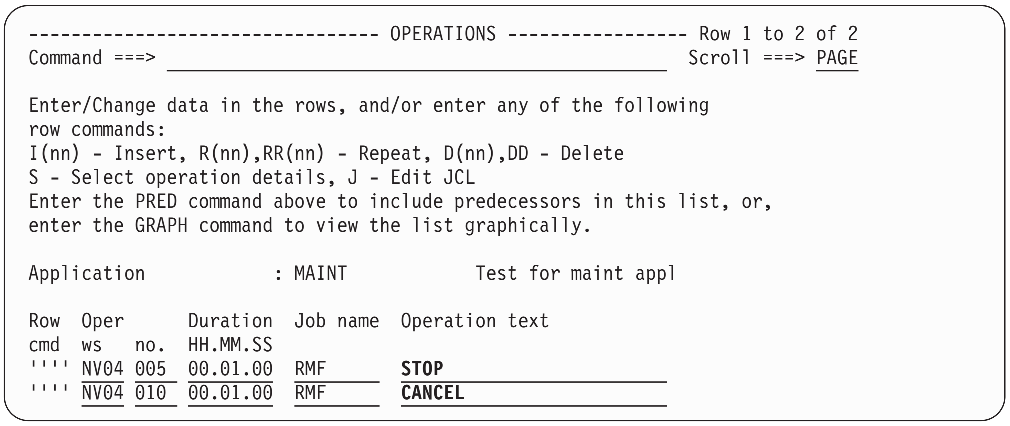


```
--------------------------- CREATING AN APPLICATION ---------------------------
Command ===>

Enter/Change data below:
Enter the RUN command above to select run cycles or enter the OPER command
to select operations.

Application:
 ID                ===> MAINT___________
 TEXT              ===> RMF Maintenance_________  Descriptive text
 TYPE              ===> A           A - Application, G - Group definition
Owner:
 ID                ===> SAOPER_TEAM_____
 TEXT              ===> SAOPER Team____________
                                    Descriptive text of application owner
PRIORITY           ===> 5           A digit 1 to 9 , 1=low, 8=high, 9=urgent
VALID FROM         ===> 07/12/06    Date in the format YY/MM/DD
STATUS             ===> A           A - Active, P - Pending
AUTHORITY GROUP ID ===> ________    Authorization group ID
CALENDAR ID        ===> DEFAULT_________  For calculation of work and free days
GROUP DEFINITION   ===> ________________  Group definition id
SMOOTHING FACTOR   ===> ___   LIMIT  ===> ___  Deadline Feedback options
```

Figure 42. Defining the MAINT Application in TWS

Function Requested: The function requested and the request parameters, if any, are not standard TWS definitions. Enter these fields in the **Operation text** field using blanks as delimiters.

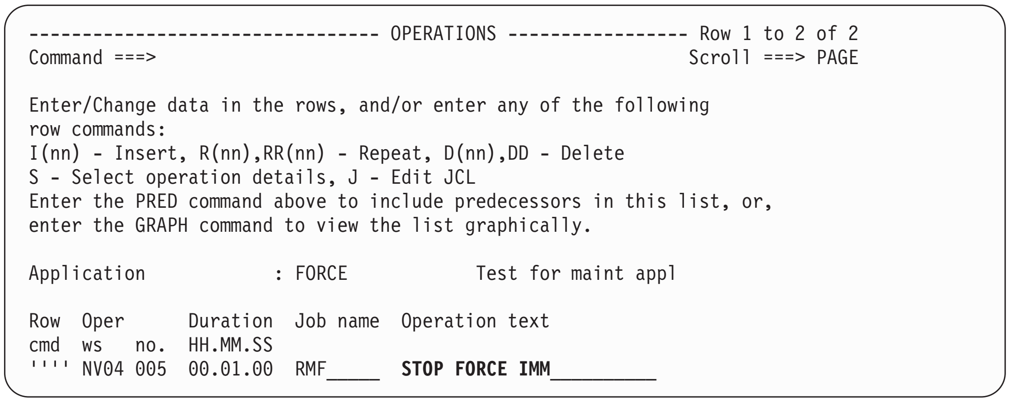
Figure 43 shows an example of how to define a request to stop and start RMF™.



```
--------------------------------- OPERATIONS ----------------- Row 1 to 2 of 2
Command ===> _______________________________________________ Scroll ===> PAGE

Enter/Change data in the rows, and/or enter any of the following
row commands:
I(nn) - Insert, R(nn),RR(nn) - Repeat, D(nn),DD - Delete
S - Select operation details, J - Edit JCL
Enter the PRED command above to include predecessors in this list, or,
enter the GRAPH command to view the list graphically.

Application            : MAINT           Test for maint appl

Row  Oper      Duration  Job name  Operation text
cmd  ws   no.  HH.MM.SS
'''' NV04 005  00.01.00  RMF       STOP
'''' NV04 010  00.01.00  RMF       CANCEL
```

Figure 43. TWS Operations Panel Showing TWS Automation Requests

TWS Automation permits the inclusion of two optional parameters in the request buffer. The target system builds the required command with these parameters, using information contained in the SA z/OS policy database. Alternatively, the parameters pass control information to optional user-written modules. Figure 44 on page 78 shows an example of the request using these optional parameters.

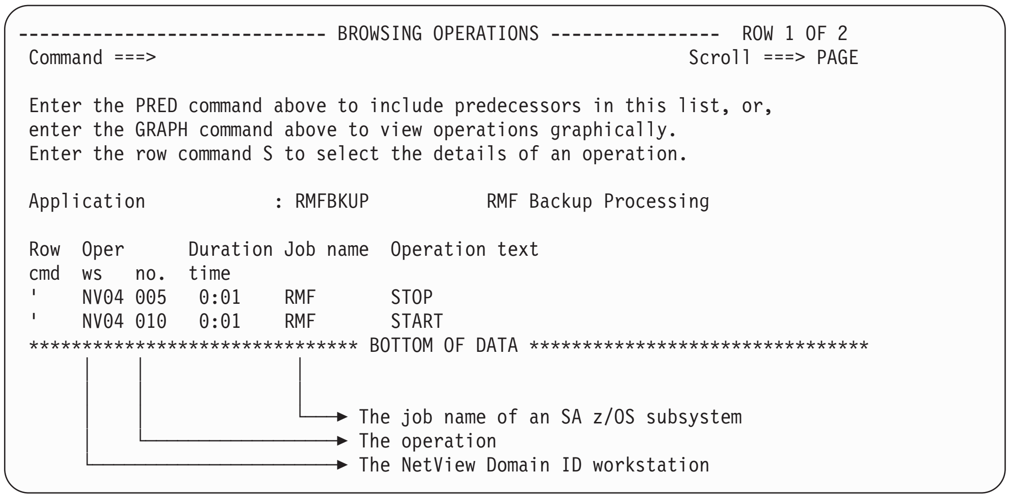


```
--------------------------------- OPERATIONS ----------------- Row 1 to 2 of 2
Command ===>                                              Scroll ===> PAGE

Enter/Change data in the rows, and/or enter any of the following
row commands:
I(nn) - Insert, R(nn),RR(nn) - Repeat, D(nn),DD - Delete
S - Select operation details, J - Edit JCL
Enter the PRED command above to include predecessors in this list, or,
enter the GRAPH command to view the list graphically.

Application           : FORCE           Test for maint appl

Row  Oper      Duration  Job name  Operation text
cmd  ws   no.  HH.MM.SS
'''' NV04 005  00.01.00  RMF_____  STOP FORCE IMM__________
```

Figure 44. Request Using Optional Parameters

Displaying TWS Automation Requests in TWS

Because TWS Automation requests are stored as operation text, you can view them in TWS, as shown in Figure 45.



```
----------------------------- BROWSING OPERATIONS ----------------  ROW 1 OF 2
Command ===>                                              Scroll ===> PAGE

Enter the PRED command above to include predecessors in this list, or,
enter the GRAPH command above to view operations graphically.
Enter the row command S to select the details of an operation.

Application           : RMFBKUP          RMF Backup Processing

Row  Oper      Duration Job name  Operation text
cmd  ws   no.  time
'    NV04 005   0:01    RMF       STOP
'    NV04 010   0:01    RMF       START
******************************* BOTTOM OF DATA ********************************
```

The job name of an SA z/OS subsystem
The operation
The NetView Domain ID workstation

Figure 45. Browsing Operations Including TWS Automation Requests

The following list defines several of the fields that are shown on the panel in Figure 45.

NV04	Represents the target NetView domain or the sysplex that the request is sent to.
RMFBKUP	The application submitting the request to TWS Automation.
RMF	Target subsystem. This looks like a job to TWS, but is actually a subsystem.
005, 010	Standard operation sequence numbers used by TWS.
STOP, START	Requested function. There are no parameters in this example.

Example of an Application Making a Request

This section provides an example of how an application that puts a request to TWS Automation is defined in TWS.

The application name is MAINT. This application consists of three operations:

- Stop RMF on the target system
- Schedule a batch job
- Restart RMF on successful completion of the batch job

Figure 46 shows the initial panel in the application creation process.

```
--------------------------- CREATING AN APPLICATION ---------------------------
Command ===> oper

Enter/Change data below:
Enter the RUN command above to select run cycles or enter the OPER command
to select operations.

Application:
 ID                ===> MAINT___________
 TEXT              ===> RMF Maintenance_________  Descriptive text
 TYPE              ===> A           A - Application, G - Group definition
Owner:
 ID                ===> SAOPER_TEAM_____
 TEXT              ===> SAOPER Team____________
                                    Descriptive text of application owner
PRIORITY           ===> 5           A digit 1 to 9 , 1=low, 8=high, 9=urgent
VALID FROM         ===> 07/12/06    Date in the format YY/MM/DD
STATUS             ===> A           A - Active, P - Pending
AUTHORITY GROUP ID ===> ________    Authorization group ID
CALENDAR ID        ===> DEFAULT_________  For calculation of work and free days
GROUP DEFINITION   ===> ________________  Group definition id
SMOOTHING FACTOR   ===> ___    LIMIT  ===> ___  Deadline Feedback options
```

Figure 46. RMF Maintenance Application Primary Panel in TWS

In Figure 46, certain fields, such as calendar ID, are not used. However, TWS Automation does not preclude the use of normal application and operation functions.

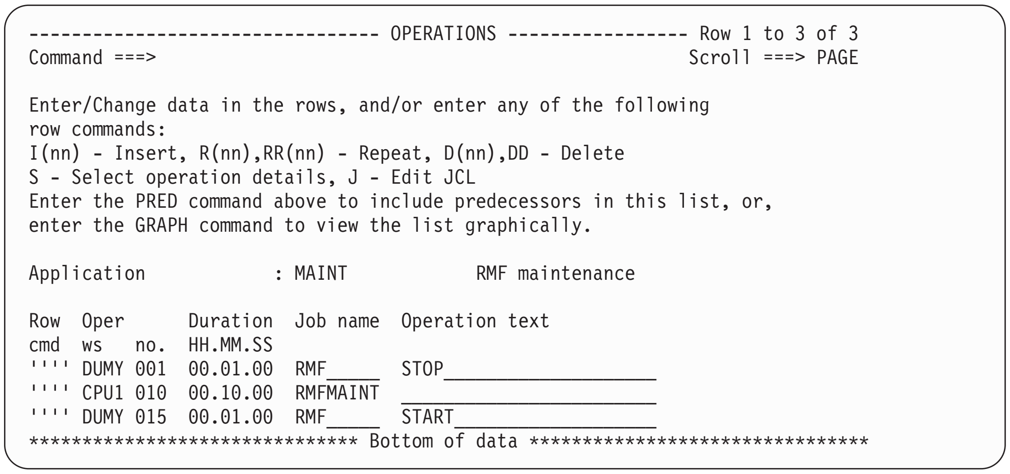
Selecting OPER as a primary command allows the entry of individual operations for this application.

In Figure 47, three operations are defined. The first and third send requests to TWS Automation in the NetView domain that is associated with the NV00 workstation. The second is a batch job named RMFMAINT that performs the batch maintenance tasks.

```
-------------------------------- OPERATIONS ----------------- Row 1 to 3 of 3
Command ===> text                                             Scroll ===> PAGE

Enter/Change data in the rows, and/or enter any of the following
row commands:
I(nn) - Insert, R(nn),RR(nn) - Repeat, D(nn),DD - Delete
S - Select operation details, J - Edit JCL
Enter the TEXT command above to include operation text in this list, or,
enter the GRAPH command to view the list graphically.

Application           : MAINT            RMF maintenance

Row  Oper      Duration  Job name  Internal predecessors          Morepreds
cmd  ws   no.  HH.MM.SS                                           -IntExt-
'''' NV00 001  00.01.00  RMF_____  ___ ___ ___ ___ ___ ___ ___ ___   0  1
'''' CPU1 010  00.10.00  RMFMAINT  005 ___ ___ ___ ___ ___ ___ ___   0  0
'''' NV00 015  00.01.00  RMF_____  010 ___ ___ ___ ___ ___ ___ ___   0  0
****************************** Bottom of data *******************************
```

Figure 47. Operations in the MAINT Application

Selecting TEXT as a primary command allows entry of the operation text. TWS Automation uses the **Operation text** field to contain the request and up to two optional parameters for operations with the workstation defined for TWS Automation. Figure 48 shows the resulting operations text detail panel.



```
--------------------------------- OPERATIONS ----------------- Row 1 to 3 of 3
Command ===>                                                  Scroll ===> PAGE

Enter/Change data in the rows, and/or enter any of the following
row commands:
I(nn) - Insert, R(nn),RR(nn) - Repeat, D(nn),DD - Delete
S - Select operation details, J - Edit JCL
Enter the PRED command above to include predecessors in this list, or,
enter the GRAPH command to view the list graphically.

Application            : MAINT            RMF maintenance

Row  Oper      Duration  Job name  Operation text
cmd  ws   no.  HH.MM.SS
'''' DUMY 001  00.01.00  RMF_____  STOP____________________
'''' CPU1 010  00.10.00  RMFMAINT  ________________________
'''' DUMY 015  00.01.00  RMF_____  START___________________
****************************** Bottom of data ********************************
```

Figure 48. Operations Text Detail Panel

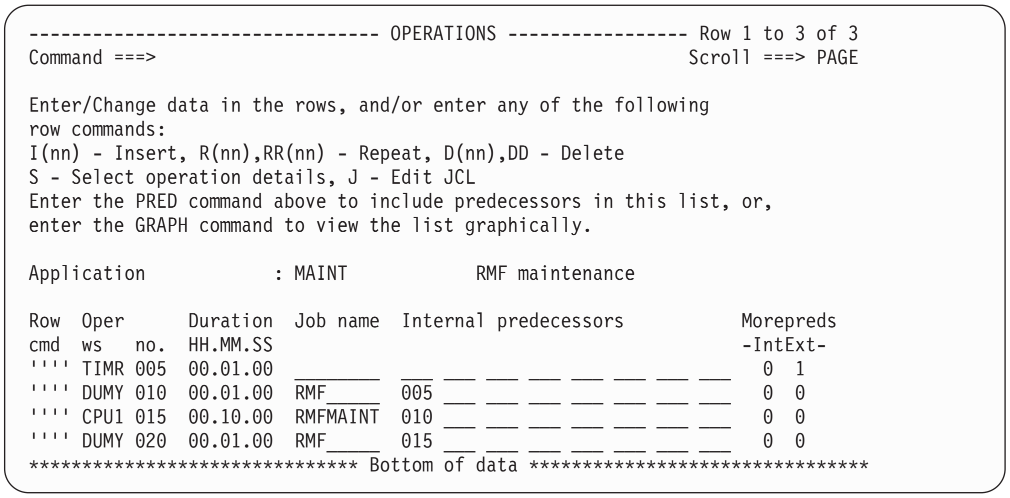
In the applications, TWS Automation defines TWS requests in a generic manner. Figure 48 shows the MAINT application with the first and last operations defined for the NetView workstation NV00. The requests that are forwarded to the NetView workstation NV00 are STOP and START. These requests are expanded by definitions in the policy database into commands; see “Executing TWS Requests with TWS Automation” on page 82.

Handling Time Dependencies

If you require a time dependency, do not place the time consideration on the NV*xx* defined operation because the status change drives the TWS user exit EQQUX007, regardless of the timer status. For a general workstation, such as those defined for TWS Automation, this occurs when all dependencies are fulfilled except the time consideration.

To avoid this problem, define a dummy, non-reporting workstation. Place the timer dependency on this dummy workstation. Define any dependencies on the dummy workstation, which is the predecessor to the NV*xx* workstation. Once you satisfy all other dependencies and complete the time dependency, the dummy timer workstation completes immediately and starts the operation on the NV*nn* workstation.

As an example, redefine the MAINT application shown in Figure 46 on page 79, and Figures 47 and 48 on page 80, with a timer dummy workstation (TIMR) as the first operation of the application. The panel in Figure 49 on page 81 shows this new definition.



```
--------------------------------- OPERATIONS ----------------- Row 1 to 3 of 3
Command ===>                                                  Scroll ===> PAGE

Enter/Change data in the rows, and/or enter any of the following
row commands:
I(nn) - Insert, R(nn),RR(nn) - Repeat, D(nn),DD - Delete
S - Select operation details, J - Edit JCL
Enter the PRED command above to include predecessors in this list, or,
enter the GRAPH command to view the list graphically.

Application           : MAINT           RMF maintenance

Row  Oper      Duration  Job name  Internal predecessors          Morepreds
cmd  ws   no.  HH.MM.SS                                           -IntExt-
'''' TIMR 005  00.01.00  ________  ___ ___ ___ ___ ___ ___ ___ ___   0  1
'''' DUMY 010  00.01.00  RMF_____  005 ___ ___ ___ ___ ___ ___ ___   0  0
'''' CPU1 015  00.10.00  RMFMAINT  010 ___ ___ ___ ___ ___ ___ ___   0  0
'''' DUMY 020  00.01.00  RMF_____  015 ___ ___ ___ ___ ___ ___ ___   0  0
****************************** Bottom of data *******************************
```

Figure 49. Using Time as a Dependency

With this type of structure, TWS Automation can schedule the NV*xx* operation, rather than the timer-dependent dummy workstation, if the application needs scheduling on demand or restarting. This manually initiated procedure is independent of the time consideration, if appropriate.

Changes to the Status of the Operation

The operation with the NV*xx* workstation goes through several status changes as the request defined in the operator text is processed. The initial trigger is one of three status changes. When the operation moves to the A (arrival), R (ready), or * (ready with non-reporting predecessor) status, the TWS Automation function in the EQQUX007 exit is triggered. The exit then examines the request. If the request is valid, the exit transfers it to the target NetView.

If any definition problems are determined, TWS Automation updates the status to E with an error code of U*xxx*. See *IBM Tivoli System Automation for z/OS Messages and Codes*. TWS Automation takes no further action. The user is then responsible for correcting the error and restarting the application at the failed operation.

If TWS Automation encounters a connectivity problem, it marks the operation with a status of E (error) and an error code of S*xxx*. If this happens, TWS Automation automatically restarts the operation once the connectivity problem is resolved. However, if the operation status is changed manually, the automatic restart is suppressed.

After TWS Automation resolves the request and verifies it, the status is updated to S (started) before TWS Automation submits it. Once the request is submitted and action is requested, TWS Automation updates the status to C (completed).

If the desired result did not occur within the time period specified, the operation ends with an E status and a U*xxx* code, indicating that user intervention is required.

Extending the Daily Plan

TWS Automation does not call EQQUX007 for time-delay operations added at daily planning. To provide time-delay operations added at daily planning, you need to define an operation on a dummy workstation as a predecessor to the NV*xx* workstation. The operation on that workstation completes immediately after the daily plan is extended, and the operation on the NetView workstation is READY when all of its other dependencies are satisfied. See “Handling Time Dependencies” on page 80 for more information and an example of this technique.

Defining an operation on a dummy workstation is required because the operation-status-change exit is called whenever an operation in the current plan changes status. That exit is also called when a new operation has been added to the current plan by a function other than daily planning jobs, for example, by PIF or by the MCP dialog. The exit is called when the operation is added either to an existing occurrence or as a result of a new occurrence being added to the current plan.

Sending a Request to Optional Installation-Provided Functions

TWS Automation allows installation-specific extensions through optional user-provided modules. Two types of functions are supported:

- Issuing a non-SA z/OS command or request
- Sending a request through to TWS Automation to an installation extension

Because these user-provided modules are unique to your installation, you need to obtain information from your systems programmer/analyst on the use and syntax of these functions.

Executing TWS Requests with TWS Automation

Generally, you must use the TWS-specific OPCA, OPCACMD, and (optionally) OPCAPARM keywords to put an application defined to SA z/OS under the control of TWS Automation. You must define these entries under the MESSAGES/USER DATA policy item of the respective application in the SA z/OS policy database. See *IBM Tivoli System Automation for z/OS Defining Automation Policy* for more information on the MESSAGES/USER DATA policy item, and Chapter 11, “MESSAGES/USER DATA Entries and USER E-T Pairs for TWS Automation,” on page 91 for the TWS-specific keywords.

You can vary the amount of fine-tuning considerably. If you only want to start and stop subsystems known to SA z/OS through TWS in a standard way, you need not even code any of these keywords. On the other hand, you can call user-written modules with OPCACMD that perform tasks not related to any SA z/OS subsystem and that must inform TWS about the success of their execution independently of TWS Automation.

The following sections explain the connection between the TWS request and the TWS-specific MESSAGES/USER DATA keywords by a fairly typical example, describe the use of request parameters, and give an overview of the different request types.

Request Types

TWS requests can be classified into four types depending on the existence or non-existence of TWS-specific keywords and on the type of command associated with the OPCACMD keyword. The following sections give an overview of these types.

Starting and Stopping Subsystems without TWS-Related Keywords

For START and STOP requests, you need not code any TWS-specific MESSAGES/USER DATA keyword in the policy database. TWS Automation will issue a default command when it detects that no OPCA entry exists for the START or STOP request in the MESSAGES/USER DATA policy item of the subsystem to be started or stopped. In these cases you can specify a valid startup, respectively, shutdown type. The valid startup types are NORM and any startup type that has been defined in the STARTUP policy item of the subsystem to be started. The valid shutdown types are NORM, IMMED, or FORCE.

The start command issued by TWS Automation will be as follows:

```
INGREQ subsystem_name REQ=START,OUTMODE=LINE,SOURCE=EXTERNAL,VERIFY=NO
```

If you specify a startup type, this type will be added to the command.

The format of the stop command issued by TWS Automation is:

```
INGREQ subsystem_name REQ=STOP,OUTMODE=LINE,SOURCE=EXTERNAL,VERIFY=NO
```

If you specify a shutdown type, this type will be added to the command.

Where specified, the first parameter is a token that is taken as the type.

Note that if you want to add or change any parameter other than the TYPE parameter, you must specify your INGREQ command in an OPCACMD entry and code the associated OPCA entry. For more information on INGREQ, see *IBM Tivoli System Automation for z/OS Operator's Commands*.

Also note that when a default start or stop is selected, SA z/OS does *not* set a start or stop delay timer to ensure that the request finishes within a specified time. If a start or stop delay timer is required, the OPCA and OPCACMD entries should be coded.

When the startup or shutdown type is invalid for the subsystem in question, or when the command encounters any problem, the operation will fail with a user abend code of U003.

Canceling a Start or Stop Request

In many cases after a subsystem has been started or stopped, the previous request to SA z/OS needs to be removed to allow normal automation actions to continue. This can be achieved by issuing a CANCEL request type without parameters. Internally an INGSET CANCEL command will be issued to remove the previous START or STOP request for the resource.

Requests Using SA z/OS Automation Functions

These are requests that an OPCACMD entry is coded for, and where the command specified in that entry changes the automation status of the subsystem to UP, RUNNING or AUTODOWN. This can be an SA z/OS base command (for example, INGREQ), but also a user-written command that calls INGREQ. The name of the request must not begin with 'UX'. When the request contains parameters, these are stored in the &EHKVAR1 and &EHKVAR2 variables, respectively, and you can pass them to the command by incorporating these variables into the command text.

For every OPCACMD entry you must code an associated OPCA entry. An OPCAPARM entry is not required.

For START and STOP requests, you may want to use this type instead of coding no OPCACMD entry at all, if you want to override the default values for certain INGREQ parameters.

Subsystem Related Requests not Using SA z/OS Automation Functions

These are requests that an OPCACMD entry is coded for, and where the command specified does not trigger a status change of the subsystem that it relates to. The name of the request must not begin with 'UX'. When the request contains parameters, these are stored in the &EHKVAR1 and &EHKVAR2 variables, respectively, and you can pass them to the command via these variables.

For every OPCACMD entry with such a command you must code an associated OPCA entry. You must also define a corresponding OPCAPARM entry. This entry must specify a user-written timer module that informs TWS whether or not the request was successful (by calling OPCACOMP); this module is called by TWS Automation after the timer defined in the OPCA entry has expired.

For more information on this request type, see "User Functions Related to an SA z/OS-Defined Subsystem" on page 123.

Non-Subsystem Requests

These are requests that an OPCACMD entry is coded for, and where the command specified in that entry does not relate to a subsystem known to SA z/OS. The name of these requests must begin with 'UX'. The OPCACMD entry for a non-subsystem request must be coded as a USER E-T pair. With requests of this type, the complete request buffer will be stored in &EHKVAR1, and it is up to the user-written command to analyze this information. &EHKVAR2 contains the input arrival time.

For the format of the request buffer for 'UX' requests, see Table 9 on page 120.

No OPCA or OPCAPARM entry is needed for non-subsystem requests. The responsibility for informing TWS about the success of the operation lies entirely with the user-written command.

For more information on this request type, see "Non-Subsystem Operations" on page 127.

TWS Requests and MESSAGES/USER DATA Keywords

You put a request to TWS Automation by specifying the requested function and (optionally) one or two parameters in the **Operation text** field of the **Operations** panel for a TWS application (for details, see "Defining Applications for TWS Automation in TWS" on page 76). For example:

```
Row  Oper      Duration Job name  Operation text
cmd  ws   no.  HH.MM
'''' NV04 005   0.01    RMF_____  START__________________
'''' NV04 010   0.01    CICS1H__  START COLD_____________
```

Figure 50. OPC/ESA Operations Panel

The request is START in both entries. The second entry has one parameter, namely COLD.

OPCACMD Keyword

TWS Automation uses the **Job name** and the **Operation text** fields of the TWS operation to translate requests into commands. After identifying the subsystem through the **Job name** entry, it consults the OPCACMD entry in the MESSAGES/USER DATA policy item of this subsystem. The CMD attributes of this entry contain the commands that are to be issued in response to various requests, or combinations of request and parameters that can be specified in the **Operation text** field. The following panel gives an example entry for RMF:

```
Pass/Selection Automated Function/'*'
Command Text
START
INGREQ RMF REQ=START,TYPE=NORM,SOURCE=EXTERNAL


OPMSG
MSG ALL RMF MSG

```

Figure 51. Specifying the Command for a Request

These entries specify in the **Command Text** field the commands that are to be issued for RMF when START, respectively, OPMSG requests are put to TWS Automation. Thus, when the 005 request of Figure 50 on page 84 has been put to TWS Automation, TWS Automation will issue the command

```
INGREQ RMF REQ=START,TYPE=NORM,SOURCE=EXTERNAL
```

OPCA Keyword

Besides specifying the command to be issued in response to the request, you must also tell TWS Automation:

- The status you expect the subsystem to assume as a result of this command
- The time interval within which the subsystem must assume this status

This is done with the OPCA keyword. The following panel continues the example of Figure 51.

```
Code 1          Code 2          Code 3          Value Returned
START                                           UP,2,RMFUTMER
OPMSG                                           UP,1,
STOP                                            DOWN,2,
```

Figure 52. Specifying Expected Status and Time Interval

Here, the request is specified in the **Code 1** column. The **Value Returned** column contains the expected status, the time interval (in minutes), and optionally a timer name. The **Code 2** and **Code 3** columns are intended for eventual request parameters and consequently left blank in this example.

Flow of Control

By the entries of Figure 51 and Figure 52, TWS Automation will execute the start request of Figure 50 on page 84 for RMF as follows:

1. It sets the RMFUTMER timer to two minutes.
2. It issues the command `INGREQ RMF REQ=START,TYPE=NORM,SOURCE=EXTERNAL`.
3. If RMF has assumed the UP state before two minutes have passed, TWS Automation cancels the timer and posts the completion code C (for COMPLETED) back to TWS.
4. After the timer has expired, TWS Automation checks the status of RMF. If RMF is in the UP status, TWS Automation posts C to TWS; otherwise, it posts E (for error) and an additional error code.

Request Parameters and the &EHKVAR*n* Variables

Besides the request itself, the TWS request can contain one or two parameters. You can use this additional information to associate different commands with different variants of the same request; as an example, consider the different startup types for CICS®. You can pass the parameters to your command through the task global &EHKVAR1 and &EHKVAR2 variables.

As an example, suppose you want to specify the startup type for a CICS application in a TWS start request. Then you enter it as a request parameter in the **Operation text** field (see the 010 request in Figure 50 on page 84) and pass the startup type to the command specified in the OPCACMD entry by incorporating the &EHKVAR1 variable in the command text. In the following example, this command is not an SA z/OS command, but a user-written CLIST named MYCLIST that uses the startup information to apply the desired startup method.

```
Pass/Selection Automated Function/'*'
Command Text
START
MYCLIST CICS1H &EHKVAR1
```

Figure 53. Specifying a Command that Requires Parameter Information

Then, when the request of the 010 operation in Figure 50 on page 84 is put to TWS Automation, MYCLIST will be called with the arguments CICS1H and COLD. A very simple version of MYCLIST could be as follows:

```
/* MYCLIST SAMPLE */
PARSE UPPER ARG CICSNAME STARTTYPE

IF STARTTYPE='COLD' THEN
   "INGREQ "||CICSNAME||" REQ=START,TYPE=COLD,OUTMODE=LINE"
ELSE
   "INGREQ "||CICSNAME||" REQ=START,TYPE=AUTO,OUTMODE=LINE"
EXIT 0
```

If you use parameters, you must code an OPCA entry for every parameter (combination). For the example of Figure 53, the OPCA entry could look like Figure 54.

```
Code 1         Code 2         Code 3         Value Returned
START          COLD                          UP,10,CICS1TMR
START          AUTO                          UP,5,CICS1TMR
```

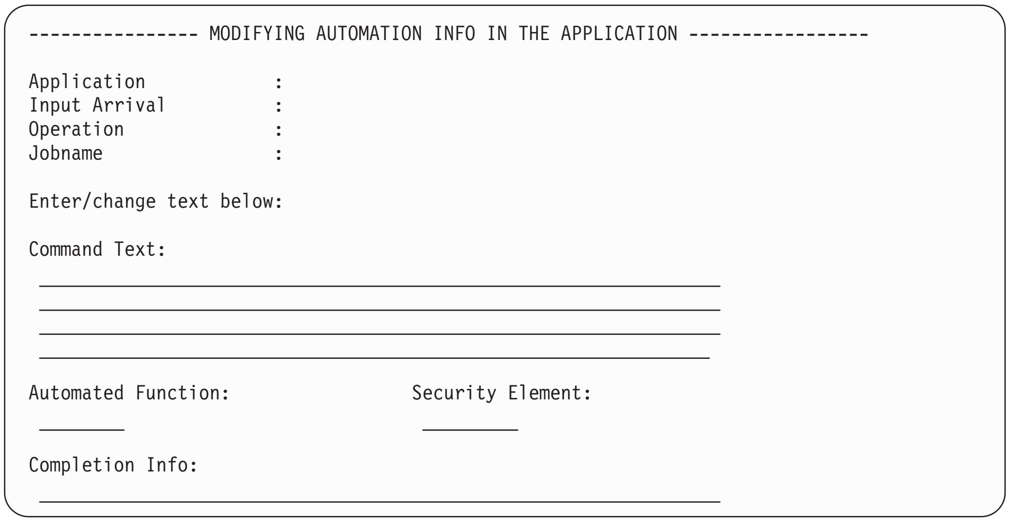
Figure 54. Specifying Expected Status and Time Interval for Different Request Parameters

Defining a Command with an Automation Workstation

The TWS application description panels allow the TWS user to specify a free-format SA z/OS or NetView command (of up to 255 characters) that is passed to SA z/OS for execution. The command must be pipeable.

The workstation destination can optionally be defined. It is the NetView domain where the command should be executed.

This then leads to a panel similar to Figure 55 on page 87.



```
---------------- MODIFYING AUTOMATION INFO IN THE APPLICATION -----------------

Application           :
Input Arrival         :
Operation             :
Jobname               :

Enter/change text below:

Command Text:
 ________________________________________________________________
 ________________________________________________________________
 ________________________________________________________________
 _______________________________________________________________

Automated Function:               Security Element:
 ________                          _________

Completion Info:
 ________________________________________________________________
```

Figure 55. TWS Operations Interface Panel to Define Commands

Use this panel to enter the following:

Command Text
The complete free-format command. This can also contain TWS variables: Before the operation is passed to SA z/OS, TWS performs variable substitution processing.

| **Note:** MVS commands *must* be issued within a NetView PIPE command to
| ensure that the command output is correlated.

Automated Function
The name of the automated function (optional). If specified, it causes the specified command to be executed on the NetView task that is assigned to the automated operator. This allows an installation to serialize the execution of commands—they are executed sequentially in the order they arrive in SA z/OS. If omitted, the command is processed by one of the tasks that is designated for handling the command received via the TWS request interface.

Security Element
This name is passed to the AOFEXC20 exit so that the installation can perform any security-related checking.

Completion Info
This includes:

- The maximum wait time in NetView notation (mm, :ss, mm:ss, hh:mm:ss). If specified, it causes SA z/OS to wait for the completion of the command for the specified time interval. If the command does not complete within the specified time interval, SA z/OS posts the operation in error.

  For an INGREQ or INGMOVE command the following applies:

  – The command is considered to be complete when the specified resource has reached or is already in the requested state.
  – If more than one resource is specified in the INGREQ or INGMOVE command, all resources must be in the requested state before the command is considered complete.

- The timer value that is specified should be large enough to accommodate the longest interval of time that the requested function may take under normal operating conditions.
- If the wait time is omitted, the installation default value is used. This default is set using the AOF_AAO_TWS_MAX_WAIT_TIME advanced automation option. If neither of these values is set, the SA z/OS default of 5 minutes is used.

• The maximum return code that is acceptable. The default is 0.

• The name of the user-supplied completion checking routine (optional). If specified, the completion checking routine is responsible for ensuring that the command achieved the expected results before posting the operation as complete. This allows a TWS user to perform commands that are independent of an SA z/OS-controlled resource, for example the activation of the VTAM® major node.

For example, `05,4,MYRTN` sets a maximum wait time of 5 minutes, a maximum tolerated return code of 4 and the completion checking routine is MYRTN. The parameters must be separated by a comma.

Invoking the Command with SA z/OS

The following task global variables are set when SA z/OS issues a command with the Command Request interface:

EHKVAR1	Application description name
EHKVAR2	Input arrival time
EHKVAR3	Workstation name
EHKVAR4	Operation number
EHKVAR5	Job name
EHKVAR6	Maximum wait time, if present
EHKVAR7	Maximum return code
EHKVAR8	Name of checking routine

Completion Checking Routine

When invoking the completion checking routine task globals are set that hold the command that has been issued by SA z/OS on behalf of the TWS operation. The following parameters are passed to the routine:

1. The application description ID (ADID)
2. The workstation name
3. The operation number (OPNUM)
4. The input arrival timestamp in the form YYMMDDHHMM
5. The timer interval (maximum wait time) in hh:mm:ss (NetView notation)
6. The maximum return code as specified in the completion indicators of the operation specification

The following task globals are set:

EHKVAR1	The command that is issued.
EHKVAR2	The job name that is associated with command, if any.

The operation is considered to be in error when the return code is greater than zero. The following return codes are issued:

0 Successful completion of the command.

>0 Unsuccessful completion of the command.

Chapter 11. MESSAGES/USER DATA Entries and USER E-T Pairs for TWS Automation

Because TWS Automation is integrated into SA z/OS, you must enter any information for TWS Automation in the policy database via the customization dialogs. In most cases the customization dialogs precisely determine the format that this information must be entered in. There are, however, some TWS application-specific automation parameters that must or can only be specified as entries in the MESSAGES/USER DATA policy items of the respective application, or as USER E-T pairs. For general information on the MESSAGES/USER DATA policy item and USER E-T pairs, see *IBM Tivoli System Automation for z/OS Defining Automation Policy*. In these cases, the customization panels provide no information about the keywords and the format of their parameters.

The following chapter contains detailed descriptions of these automation entries. Note, however, that a general understanding of the MESSAGES/USER DATA policy item will be assumed.

The OPCACMD entry must be coded in a USER E-T pair if the command to be specified is *not* related to a subsystem known to SA z/OS.

TWS-Specific MESSAGES/USER DATA Keywords

The following keywords are specific for TWS Automation.

Entry	Description
"OPCA" on page 92.	Use this to define the expected state of the subsystem and the time interval within which this state must have been reached.
"OPCACMD" on page 94.	Use this to specify the command to be issued in response to a TWS request.
"OPCAPARM" on page 96.	Use this to specify modifications of eventual request parameters and a timer module for user-written commands.

OPCA

Purpose

With the OPCA entry, you define the state that is the expected result of a request (with or without parameters), and the time interval within which this state must have been reached. The OPCA entry is defined in the MESSAGES/USER DATA policy item of the subsystem that is to be put under control of TWS.

Format

Code 1	Code 2	Code 3	Value Returned
START			UP,3,RMFUTMER
STOP			DOWN,2,RMFDTMER

Parameters

Code 1
Request specified in the TWS operation text.

Code 2
Parameter 1 as specified in the TWS operation text.

Code 3
Parameter 2 as specified in the TWS operation text.

Value Returned

expstatus
Expected status of the subsystem at the completion of the request. *expstatus* stands for one of the following values: UP, RUNNING or DOWN.

Note: For backward compatibility CTLDOWN and AUTODOWN are also allowed for this entry and will be treated the same as DOWN.

timerint
Timer interval in minutes. The maximum value permitted is 1439 (23 hours and 59 minutes).

Set a timer interval that is long enough for the operation to complete reasonably. If the operation does not complete in the interval specified, then an error is posted to TWS.

timerid
Timer ID — from 1 to 8 characters.

This must be a valid NetView timer ID with a value not equal to `ALL` or beginning with `SYS`, `ING`, or `AOF`. This field is optional.

Usage Notes

For every OPCACMD entry there must be a corresponding OPCA entry.

Example

Code 1	Code 2	Code 3	Value Returned
START	AUTO		UP,5,CICS1TMR
START	COLD		UP,10,CICS1TMR

This example shows two entries, one for an automatic start, and one for a cold start. The AUTO or COLD parameter is added to the START request in TWS to indicate which one of several startup procedures is to be used. Presumably the two operations will take differing amounts of time, so the timer intervals are different.

This OPCA code entry is used in conjunction with a user-written CLIST, specified in the OPCACMD entry; see “Example 1” on page 94 for details of that entry and the sample CLIST.

OPCACMD

Purpose

With the OPCACMD entry, you define the command that is executed in response to a request (with or without parameters). Except for non-subsystem commands, there must be a corresponding OPCA entry for every OPCACMD entry.

Format

```
Pass/Selection Automated Function/'*'
Command Text
START
INGREQ RMF REQ=START,VERIFY=NO,SOURCE=EXTERNAL,TYPE=NORM,OUTMODE=LINE

STOP
INGREQ RMF REQ=STOP,VERIFY=NO,SOURCE=EXTERNAL,TYPE=NORM,OUTMODE=LINE
```

Parameters

Pass/Selection
Request specified in the TWS operation text.

Command Text
The actual command to be executed.

Usage Notes

This entry is necessary for all request types except START, STOP, and CANCEL requests. If no entries are supplied for START, STOP and CANCEL, the actions taken are detailed in “Starting and Stopping Subsystems without TWS-Related Keywords” on page 82 and “Canceling a Start or Stop Request” on page 83. The place where the OPCACMD entry is defined depends on the request type. When the request is related to a subsystem, it is defined in the MESSAGES/USER DATA policy item of the respective application. When the request is a non-subsystem request (see “Non-Subsystem Operations” on page 127), the OPCACMD entry must be entered in a USER E-T PAIRS entry.

Use SA z/OS commands to shut down and start up subsystems. This avoids the problem of having to determine the specific commands required for each subsystem.

Example 1

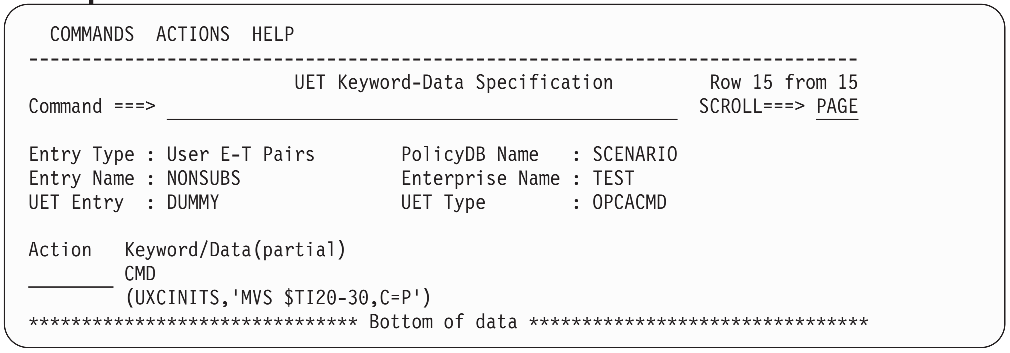
```
Pass/Selection Automated Function/'*'
Command Text
START
MYCLIST CICS1 &EHKVAR1

RECYCLE
INGREQ TESTAPPL REQ=STOP RESTART=YES SOURCE=EXTERNAL SCOPE=ALL
```

This example assumes that AUTO or COLD is added as a parameter to the START request in TWS to indicate which one of several startup procedures is to be used. The command specified in the **Command Text** field is a user-written CLIST. This CLIST is passed the parameter value (AUTO or COLD) in the &EHKVAR1 variable.

It also shows a RECYCLE type of operation (that is, bringing the subsystem down and restarting it immediately) being defined. The command is issued in linemode and verifies that it is accepted otherwise it posts the operation in error.

The OPCACMD entry of this example must be supplemented by an OPCA entry as in “Example” on page 92.

Example 2



```
  COMMANDS  ACTIONS  HELP
------------------------------------------------------------------------------
                         UET Keyword-Data Specification         Row 15 from 15
Command ===> ________________________________________________  SCROLL===> PAGE

Entry Type : User E-T Pairs        PolicyDB Name   : SCENARIO
Entry Name : NONSUBS               Enterprise Name : TEST
UET Entry  : DUMMY                 UET Type        : OPCACMD

Action   Keyword/Data(partial)
________ CMD
         (UXCINITS,'MVS $TI20-30,C=P')
****************************** Bottom of data ********************************
```

This example shows a command that is not related to a subsystem known to SA z/OS (see “Non-Subsystem Operations” on page 127), and which, accordingly, must be defined as a USER E-T pair (see “Non-Subsystem Operations” on page 127). The job name of the TWS request would be `DUMMY`.

TWS Automation recognizes such requests by the fact that the request name begins with 'UX'. In the example, TWS Automation simply issues the MVS command `$TI20-30,C=P`, which tells JES to change initiators 20 to 30 so that they process jobs of class P.

OPCAPARM

Purpose

The OPCAPARM entry supplies replacements for eventual request parameters and the name of a user-written timer module. The OPCAPARM entry is defined in the MESSAGES/USER DATA policy item of the subsystem that is to be put under control of TWS.

Format

```
Code 1          Code 2          Code 3          Value Returned
START                                           ,,
OPMSG                                           ,,USER_RTN1
```

Parameters

Code 1
Request specified in the TWS operation definition.

Code 2
Parameter 1 as specified in the TWS operation text.

Code 3
Parameter 2 as specified in the TWS operation text.

Value Returned
Enter the following parameters separated with commas.

parm1value
Substitution value used in the actual command.

parm2value
Substitution value used in the actual command.

timermod
Module called at the timer interval specified in the OPCA CODE entry for this subsystem. You must specify a timer module when the command specified in the OPCACMD entry relates to a subsystem defined to SA z/OS, but does not trigger a status change; see “User Functions Related to an SA z/OS-Defined Subsystem” on page 123.

Usage Notes

OPCAPARM is optional except for requests that relate to a subsystem defined to SA z/OS, but where the command specified in the OPCACMD entry is a user-supplied module that does not trigger a status change of the subsystem. In this case, a you must specify a timer module. See “Implementing Completion of a Request” on page 124 for more details.

Example

```
Code 1          Code 2          Code 3          Value Returned
STOP            FORCE           IMM             FORCE,,
```

This example shows a forced stop request for a subsystem. The TWS operation text would be STOP FORCE IMM. This would result in EHKVAR1 being set to FORCE and the INGREQ STOP command having a TYPE of FORCE. Because OPCAPARM does not have a *timermod* value, the default module EVJESPTE is used instead.

Chapter 12. The Structure of TWS Request Automation

This chapter explains the structure of TWS request automation in some detail.

Flow Overview

TWS Automation is an interface between NetView, TWS, and SA z/OS. These components provide the facilities that make up the interface. This section provides an introduction to these components and their interactions. The following are discussed:

- Initialization of the various components. See "Initialization"
- A description of the flow of a request from TWS to NetView and the return confirmation, including the modules that are involved. See "Conventional Request Flow"
- A description of the "Command Request Flow" on page 105

Initialization

Initialization involves the following two sequences:

1. Initialization of the TWS components.
2. Initialization of TWS Automation functions in each NetView. TWS Automation initialization includes the automated recovery sequences described in "Automated Recovery Functions" on page 136.

Conventional Request Flow

This section contains a detailed description of the flow of a request from TWS to NetView and the return confirmation, including the modules that are involved.

Figure 56 on page 98 uses a request to start RMF, located in a NetView domain NVREG with a workstation definition of NV04. This request is an operation in a TWS-defined application known as MAINT.

© Copyright IBM Corp. 1990, 2010

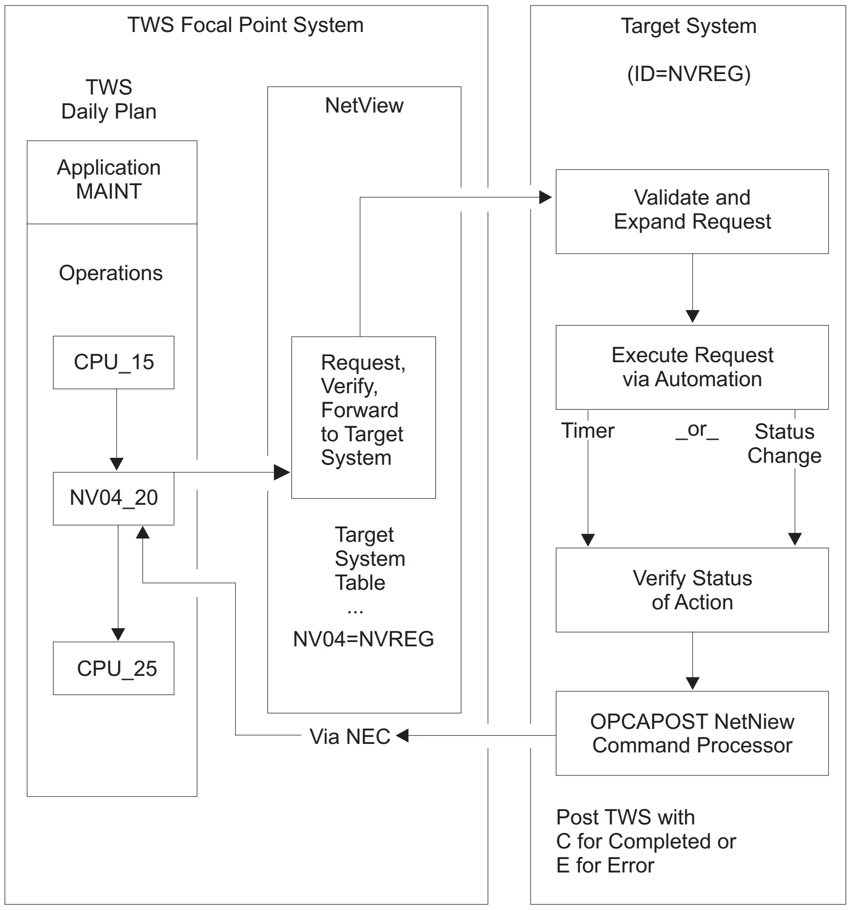


Figure 56. NetView-TWS Interface Flow. Note that syntax and definition errors, target system availability, recovery, and resynchronization via TWS API and NetView PPI are not shown in this example.

Using dependency control to ensure an orderly flow of operations, TWS defines the TWS-controlled application named MAINT. TWS defines the application on an automatic general workstation, specifying the NetView that the request is sent to. NV*xx* specifies a NetView automatic general workstation with a NetView domain index of *xx*. This is resolved in the Controller NetView to the target NetView domain ID using the definitions in the SA z/OS policy database. TWS can define the NV*xx* workstation with all regular specifications, such as parallel servers and special resources. If the NV*xx* index specifies *LOCAL, the command is processed locally.

In the MAINT example in Figure 56, TWS defines the last batch application that is processed before starting RMF with an operation number of 15. Once this completes properly, the normal TWS dependency control readies the NV04_20 operation on the NV04 workstation. This signifies that the request contained within the operation description field is sent to the NetView with a domain ID of NVREG.

TWS Automation uses the NetView PPI to transfer the request from TWS to NetView. This transfer is through the EQQUX007 exit in the OPC/ESA controller.

EQQUX007 Exit

Each change of status on any workstation causes TWS to call user exit 7 (EQQUX007). The TWS Automation user exit EVJUX007 calls modules EVJ07001 and EVJ07004:

- EVJ07001 sends automation commands and data across the NetView PPI for all status changes on NV*xx* workstations.
- EVJ07004 WTOs Operation status information for any operation that ends in Error or is changed from an Error state to any other TWS state. This information is used to update SDF and NMC monitoring of TWS operations.

Figure 57 shows the flow of the EVJ07001 exit.

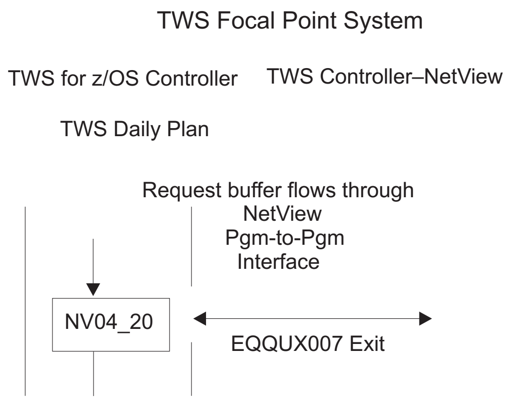


Figure 57. EVJ07001 Exit

When an NV*xx* workstation moves to the R (ready) status, the workstation generates a request buffer. Fields that are pointed to by registers in the EQQUX007 exit provide all of the data for the request buffer. For the layout of the fields in the request buffer, see Table 8 on page 120 and Table 9 on page 120.

The EQQUX007 exit logic that is supplied by TWS Automation verifies that all fields exist (except the optional request parameter fields). If this exit logic determines that any field is missing or the value is not valid, it issues an error WTO and changes the operation to E (error) status, with an error code indicating a user-definition error. Because the EQQUX007 exit cannot directly change the status of a TWS operation when an error code is posted to TWS, the EQQUX007 exit uses the EQQUSINT module to respond.

If the information is correct, TWS builds the request buffer and calls the CNMCNETV module, which is the NetView PPI module. This module transfers the request to the Controller NetView, where TWS verifies the return codes from the call function to ensure that there are no errors. If TWS detects errors, the EQQUSINT module changes the status to E (ended-in-error), with the error code on the basis of the PPI module return code. The module issues a WTO and completes processing the EQQUX007 logic. TWS Automation then restores registers and returns control to TWS.

If the TWS Automation EQQUX007 exit is unable to load the CNMCNETV module or use it to send data, it directs TWS to mark the requested operation in error, with an error code of UNTV. TWS Automation will attempt to reset operations that have ended in a UNTV error, subject to a user-defined time limitation, whenever the TWS controller is restarted.

Program-to-Program (PPI) Interface Dispatcher

The NetView program-to-program interface passes the request buffer to the PPI dispatcher task in the SA z/OS application. The PPI dispatcher task (EVJTOPPI), a NetView subtask, receives the requests for an SA z/OS action from the buffers of the EQQUX007 exit. Figure 58 shows this flow.

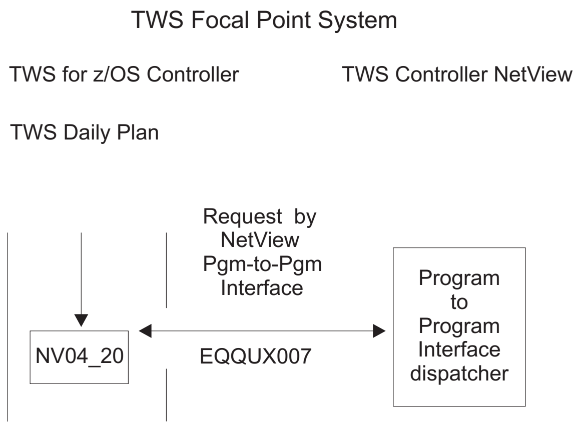


Figure 58. PPI Dispatcher

Based on the sending task identifier, the PPI dispatcher determines the function in SA z/OS that is sent. For TWS Automation, the dispatcher selects the verify function.

Verify Module (EVJESPVY)

The verify module, which runs on a NetView autotask, runs only in the Controller NetView. This module receives the action request buffer from the PPI dispatcher task. Figure 59 shows this process.

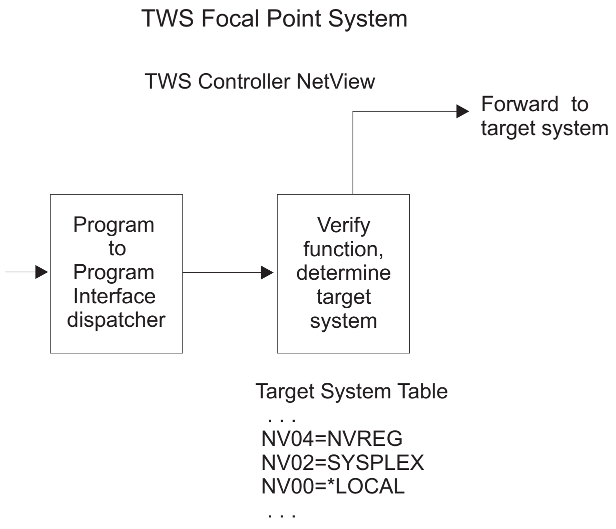


Figure 59. Verify Module

The verify module uses the NV*xx* index to obtain the destination NetView domain ID from the SA z/OS policy database. If the relevant NV*xx* index specifies SYSPLEX, then all SA z/OS systems in the local sysplex are queried for the status of the application that is associated with the job name of the request. The destination is determined to be the system that has the application in the most active state.

If the destination NetView and the requesting NetView are the same, TWS Automation logs the request buffer and invokes the request module. If the destination NetView and the requesting NetView are different, TWS Automation sends the request to the proper NetView domain by message forwarding.

If TWS Automation does not find the NV*xx* index then TWS Automation issues a message, posts the operation status to E (ended-in-error, U003), and logs the results. No communications can occur with this workstation until the definition is corrected. On the domain where the TWS controller is running, the workstation must be defined in the WORKSTATION DOMAINS policy object (ODM entry type). You must manually reset operations that are posted-in-error because TWS Automation carries out no automated recovery for definition errors.

If NV*xx* is associated with the sysplex that the TWS controller is running on (SYSPLEX keyword in the WORKSTATIONS DOMAIN entry) and TWS Automation does not find the job defined to any online SA z/OS in the local sysplex, then TWS Automation issues a message, posts the operation status to E (ended-in-error, S998), and logs the results. To cater for the situation where all domains where the job runs are offline, the operation will be retried if a gateway connection to another SA z/OS becomes active.

If TWS Automation successfully forwards messages, it logs the request buffer and returns control to the module. If TWS Automation cannot send the request, it issues an error message and logs it to indicate communication loss with the requested NetView domain. TWS Automation then posts the operation status to E (ended-in-error, S999) due to loss of contact. When TWS Automation re-establishes communications with this NetView domain, it checks for all outstanding errors because of loss of communications on this workstation. If TWS Automation finds any of these errors, it resets the TWS-operation status to R (ready), which re-invokes the EQQUX007 exit.

Request Module (EVJESPRQ)

The arrival of a request from the verify module drives the request module in the Tracker NetView. TWS Automation installs the request module on each system running an OPC/ESA Tracker. Figure 60 on page 102 shows the flow of this process.

The main function of the request module is to translate the TWS-generated request into a subsystem related command, or to schedule a user-defined function that is not related to a subsystem. The control flow of the module is shown in Figure 60 on page 102.

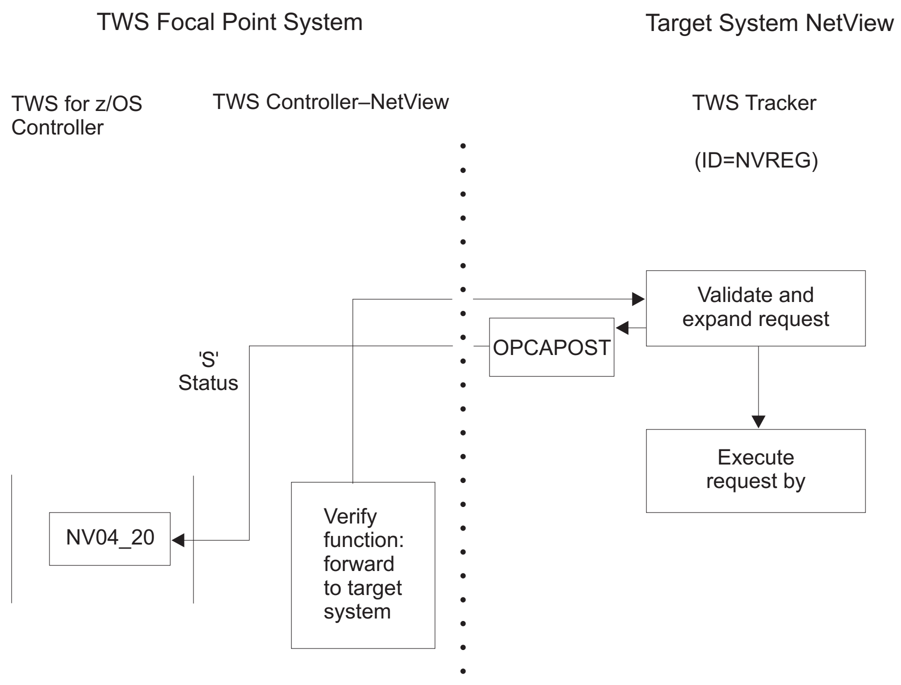


Figure 60. Request Module

If required, the request module uses definitions in the SA z/OS policy database to create the command that initiates the function requested. The policy database contains entries (OPCA, OPCACMD keywords; see Chapter 11, “MESSAGES/USER DATA Entries and USER E-T Pairs for TWS Automation,” on page 91) that the command text and the parameter syntax for the actual request are obtained from. If any of these entries are not found, the processing cannot continue. The OPCAPOST module posts an error to TWS, which logs the error and issues a WTO. Because this is a user-definition error, TWS attempts no automation recovery. The user must correct the definitions and reset the operations in error.

In Figure 60, the request module translates the requested action in the buffer to the SA z/OS command required to start RMF. The SA z/OS command then starts RMF.

Except for starting, stopping, or recycling SA z/OS-controlled subsystems, other functions may require user programming. To support these functions, TWS Automation provides a user exit capability. For a detailed description of user responsibilities required to handle a user call, see Chapter 14, “Guidelines for User-Written Operations,” on page 123.

For subsystem-related operations, the OPCAPOST command processor posts to TWS if the required entries are found in the policy database. TWS changes the status from R (ready) to S (started). TWS Automation then issues a timer request on the basis of the delay specified in the policy database (OPCA keyword, see “OPCA” on page 92), issues the command, and checks the return code. Then the request module terminates.

A change of subsystem status calls the status-change exit module. If the status change does not occur, the timer-driven module executes when the timer interval expires. This ensures that a request resulting in an unexpected status processes. For example, if TWS requests a START operation, and the subsystem fails to start due to a JCL error or other problem, then the OPCAPOST module posts TWS with an error status.

TWS Automation dynamically generates the TWS request by using definitions in the policy database and dynamic substitution of command fragments on the basis of the parameters.

Status Change Module (EVJESPSC)

SA z/OS calls the status-change module for each change of status. This module determines whether a status change is the result of a previous TWS Automation request. If the status change is not the result of a previous request, TWS Automation ignores the status change. Figure 61 shows the flow of this process.

If an outstanding request for the changed subsystem exists, and the new status is compliant with the expected status, TWS Automation cancels the timer. OPCAPOST updates the TWS operation status to C (completed) status.

With the timer values properly set and the operation processing normally, the change of status should always occur before the timer interval expires.

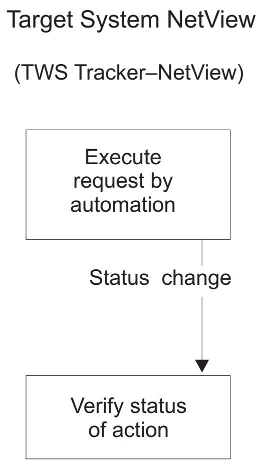


Figure 61. Status Change Module

Timer Module (EVJESPTE)

Under normal conditions, a request passed to SA z/OS results in the desired status change before the timer expires, and TWS Automation purges the timer. When this sequence does not occur, and the timer remains at the end of the timer interval, SA z/OS drives the timer module.

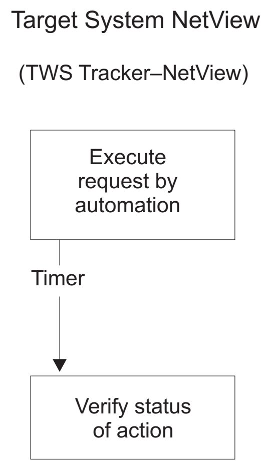


Figure 62. Timer Module

For subsystem-related functions, EVJESPTE compares the current status with the expected status. If a match is obtained, the OPCAPOST command processor posts a C (completed) status to TWS. If EVJESPTE determines a mismatch between the current and expected status, OPCAPOST posts an error to TWS for review by the TWS administrator. Figure 62 shows the flow of this process.

OPCAPOST Command Processor

The OPCAPOST command processor calls EQQUSINT, which passes the completion code to the TWS Tracker in this system. The TWS Tracker forwards the completion code to the system running the Controller through the mechanism used by OPC/ESA. Figure 63 shows the flow of this process.

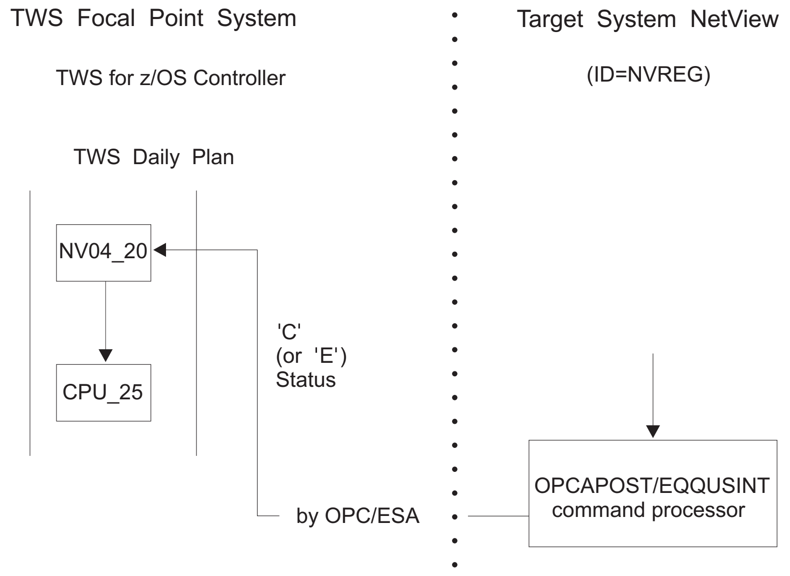


Figure 63. OPCAPOST Command Processor

Other functions use the OPCAPOST command. See “OPCAPOST” on page 117 for documentation on the syntax.

This module completes the processing for this specific TWS operation. If the request executes successfully, TWS Automation sets the TWS operation status to C (completed) and normal TWS-dependency control allows the next operation to start. See the CPU_25 batch job in Figure 63 on page 104. If the operation completes in error, TWS Automation sets an E status and a 4-character return code. The application does not continue processing until some intervention occurs. An operator or TWS Automation's recovery can sometimes provide this intervention.

Completion and Timer Flags

Both the status change and timer modules check for each other's completion. If one function completes first, the second function exits to avoid false double posting to TWS. Double posting is avoided by using timer and completion flags, as the following text discusses.

NetView schedules work on automated operator tasks on a first-in/first-out basis. Work elements resulting from the status change or timer modules become ready as NetView schedules them on the queue. Because NetView automated operator tasks process in a sequential manner, the queue can hold both the status change and the timer work elements at the same time. When this occurs, NetView must process only one work element, because handling both results in double posting to TWS, leading to errors in the TWS-defined application.

To avoid these errors, TWS Automation uses two flags. One of these flags is related to the status change module, the other to the timer module. When the modules are called, they examine the flags. If neither flag is on, the respective module turns on the flag associated with the active function and continues processing. If a flag is found on, the function exits, because the processing is completed by the other function while this work element was queued. This process depends on defining a single automated operator task for each NetView Tracker.

Command Request Flow

The request interface allows the TWS user to send SA z/OS or NetView commands to SA z/OS. This section describes:

- The TWS process flow, see "Process Flow"
- Synchronous and asynchronous commands, see "Synchronous and Asynchronous Commands" on page 107

Process Flow

The flow consists of the following steps, as shown in Figure 64 on page 106:

1. TWS builds and sends the request buffer
2. SA z/OS verifies the request buffer
3. SA z/OS determines the destination NetView
4. The EVJESCMD routine processes the command
5. The EVJESCMD routine posts completion of the command

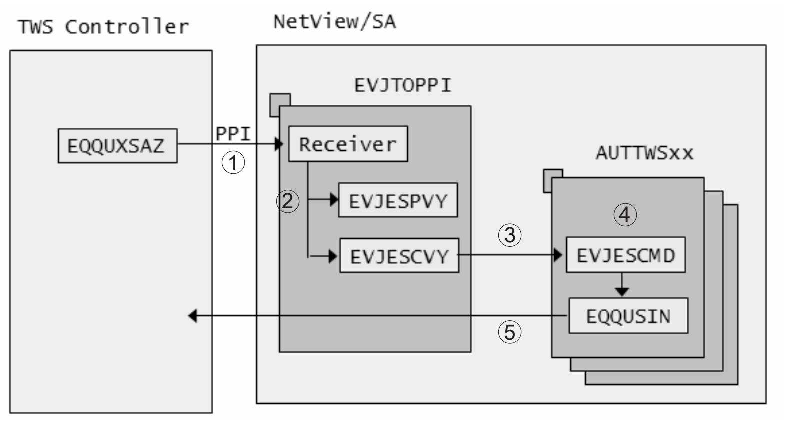


Figure 64. Command Request Process Flow

1. TWS Sends the Request Buffer

Exit EQQUXSAZ, which is associated with an automation workstation, is called when an operation that is defined to the automation workstation is submitted. Prior to calling the exit, TWS performs variable substitution on the command text. The exit then builds the request buffer from the information that is provided by TWS and sends it to the local SA z/OS (that is, one that is running on the same system as the TWS controller) via the NetView PPI.

2. SA z/OS Verifies the Request Buffer

The NetView PPI passes the request buffer to the SA z/OS receiver task (EVJTOPPI), which determines the function that is being asked for based on the request identifier. If the request identifier is EVJESCMD, it calls routine EVJESCVY, which verifies the request buffer.

3. SA z/OS Determines the Destination NetView

If the NetView destination ID is present, this is used. Otherwise, the EVJESCVY routine uses the workstation name to obtain the destination NetView domain ID from the SA z/OS policy database (OPCA domain ID):

- If the relevant workstation name specifies *LOCAL, the command is processed locally.
- If the workstation name specifies SYSPLEX, the command is processed locally.

If an automated function is specified, the command is executed on the specified task. This gives the user an easy way to serialize the processing of commands if needed.

4. EVJESCMD Routine Processes the Command

The main function of the EVJESCMD routine is to process the command that is passed in the request buffer, as follows:

1. First, it invokes the AOFEXC20 installation exit that checks whether the workstation is authorized to execute the command. If the exit rejects the command, SA z/OS posts the operation in error.
2. After logging the command in the netlog, the EVJESCMD routine executes the command in a PIPE, using the timer interval that is specified in the completion info option as the maximum wait time. If the command takes longer than the specified time interval, error code U004 is posted.

5. The EVJESCMD Routine Posts Completion of the Command

If the command does not complete within the specified time interval, SA z/OS posts the operation in error via the EQQUSIN service call. Otherwise, it updates the TWS operation status to C (completed).

Return codes from the command execution that are greater than the maximum expected (as specified in the completion information) are sent back to TWS via the EQQUSIN interface by setting the error code to R*nnn* ,where *nnn* is the return code from the command. If an error message is given by the command, it is passed back to TWS. All messages that result from the failed command execution are written to the netlog, together with message EVJ061I.

Processing INGMOVE and INGREQ Commands

The INGMOVE and INGREQ commands use the FDBK parameter that causes the final result of the command execution to be reported back to the requested destination. It thus makes the these commands quasi-synchronous.

Processing for INGMOVE differs in that it initiates the move operation, but does not wait for the completion of the move operation.

Processing of an INGREQ command is slightly different because, by definition, the INGREQ command works asynchronously.

Invoking the INGREQ command causes a start or stop vote to be placed against the resources that are specified in the command. This does not necessarily mean that the resource is immediately started or stopped: It depends on the presence of other, higher priority votes (that may request the opposite) or whether all dependencies are fulfilled.

After informing the automation manager about the new request, the INGREQ command checks whether the specified resource is already in the desired state. If so, it posts the operation complete. Otherwise, it subscribes a status observer for the specified resource and sets up a timer. If the timer expires and resource is not in the requested state, INGREQ sets the operation status in error. It calls the EQQUSINT service, which passes the completion code to the TWS tracker on the local system.

Synchronous and Asynchronous Commands

Processing of commands can be either synchronous or asynchronous.

A synchronous command is considered to have achieved the expected result when the command finishes and returns to SA z/OS. SA z/OS then posts completion of the command once control is returned to SA z/OS. The maximum wait time allows the user to control how long SA z/OS should wait before considering the command to be in error and posting the operation in error.

An asynchronous command is a command that may not necessarily have achieved the expected result when control is returned to SA z/OS, as shown in Figure 65 on page 108.

Command Request Flow

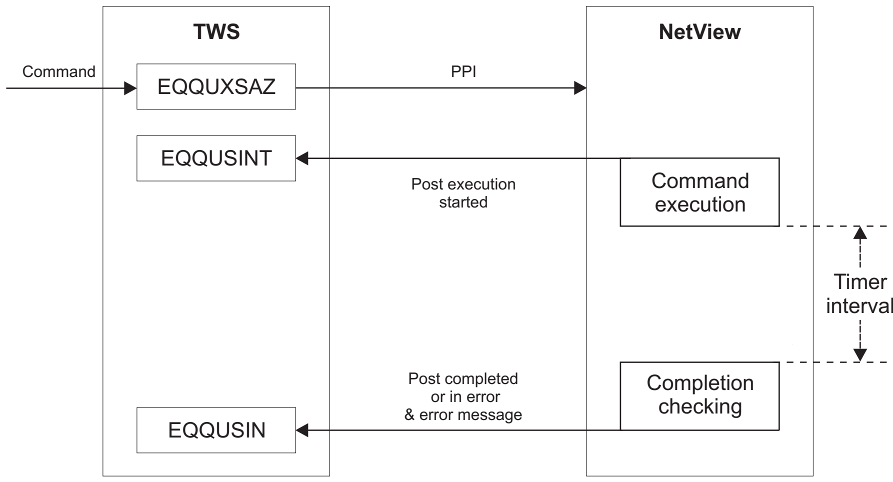


Figure 65. Asynchronous Command Processing

User-supplied Completion Checking Routine

It is the installation's responsibility to check whether the command has achieved what it has been asked to do within the specified time interval.

A completion checking routine can be used to examine whether the command has achieved the expected result. SA z/OS uses the maximum wait time to set up a timer. When the timer interval expires, SA z/OS invokes the checking routine. The checking routine is responsible for posting the operation as either complete or in error using the OPCAPOST command.

Note: A command is considered to be asynchronous if a checking routine is provided.

INGREQ and INGMOVE are synchronous commands, therefore a maximum wait time is required. The command is considered to be complete when the specified resource has reached or is already in the requested state. If more than one resource is specified in the INGREQ or INGMOVE command, all resources must be in the requested state before the command is considered complete. The timer value specified should be large enough to accommodate the longest interval of time that the requested function may take under normal operating conditions.

Chapter 13. TWS Automation Common Routines and Data Areas

This chapter contains the common routines that are supplied by TWS Automation. Furthermore, it describes the data areas that are used to transfer requests from TWS to SA z/OS.

TWS Automation Common Routines

This chapter describes TWS Automation common routines that request information or perform tasks associated with TWS Automation. You can use these common routines in automation procedures you create. Examples, sample routines, and data area information are given to show how this might be done.

TWS Automation provides INGOPC and OPCAQRY to retrieve and update TWS Automation-unique information. These routines can also be used in user-written extensions of TWS Automation. The following routines are arranged alphabetically for easy reference.

Note:

The common routines listed below will automatically locate the active TWS Controller before executing PIF requests and will schedule all PIF interface calls on the same autotask to serialize access to the EQQMLOG data set.

The active TWS Controller can be located in user routines by issuing the following sequence of commands:

```
INGLIST CATEGORY=OPC,SUBTYPE=CONTROLLER,OBSERVED=AVAILABLE,
OUTMODE=LINE
```

Then for each subsystem returned from INGLIST issue:

```
INGVARS GET subsystems_returned_above TWSACT OUTMODE=LINE
```

The active TWS Controller will be in AVAILABLE status with the manager variable 'TWSACT' set to 'ACTIVE'.

OPCACAL

Purpose

The OPCACAL command retrieves TWS calendar status information. Use this command for your automation CLISTs. OPCACAL uses the EQQYCOM (also called the PIF) interface in TWS. For more information about this interface, see the *OPC/ESA Interfaces Guide*.

To show the use of this command, TWS Automation provides a REXX sample called EVJERCAL.

It is recommended that you use INGOPC.

Format

Syntax

```
OPCACAL [SUBSYS=subsystem,CALENDAR=calname]
```

Parameters

SUBSYS=*subsystem*
OPC/ESA subsystem ID — 4 characters.

OPCA is the default subsystem name.

CALENDAR=*calname*
Calendar ID — 16 characters.

DEFAULT is the default calendar name, used if this parameter is not coded.

Restrictions

OPCACAL supports multiple active controllers only if an appropriate value is supplied for the SUBSYS= parameter.

Usage Notes

OPCACAL returns data in the following format:

- EVJ440I date_format day_of_week Work, or
- EVJ440I date_format day_of week Free

The date format is set by the NetView DEFAULTS LONGDATE value. For example:

```
EVJ440I 05/03/04 MONDAY Work
```

or

```
EVJ440I 20040502 SUNDAY Free
```

The EVJ440I message is PIPEd to the CONSOLE ONLY and does not appear in the netlog. The message can be processed by calling OPCACAL in a PIPE.

OPCACOMP

Purpose

The OPCACOMP command completes execution of a subsystem-related request by updating the TWS Automation control information and calling OPCAPOST (see "OPCAPOST" on page 117).

Format

Syntax

```
OPCACOMP  subsys,sequence_number,status [,error_code]
```

Parameters

subsys
Subsystem or pseudo-subsystem to identify the request.

sequence_number
Sequence number assigned to this request by the EVJESPVY module.

status
Operation status reflected to TWS Valid statuses:

C	Complete
E	Error

error_code
Error code — 4-character value
Takes the form *annn*, where *a* is alphabetic and *nnn* are numerics.
Do not specify the values U*xxx* and S*xxx*; reserve them for TWS Automation.
If the status is error, the error code is returned to TWS.

Return codes

0	Okay.
1	Error occurred, message issued.

Usage Notes

Call this routine in user-written functions that are related to a subsystem defined to SA z/OS whenever the standard modules for completing a request (EVJESPSC and EVJESPTE) cannot be used. See "Implementing Completion of a Request" on page 124.

Example

```
OPCACOMP  RMF,842,E,R028
```

This example shows setting the operation requested for RMF in an error status with an error code of R028.

OPCALIST

Purpose

The OPCALIST command retrieves TWS data. Use this command in your own automation CLISTS. It is recommended that you use the command INGOPC REQ=LIST.

The module creates a CPOPCOM call to TWS to retrieve the data. OPCALIST uses the EQQYCOM (also called the PIF) interface in TWS. For more information about this interface, see the *OPC/ESA Interfaces Guide*.

Format

Syntax

```
OPCALIST SUBSYS=subsystem,ADID=id,IA=yymmddhhmm,
         PRIORITY=nnnn,ERRCODE=cccc,STATUS=s,OPNO=nnnn,
         JOBNAME=name,WSNAME=name,
         GROUP=groupname,OWNER=name
```

Parameters

SUBSYS=*subsystem*
TWS subsystem ID — 4 characters.

ADID=*id*
Application description ID — up to 16 characters.

IA=*yymmddhhmm*
Input arrival date *yymmdd* and time *hhmm*.

PRIORITY=*nnnn*
Priority — 4 digits.

ERRCODE=*cccc*
Error code — 4 characters.

STATUS=*s*
Occurrence status. Valid statuses:

R	Ready
S	Started
C	Completed
E	Error
I	Interrupted

Refer to the TWS documentation for more information.

OPNO=*nnnn*
Operation number — 4 digits.

JOBNAME=*name*
Job name — up to 8 characters.

WSNAME=*name*
Workstation name — 4 characters.

GROUP=*groupname*
OPC/ESA application group name — up to 8 characters.

OWNER=*name*
OPC/ESA application owner name — up to 16 characters.

Restrictions

OPCALIST does not support multiple active controllers. To obtain information from TWS in a multiple controller environment, use INGOPC.

Usage Notes

Only as many parameters are required as necessary to identify the application(s) requested. Some parameters may be left unspecified (they will default). Some may be generic; that is they may be only partial and end with an asterisk (*) to indicate a partial match. You may also use a percent sign (%) to substitute for a single character.

The response to the OPCALIST is made up of three messages. The following example below shows a typical response where:

EVJ410I
This is the message header, showing row titles. This is always present.

EVJ411I
This is the detail message. If there are no entries matching the selection criteria this message is not produced.

EVJ412I
This is the end of request message.

Response details are:

ADID
Application Description Id — up to 16 characters

JOBNAME
Job Name — up to 8 characters

WS
Workstation Name — up to 4 characters

OPNO
Operation Number — up to 4 numbers

S Status (See “Parameters” on page 112 for valid statuses)

ERRC
Error Code (set to none for no error)

IA Input Arrival — date (yymmdd) and time (hhmm)

OPTEXT
Descriptive Text — up to 24 characters

Example

```
EVJ410I   ADID   JOBNAME   WS  OPNO S ERRC IA          OPTEXT
EVJ411I   MAINT2 RMF      NV05 0010 C NONE 9707190615  STOP
EVJ411I   MAINT2 RMF      NV05 0015 C NONE 9707190615  START
EVJ411I   MAINT2 RMF      NV05 0010 C NONE 9707200615  STOP
EVJ411I   MAINT2 RMF      NV05 0015 C NONE 9707200615  START
EVJ412I   END OF REQUEST
```

OPCAMOD

Purpose

The OPCAMOD command modifies TWS data. This command is used in the OPCACMD CLIST and could be used in your own automation CLISTS. It is recommended that you use the command INGOPC REQ=MOD.

The module creates a CPOPCOM or CPOCCOM call to TWS to perform occurrence or operation changes. OPCAMOD uses the EQQYCOM (also called the PIF) interface in TWS. For more information about this interface, see the *OPC/ESA Interfaces Guide*.

Format

Syntax

```
OPCAMOD SUBSYS=subsystem,ADID=id,IA=yymmddhhmm,
        IANEW=yymmddhhmm,DEADLINE=yymmddhhmm,PRIORITY=nnnn,
        ERRCODE=cccc,OPNO=nnnn,STATUS=s,
        JOBNAME=name,WSNAME=name,DESC=text,
        EDUR=hhmm,PSUSE=nnnn,R1USE=nnnn,R2USE= nnnn,
        JCLASS=c,AEC=Y|N,ASUB=Y|N,AJR=Y|N,TIMEDEP=Y|N,
        CLATE=Y|N,HRC=value,FORM=value,OPIA=yymmddhhmm,
        OPDL=yymmddhhmm,RERUT=Y|N,USERDATA=userdata,
        RESTA=Y|N,DEADWTO=Y|N
```

Parameters

SUBSYS=*subsystem*
TWS/A subsystem ID — 4 characters (default OPCA).

ADID=*id*
Application description ID — up to 16 characters (required).

IA=*yymmddhhmm*
Input arrival date *yymmdd* and time *hhmm* (required).

IANEW=*yymmddhhmm*
New input arrival date and time.

DEADLINE=*yymmddhhmm*
Deadline date and time.

PRIORITY=*nnnn*
Priority — 4 digits.

ERRCODE=*cccc*
Error code — 4 characters.

STATUS=*s*
Occurrence status. Valid statuses:

R	Ready
S	Started
C	Complete
E	Error
I	Interrupted

Refer to the TWS documentation for more information.

JOBNAME=*name*
Job name — up to 8 characters.

WSNAME=*name*
Workstation name — 4 characters.

DESC=*text*
Descriptive text — 24 characters.

EDUR=*hhmm*
Estimated duration (hours and minutes).

PSUSE=*nnnn*
Number of parallel servers required — 4 digits.

R1USE=*nnnn*
Amount of resource 1 required — 4 digits.

R2USE=*nnnn*
Amount of resource 2 required — 4 digits.

JCLASS=*c*
MVS job class — 1 character.

AEC=Y|N

Y	Perform automatic error completion.
N	Do not perform automatic error completion.

ASUB=Y|N

Y	Perform automatic job submission.
N	Do not perform automatic job submission.

AJR=Y|N

Y	Perform automatic job hold and release.
N	Do not perform automatic job hold and release.

TIMEDEP=Y|N

Y	Time dependent job.
N	Not a time dependent job.

CLATE=Y|N

Y	Cancel if time job and late.
N	Do not cancel if time job and late.

HRC
Highest successful return code.

FORM=*value*
Form number — 8 characters.

OPIA=*yymmddhhmm*
Operation input arrival date and time.

OPDL=*yymmddhhmm*
Operation deadline date and time.

RERUT=Y|N

Y	Reroutable operation.
N	Not a reroutable operation.

USERDATA=*userdata*
User data — up to 16 characters.

RESTA=Y|N

Y Restartable operation.

N Not a restartable operation.

DEADWTO=Y|N

Y Issue WTO if deadline missed.

N Do not issue WTO if deadline missed.

Restrictions

OPCAMOD does not support multiple active controllers. To obtain information from TWS in a multiple controller environment, use INGOPC.

Usage Notes

OPCAMOD can be used to set the status of operations occurring on general workstations that do not automatically report (such as CPUs). OPCAPOST sets the status of operations that occur on automatically reporting general workstations (a NetView, for example).

Example 2

```
OPCAMOD SUBSYS=OPCA,ADID=TEST,IA=9403100900,OPNO=0010,
  STATUS=W
```

This example will set the status of operation number 0010 in application test, to waiting.

Example 3

```
OPCAMOD SUBSYS=OPCA,ADID=TEST,IA=9403100900,STATUS=W
```

This example will set the occurrence status of application TEST to WAITING. All operations in application TEST will be set to WAITING.

OPCAPOST

Purpose

OPCAPOST posts the status of an TWS Automation operation back to TWS. Because OPCAPOST uses the EQQUSINT interface, it can only change the status of operations on automatic reporting workstations. For more information on this interface, see *OPC/ESA Installation and Customization*.

Format

Syntax

```
OPCAPOST ADNAME=adname,WSNAME=wwww,OPNUM=nnnn,TYPE={S|C|I|E|X},
         ERRCODE=xxxx[,SUB=subsystem][,JOBNAME=jobname]
         [,ITIME=hhmm][,IDATE=yymmdd][ERRMSG=message]
```

Parameters

ADNAME=*adname*
Application name: 1 to 16 characters.

WSNAME=*wwww*
Workstation name: 1 to 4 characters.

OPNUM=*nn*
Operations number: 4 digits.

TYPE={S|C|I|E|X}
Type of call: 1 character. Acceptable event types are:

S	Started
C	Complete
I	Interrupted
E	Error
X	Reset

ERRCODE=*xxxx*
Error code: 4 characters.

Note: This parameter is only valid with TYPE=E.

SUB=*subsystem*
The MVS subsystem ID of the tracker that is associated with the controller: 4 characters (optional).

JOBNAME=*jobname*
The jobname that is associated with the operation: 1 to 8 characters (optional).

ITIME=*hhmm*
The input arrival time.

IDATE=*yymmdd*
The input arrival date.

ERRMSG=*message*
The message that is associated with the error code. The message can be examined in the TWS Controller MLOG.

Usage Notes

OPCAPOST can be used to set the status of an operation that occurs on an automatically reporting general workstation (a NetView, for example) only. Due to restrictions in the TWS interface, OPCAPOST cannot set the status of operations on general workstations that are not automatically reporting (such as CPUs). INGOPC REQ=MOD is available to set the status of such operations if this is required.

TWS Automation sets a return code on completion of the execution of the OPCAPOST command processor, as follows:

0	Successful command
4	Parameter error
8	OPCAPOST failed.

Data Areas

This section contains the following:

- "Requestor ID Block (&EHKVAR9)"
- "Request Buffer for Standard Subsystem Operations" on page 120

Requestor ID Block (&EHKVAR9)

TWS Automation sets the task global variable (&EHKVAR9) in the request module and passes it to the user module, as follows:

```
Jobname, Sequence #, Module Name, Domain ID
```

where the variables have the following meanings:

Jobname
The SA z/OS job name.

Sequence #
The TWS sequence number.

Module Name
Check module name that is invoked once the requested function has been completed.

Domain ID
NetView domain ID that the function was executed in.

For example:

```
RMF,7842,OPCACOMP,NETVT
```

Note: Other options can also use this block. Although OPCACOMP is shipped with the TWS Automation option, you can also use a user-supplied module.

Request Buffer for Standard Subsystem Operations

Table 8 shows the request buffer layout for standard subsystem operations:

Table 8. Request Buffer Layout for Standard Subsystem Operations. Length represents the maximum length if the format is variable.

Field	**Length**	**Format Fixed/Variable**	**Value**	**Obtained From**
Request ID	8	F	EVJESPRQ	constant
delimiter	1	F	blank	constant
Application name	16	V	variable	ADNAME
delimiter	1	F	blank	constant
Workstation name	4	F	NV*nn*	WSNAME
delimiter	1	F	blank	constant
Operation no	3	V	1 – 255	OPNO
delimiter	1	F	blank	constant
Job name	8	V	variable	JOBNAME
delimiter	1	F	blank	constant
Request	8	V	variable	first field in TXTOP
delimiter	1	F	blank	constant
Parameter 1 (optional)	*	V	variable	second field in TXTOP
delimiter	1	F	blank	constant
Parameter 2 (optional)	*	V	variable	third field in TXTOP

Note: The length of Parameter 1 or 2 is from 1 to 8 characters.

Table 9 shows the request buffer layout for non-subsystem user extension (UXaaaaaa) operations:

Table 9. Request Buffer Layout for Non-Subsystem, User Extension (UXaaaaaa) Operations. Length represents the maximum length if the format is variable.

Field	**Length**	**Format Fixed/Variable**	**Value**	**Obtained From**
Request ID	8	F	EVJESPRQ	constant
delimiter	1	F	blank	constant
Application name	16	V	variable	ADNAME
delimiter	1	F	blank	constant
Workstation name	4	F	NV*nn*	WSNAME
delimiter	1	F	blank	constant
Operation no	3	V	1 – 255	OPNO
delimiter	1	F	blank	constant
Job name	8	V	variable	JOBNAME
delimiter	1	F	blank	constant
Request	24	V	variable	TXTOP

Any parameter with a variable length is left-adjusted and all trailing blanks are ignored.

Chapter 14. Guidelines for User-Written Operations

TWS Automation allows two types of user-supplied extensions for implementation of functions beyond those provided by TWS Automation. These facilities provide support for the following types of user-supplied modules:

- A non-SA z/OS command or function that performs an action for a subsystem known to SA z/OS.
- An independent user-supplied function that is scheduled for the user. This type of function uses TWS Automation as a communications vehicle between TWS and the user-supplied module. A relationship is not required with any SA z/OS-defined subsystems.

The following sections describe an overview of each of these types of user-supplied modules and provide examples of each module's possible use.

User Functions Related to an SA z/OS-Defined Subsystem

TWS Automation provides support for stopping and starting SA z/OS-defined subsystems. Certain environments require you to issue a command or to perform a function outside the scope of SA z/OS. This may include a situation where a system command needs issuing or where a user-written function needs to perform a logical decision.

For example, you may need to issue a system command before taking action on a subsystem. If you always issue this command, specify it as part of the startup sequence in the SA z/OS policy database. However, because you may not need to use this command under certain conditions, TWS can initiate a user-supplied module to perform the command. You can split the startup sequence with the system commands, so TWS executes them separately from the subsystem startup commands. If this is the case, define each command sequence to TWS as an operation. Using scheduling parameters, such as specific types of days, you can include or exclude certain operations.

For example, consider a subsystem that normally runs on a specific processor. On weekends, you use this processor for testing purposes and move the application to another, perhaps smaller, processor within the same complex. On the days that the application needs moving, you need several VTAM VARY commands to start the VTAM application statements.

In TWS, you can define an extra operation or application that runs on the first free day of each period, and another that runs on the first working day of each period. TWS calls the CLIST containing the VTAM commands. This allows the issuing of the appropriate VARY commands when needed before you start the application subsystem on the correct processor.

Triggering a user-written CLIST provides another example. This determines if all users of a specific application are logged off before issuing the commands to take down the subsystem.

Flow of Control

In a situation where a non-SA z/OS command needs issuing, specify the user CLIST or command processor in the OPCACMD entry of the subsystem. In

© Copyright IBM Corp. 1990, 2010

response to the request, instead of issuing an SA z/OS command, TWS Automation passes control to this user CLIST or command processor. All information available is made accessible to the user-supplied module. If the user-supplied module does not trigger a status change of the subsystem and returns control to TWS Automation synchronously, you are responsible for completing the operation. This should be done by calling OPCACOMP once the results of these commands are analyzed. The OPCACOMP module ensures that actions are accomplished in the correct sequence, does some housekeeping, updates the SA z/OS control information, and calls the OPCAPOST command processor to return the specified completion code to TWS; for more details see "Implementing Completion of a Request."

Figure 66 shows the flow including the user responsibilities.

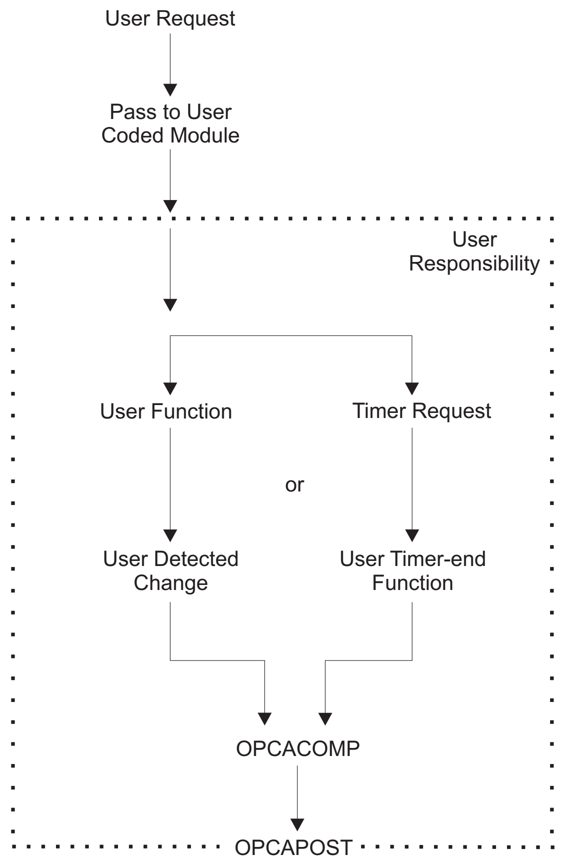


Figure 66. Request Flow for a Subsystem-Related User Function

To simplify implementation, you may plan to only use the timer function or only the detection of the completion of the command. If you use only the event-driven method, then consider what happens if the anticipated event fails to occur.

Implementing Completion of a Request

The general mechanism for executing requests for subsystems that have an OPCA entry coded in their MESSAGES/USER DATA policy item is as follows:

1. The request (with one or two optional parameters) and the job name of the TWS operation are passed to TWS Automation.
2. TWS Automation identifies the SA z/OS definition of the subsystem through the job name of the TWS operation.

3. TWS Automation retrieves the expected result, the timer interval, and possibly a timer name, for the respective request/parameter combination from the OPCA entry of the subsystem.
4. TWS Automation then checks if an OPCAPARM entry is present and if that entry contains a timer module name.
5. If a timer module is specified, TWS Automation sets the timer with this timer module. Otherwise, it uses a standard timer module (EVJESPTE).
6. TWS Automation issues the command that is specified in the OPCACMD entry.

After the command has been issued, two things remain to be done:
- If the command was executed successfully, the TWS control information for this subsystem must be updated.
- The (positive or negative) result of the request must be posted back to TWS.

How this is done depends on the type of command specified in the OPCACMD entry.

Using TWS Automation Standard Modules

If the command specified in the OPCACMD entry triggers a status change of the subsystem to RUNNING, UP or AUTODOWN (no matter whether it is a user-written command or an SA z/OS standard command), you can leave it to TWS Automation to perform these two tasks. The modules responsible for this are the status change module (EVJESPSC) and the standard timer module (EVJESPTE), already mentioned in step 5 above; two flags, a completion flag and a timer flag, ensure that both modules are not active at the same time. The two standard modules operate as follows:

1. The *status change module* is called whenever the status of the subsystem changes. It first checks the timer flag. If that flag is set, EVJESPSC terminates at once. If the flag is not set, EVJESPSC checks if the status change is the result of a TWS request, and if the new status is identical to the expected result as specified in the OPCA entry. When both conditions are satisfied, the status change module assumes that the request was successfully executed and
   a. Purges the timer defined in the OPCA entry.
   b. Sets the completion flag in the TWS control information for this subsystem.
   c. Actualizes further fields of the TWS control information.
   d. Posts the result of the request back to TWS.
2. The *timer module* (EVJESPTE) is called after the timer set in step 5 has expired (except when you have specified your own timer module in the OPCAPARM entry). It first checks the completion flag. If that flag is set, EVJESPTE will terminate at once. If the completion flag is not set, EVJESPTE sets the timer flag and compares the current state of the subsystem with the expected result of the OPCA entry. If both are compliant, the timer module assumes that the request was executed successfully; it updates the TWS control information accordingly and posts a positive result back to TWS. If they are not compliant, it only posts the failure of the request back to TWS.

Programming your own Completion Routines

If you specify a command in the OPCACMD entry that does not change the status of the subsystem to UP, RUNNING or AUTODOWN, then you cannot use the standard modules for completing the request. In this case, you must perform the update of the TWS control information and the posting of the result to TWS. You can do that in the command module (specified in the OPCACMD entry) or in a user-written timer module (specified in the OPCAPARM entry) or in both. The user-written timer module is called by TWS Automation in the following format:

```
MODULE_NAME subsystem_name expected_result TWS_application_ID
            TWS_workstation_ID request_sequence_number
```

To simplify completion of a user-written module, TWS Automation provides the following facilities.

The OPCACOMP Command: This command, which is described in more detail in "OPCACOMP" on page 111, updates the TWS control information and posts the result of the request back to TWS.

In particular, OPCACOMP first checks if the timer flag is set. If so, it will terminate at once. If not, it will

1. Set the completion flag in the TWS control information of the subsystem that it is called for.
2. Actualize further fields of the TWS control information.
3. Post the result of the request back to TWS.

The main difference between OPCACOMP and the standard modules (EVJESPSC and EVJESTPE) is that OPCACOMP does not check if the current status of the subsystem is in agreement with the expected result. Rather, it requires the (positive or negative) result of the request as one of its input parameters, and usually simply forwards this result to TWS. Thus, a user-written module must itself decide whether or not the request was executed successfully. It can then pass that information to OPCACOMP in order that the request be completed in an orderly manner.

One of the input parameters for OPCACOMP is the sequence number of the current request (see "OPCACOMP" on page 111). TWS Automation provides this and other information in some task global variables. Note, however, *that it will do this only when you have specified a timer module in the OPCAPARM entry*. You can specify a user-written timer module or the EVJESPTE standard module in the OPCAPARM entry. The following section describes the information contained in the global variables.

The &EHKVAR7, &EHKVAR8, and &EHKVAR9 Variables: When you supply a timer check module in the OPCAPARM entry (third value of the **Value Returned** field) TWS Automation sets some task global variables as follows:

&EHKVAR7	This variable contains the expected status, the timer interval, and the timer ID as specified in the OPCA entry. The values are separated by commas.
&EHKVAR8	This variable contains the string 'OPC'.
&EHKVAR9	This value contains the subsystem name, the sequence number, the name of the timer check module and the domain, separated by commas. This is also known as the Requestor ID block; see "Requestor ID Block (&EHKVAR9)" on page 119.

Do not modify the information in the task global variables. TWS uses information in &EHKVAR7 if the timer is purged. The SA z/OS problem determination uses information in &EHKVAR8. In order to call OPCACOMP, the sequence number of the current request must be known; this number is stored in &EHKVAR9.

Non-Subsystem Operations

Operations of this type, containing requests named UX*xxxxxx*, allow you to perform commands that are independent of a specific subsystem. Figure 67 shows the flow for these types of operations.

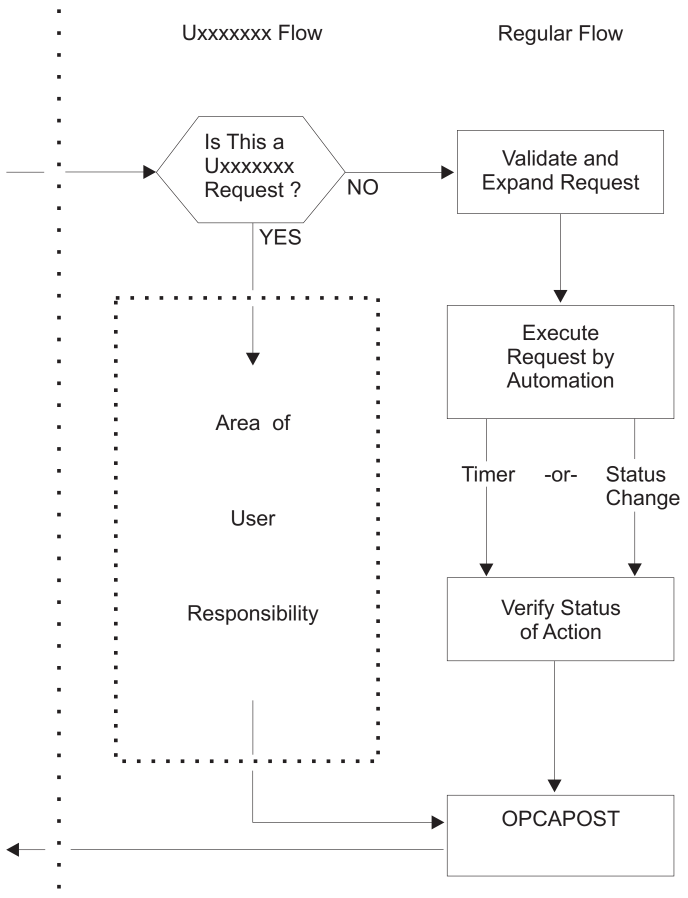

Uxxxxxxx Flow
Regular Flow
Is This a Uxxxxxxx Request ?
NO
YES
Validate and Expand Request
Execute Request by Automation
Area of User Responsibility
Timer
-or-
Status Change
Verify Status of Action
OPCAPOST

Figure 67. User Exit UXxxxxxx Flow

TWS Automation uses this type of exit for several purposes. At any point in the production cycle, TWS Automation allows you to invoke a user CLIST or procedure that can interact with system resources, such as the storage management subsystem.

Let's consider an example. Suppose, in a specific application flow within TWS, return codes show action that is taken by operations. When a specific job in this application completes, one of several user completion codes can result.

- A completion code of 0 indicates that application processing is to continue to the next operation.
- A user completion code of 50 indicates that the next two operations are skipped.
- A user completion code of 70 indicates that the application is completed at this operation.

Any other completion codes are treated as errors. Figure 68 on page 128 shows the subject operations in this application, and the desired flow of control on the basis of the completion codes of the job that runs as part of the CPU_20 operation.

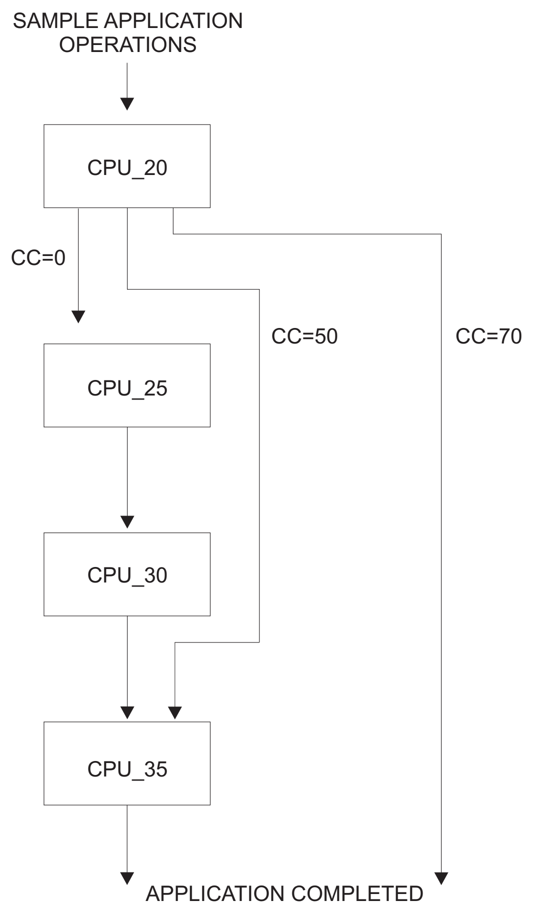


Figure 68. Completion Code Driven Application Flow

In the preceding example, TWS handles all completion code situations, except 50 and 70, which it intercepts. TWS accomplishes this interception in several fashions, such as user code in a JJC error exit. This code could then drive TWS Automation with a user exit (UX*xxxxxx*) request. This request would be passed to the specified NetView to drive a user-written script. In the user script the INGOPC REQ=MOD could then be used to modify the current plan for the application in question on the basis of the completion code received as part of the user exit request.

Flow of Control

When the name of a request starts with `UX`, TWS Automation assumes that the request is not related to a subsystem known to SA z/OS. As before, it expects to find an OPCACMD entry within a policy object that is identified through the **Job name** field of the TWS operation. However, if no match is found for USER E-T pair 'OPCACMD jobname', then TWS Automation will check for USER E-T pair 'OPCACMD OPCA', and if again no match is found, TWS Automation will check for USER E-T pair 'OPCACMD subsystem'. Although user exit processing is designed to be non-subsystem related, this approach provides flexibility for users who have jobnames that do not match subsystems names but still prefer subsystem-related processing. There are however two differences compared to subsystem-related requests:

- The only keyword that is needed is OPCACMD. OPCA and OPCAPARM are ignored.The CMD attributes of the OPCACMD entry should have the following format:

  ```
  CMD=(UXxxxxxx,,'userfunc &EHKVAR1')
  ```

For &EHKVAR1, see “Parameters Passed to a User Exit.”
- The policy object identified through the **Job name** field of the TWS operation should be a USER E-T pair (see “OPCACMD” on page 94).

For non-subsystem requests, TWS Automation immediately tries to issue the command specified in the OPCACMD entry. After issuing the command, the request module of TWS Automation terminates. It is up to the user function to determine whether or not the request was executed successfully. The user function should then call OPCAPOST (see “OPCAPOST” on page 117) with the corresponding completion code. This returns the control of the application processing to TWS. The samples contain a code template for a non-subsystem command (EVJERUX1).

Parameters Passed to a User Exit

When the request name begins with UX TWS Automation stores the complete request buffer in the &EHKVAR1 task global variable. This variable must be forwarded to the command as an input parameter, as indicated in the format description above. &EHKVAR2 contains the input arrival time.

In contrast to a subsystem operation, the request buffer for a non-subsystem operation contains the entire request in one field. The format of the request buffer for 'UX' requests is described in Table 9 on page 120.

Interaction with CICS Automation

The following example shows how to use the INGCICS command of CICS Automation to open and close CICS files. The INGCICS command allows you to perform CEMT commands on any CICS subsystem. If CICS Automation is not installed, then you can perform a similar function using the MVS MODIFY command from a NetView CLIST. First, you need these requests:

UXCICSOP	Requests CICS to open a file.
UXCICSCL	Requests CICS to close a file.

The example selects the CLIST names of CICSOPEN and CICSCLOS. Using these names, the format of the CMD attributes of the OPCACMD entry (see “OPCACMD” on page 94) is as follows:

```
Action   Keyword/Data(partial)
________ CMD
         (UXCICSOP,,'CICSOPEN &EHKVAR1')

________ CMD
         (UXCICSCL,,'CICSCLOS EHKVAR1')
```

Figure 69. OPCACMD Entry for Interaction with CICS

Figure 70 on page 130 shows the definition of the operation text and other fields in TWS.

```
--------------------------------- OPERATIONS ---------------------  ROW 1 OF 1
Command ===>                                              Scroll ===> PAGE

Enter/Change data in the rows, and/or enter any of the following
row commands:
I(nn) - Insert, R(nn),RR(nn) - Repeat, D(nn),DD - Delete
S - Select operation details
Enter the PRED command above to include predecessors in this list, or,
enter the GRAPH command to view the list graphically.

Application           : PAYMAINT        Payroll Master Update

Row  Oper      Duration Job name  Operation text
cmd  ws   no.  HH.MM
'''' NV04 015   0.01    DUMMY___  UXCICSCL CICS01 PAYROLL____
```

Figure 70. Defining Sample CICS Application in TWS

The example uses the CICS subsystem name and the file name as parameters to the request. These parameters are optional and flexible. Thus, the CICS name could also be passed through the **Job name** field. The REXX code for CICSOPEN and CICSCLOS is supplied in the samples as EVJERUX2 and EVJERUX3.

Interaction with IMS automation

The following example shows how to use the INGIMS of IMS automation to start and stop databases in IMS. The INGIMS command allows you to perform IMS MTO commands on any IMS in the system. Other IMS commands could be imbedded into IMSCMD and incorporated in NetView CLISTs you write yourself, using similar logic to that shown in the EVJERUX4 and EVJERUX5 CLISTs supplied with the samples. If IMS automation is not installed, then you can perform similar function by replying to the outstanding reply ID of the IMS you wish to communicate with from a NetView CLIST you write yourself. First, you will need these requests:

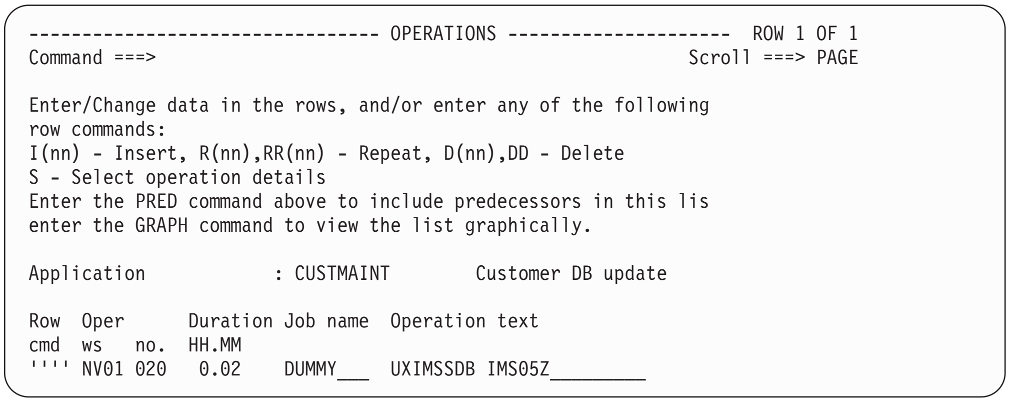
UXIMSSDB	Requests to start a database.
UXIMSPDB	Requests to stop a database.

The example selects the CLIST names EVJERUX4 and EVJERUX5. Using these names, the format of the CMD attributes of the OPCACMD entry (see "OPCACMD" on page 94) is as follows::

```
Action   Keyword/Data(partial)
________ CMD
         (UXIMSSDB,,'EVJERUX4 &EHKVAR1')

________ CMD
         (UXIMSPDB,,'EVJERUX5 EHKVAR1')
```

Figure 71. OPCACMD Entry for Interaction with IMS

Figure 72 on page 131 shows the TWS definition of the operation text and other fields.



```
--------------------------------- OPERATIONS ---------------------  ROW 1 OF 1
Command ===>                                                   Scroll ===> PAGE

Enter/Change data in the rows, and/or enter any of the following
row commands:
I(nn) - Insert, R(nn),RR(nn) - Repeat, D(nn),DD - Delete
S - Select operation details
Enter the PRED command above to include predecessors in this lis
enter the GRAPH command to view the list graphically.

Application           : CUSTMAINT       Customer DB update

Row  Oper      Duration Job name  Operation text
cmd  ws   no.  HH.MM
'''' NV01 020   0.02    DUMMY___  UXIMSSDB IMS05Z_________
```

Figure 72. Defining Sample IMS Application in TWS

The parameters of the request are the IMS subsystem name and the database name. The REXX code of EVJRUX4 and EVJERUX5 is supplied in the samples.

Chapter 15. Resynchronization and Recovery Considerations

TWS Automation combines the capabilities of two different subsystems, TWS and SA z/OS, which may reside over several systems. As a result, a failure or a scheduled interruption of services with one of the subsystems, processors, or telecommunications facilities may occur and prevent TWS Automation from processing operations. Further complications arise by shifting work load across multiple system images, either for scheduled workload balancing or as part of a recovery situation.

In such a case, a loss of synchronization can occur between the TWS schedule and the TWS Automation components in NetView. When this happens, you may need a manual process to examine the TWS schedule, to ensure that TWS Automation in NetView is performing actions as required and, in some cases, to resynchronize TWS and TWS Automation.

During a loss of contact or a failure with either TWS or NetView, TWS Automation's facilities invoke and restore the environment as it was before the failure so that event scheduling can pick up where it left off. This results in a satisfactory resolution and manual resynchronization is not required. See "Automated Recovery Functions" on page 136 for additional information.

Generally, the longer the outage, the more likely that resynchronization is required. This depends on the number of scheduled events that are not processed. A long outage during the day may have a smaller synchronization impact than a shorter outage during a period when many online facilities are started or shut down.

Examples and Scenarios

This section describes possible scenarios for resynchronization and recovery.

Loss of Contact Between TWS and TWS Automation

Under most situations, once TWS Automation re-establishes connectivity, its automatic recovery schedules requests for execution that it could not execute prior to connectivity. In some situations you may need to intervene manually, such as when the request is no longer valid. The following sections discuss several reasons for manual intervention.

Taking Action Too Late

The remaining processing window is too small to allow an operation to occur. For example, an online system may require a certain amount of time to initialize. If this amount of time is close to the scheduled shutdown time, you probably should override the request and complete the operation manually.

| You should issue `EVJESPIN CMD=RESET SUBSYSTEM=`*`subsystem`*.

Queuing Several Actions for a Specific Target Subsystem

Rather than not having enough time for a system initialization as in the previous example, the outage may have lasted long enough for a specific subsystem to receive several queued operations that frequently conflict. For example, an online task may have a start request with an ended-in-error status because of connectivity problems. This same task may also have a stop request already due for scheduling.

If you allow automatic recovery for this application, the subsystem would start, but an immediate shutdown would follow.

| Complete these operations manually. You should issue `EVJESPIN CMD=RESET`
| `SUBSYSTEM=`*`subsystem`*.

Backup on a Different Processor

If you perform a backup using a different processor, pay special attention to ensure that you properly restore the work load on the new system. Depending on the backup structure, you need to follow one of several different procedures discussed in the next sections.

Full Takeover onto a Standby System at the Same Site

This is the simplest type of backup. It becomes the same as a single-system recovery if the data is also available. TWS Automation uses the information in the status file to restore the various subsystems to the pre-backup status.

Full Takeover onto a Standby System at a Different Site

If the status file is not available, restructure the environment manually. Examine pertinent applications that control this specific NV*nn* workstation. The last completed NV*nn* operation requires manual triggering. You can achieve this with the INGOPC function, allowing the NetView operator direct access to the relevant applications. Although, at times, you may find it necessary to perform specific operations manually, in most cases, resetting an operation with EVJESPIN or restarting an application produces the desired effect.

Takeover onto a Working System

In most situations, the takeover is onto a working system and is restricted to certain critical applications. The previously discussed considerations as to whether the system is at the same site or at another site apply to this situation. Because not all applications are restarted on the backup system, several new considerations become important. Certain applications need cancelling before you can achieve a restoration of services. Some situations can result in duplications, such as with subsystems like TSO, which are frequently found on every MVS system. Although normally this is controlled by SA z/OS, TWS Automation can control this. Here, duplicate applications need cancelling for the backup period.

During the backup period, use one of these two methods to run TWS Automation:

- Add the new work load to the resident NetView by changing the NV*nn* workstation entry in the platform control file to point to the same NetView domain ID.
- Start another copy of NetView with the NetView domain ID used in the failing system.

The consideration of which method to use becomes important once you restore the original configuration.

With the single NetView solution method, you need to resolve the subsystems manually because their original identity is lost. With the extra NetView solution method, you can stop the subsystems controlled by the extra NetView by shutting it down. This simplifies the restoration process, requiring almost no manual intervention.

In both cases, restoring the environment follows similar procedures used to backup a host onto a standby backup system.

Long Term Outage

You must manually intervene when the outage duration is more than a single scheduling cycle. This type of recovery is confusing because many applications are shown as late in the TWS plan. Carefully review these applications because some of the them still need scheduling, while others need to be cancelled. For applications that need scheduling, certain operations involving the online portion need cancelling or holding. To ensure success, this type of recovery needs precise planning and monitoring. Otherwise, you can use the scenarios previously outlined.

Example Using Doubly-Defined NetView Domain IDs

The example in Figure 73 shows the WORKSTATION DOMAINS entry for a 4-processor environment:

```
Code 1        Code 2        Code 3        Value Returned
NV00                                      NVTOR
NV01                                      XBAOF
NV02                                      AOFT5
NV06                                      AOFS6
```

Figure 73. Mapping of NVxx Workstations to Domain IDs

Here, NV00 maps to the NVTOR NetView domain ID, and the NV01 workstation maps to the XBAOF NetView domain.

Under normal circumstances, each NetView domain ID represents a processor with its MVS operating system and a unique NV*xx* general automatic reporting workstation. For situations such as testing, backup, or work load management, this relationship needs no maintenance.

In the previous example, both NV00 and NV06 TWS-defined workstations represent their own specific NetView domain, NVTOR and AOFS6, respectively.

Assume that the system represented by NVTOR has failed, and you make the decision to shift the work load to the AOFS6 system. You can accomplish this by changing the domain ID in the first CODE statement from NVTOR to AOFS6. This would imply that the AOFS6 domain is associated with two TWS workstations, namely NV00 and NV06.

If you accomplish this change without altering the TWS definitions, you must reload the SA z/OS control file. The scheduler or operator needs to ensure that the TWS-defined applications that are running in the failed system are restarted on the backup system. Because the SA z/OS status records are on the failed system, the scheduler manually recovers the failed environment. Once resynchronization completes, any new scheduled event originally intended for the NetView domain ID NVTOR automatically is scheduled for AOFS6. After you resolve the problem on the NVTOR system, perform the previous scenario in reverse order to restore the system to its original configuration.

When double definitions of this type are used, exercise caution to avoid creating conflicting requests for specific subsystems. For example, if RMF exists in the AOFS6 domain, TWS can then schedule a shutdown request on NV00 and a start request on NV06.

Automated Recovery Functions

Only a small portion of TWS Automation resides in the TWS address space in TWS user exits. These exits communicate to the rest of TWS Automation that resides in the NetView address space. A loss of contact results if a NetView address space becomes unavailable or if TWS Automation code in NetView is unavailable. Also, a communication failure can prevent a request from reaching its ultimate destination.

TWS Automation automated recovery determines which operations are affected by a specific loss of communications. It also determines when the connectivity and availability of a given target NetView is corrected and the NV*nn* operation is reset to the ready state. This redrives the EQQUX007 exit, allowing it to re-create the original request.

TWS Actions in a Loss-of-Contact Situation

The EQQUX007 exit or an intermediary NetView with an ended-in-error status and a return code of Sxxx reports a connectivity loss to TWS. TWS does not schedule any dependent operations and shows an error on the operations ended-in-error Status Display Facility panel. TWS takes no further action until the connectivity is restored and TWS Automation automatic recovery is invoked.

The operator or scheduler can manually override the ended-in-error status, thus allowing the application to continue or cancelling it.

TWS Automation Actions in a Loss-of-Contact Situation

If loss of connectivity to NetView is detected in the TWS Automation portion of the EQQUX007 exit (using the EQQUSINT function directly), then TWS Automation posts an ended-in-error status with a UNTV return code. TWS Automation uses this mechanism because the EQQUX007 exit cannot directly modify operation status.

If the request is received in NetView, but TWS Automation cannot propagate the request to the appropriate target system, TWS Automation uses the OPCAPOST function to post the operation as ended-in-error with an S999 return code.

Appendix. Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
USA

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Websites are provided for convenience only and do not in any manner serve as an endorsement of those Websites. The materials at those Websites are not part of the materials for this IBM product and use of those Websites is at your own risk.

© Copyright IBM Corp. 1990, 2010

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Deutschland Research & Development GmbH
Department 3248
Schoenaicher Strasse 220
D-71032 Boeblingen
Federal Republic of Germany

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

IBM, the IBM logo, and ibm.com® are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at www.ibm.com/legal/copytrade.shtml.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Microsoft, and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

Glossary of Terms

The intent of this glossary is to define terms as TME 10 OPC uses them. However, where applicable, terms are taken from the *IBM Dictionary of Computing*, New York; McGraw-Hill, 1994. These terms are marked by an asterisk (*). Unless otherwise noted, the definitions below apply equally well to OPC/ESA and TME 10 OPC.

A

actual duration. At a workstation, the actual time in hours and minutes it takes to process an operation from start to finish.

APAR. Authorized program analysis report. A report of a problem caused by a suspected defect in a current unaltered release of a program.

all workstations closed. A user defined interval during which *all* TWS's workstations are not available for running applications under TWS's control.

Note: All the workstations could be either shut down *or* simply not available to TWS.

application. (1) A group of related operations performed together to satisfy a specific end user task. (2) A measurable and controllable unit of work that completes a specific user task such as the running of payroll or financial statements. The smallest entity that an application can be broken down into is an operation. Generally, several related operations make up an application.

application description. A database description of an application.

application ID. The name of an application. Examples: Y1976, Payroll.

arrival (A). Status of an operation that indicates it is waiting for the input to arrive before processing.

authority. The ability to access a protected resource.

authority group. A name used to generate a RACF resource name for authority checking.

automatic events. Events recognized by or triggered by an executing program. Automatic events are usually generated by TWS job tracking programs but may also be created by a user-defined program.

automatic reporting workstation. A workstation that reports events (the starting and stopping of operations) in real time to TWS, such as a processor or printer.

automatic job recovery. a TWS function that allows you to specify, in advance, alternative recovery strategies for applications or operations ended in error.

availability. * The degree to which a system (and in TWS, an application) or resource is ready when needed to process data.

B

batch loader. a TWS batch program you can use to create and update information in the application description and operator instruction databases.

bracketed DBCS. A MIXED format field consisting of a DBCS part only, that is, DBCS characters enclosed by a shift-out/shift-in control character pair.

browse. An ISPF/PDF dialog function that manages data for display only. This function lets the user view but not change data.

C

CP. Current plan.

calendar. The data that defines the operation department's processing schedule in days and periods.

capacity. The actual number of parallel servers and workstation resources available during a specified open time interval.

capacity ceiling. The maximum number of operations a workstation can handle simultaneously.

case code. A code in the automatic job recovery function that represents a group of abend codes or return codes. Any code in the JOBCODE and STEPCODE parameters is considered a potential case code if defined as such in the case code macro.

closed workstation. A workstation that is unavailable to process work for a specific time, day, or period.

command. * A request from a terminal for the performance of an operation or the execution of a particular program. A character string from a source external to a system that represents a request for system action.

complete. Status of an operation indicating that it has finished processing.

© Copyright IBM Corp. 1990, 2010

completion code. a TWS system code indicating how the processing of an operation ended at a workstation.

complex of processors. A JES2 multi-access spool system or a JES3 system with more than one processor.

computer workstation. A workstation that performs MVS processing and usually reports status to TWS automatically. A processor when used as a workstation. It can refer to single processors or multiprocessor complexes serving a single job queue (for example JES2 or JES3 systems).

controller. The portion of TME 10 OPC or OPC/ESA that runs on the controlling processor and contains the tasks that manage TWS databases and plans.

critical path. The route within a network with the least amount of slack time.

current plan. A minute by minute schedule of each operation of an application. It reflects the current state of the operating environment showing the status of work completed and work still to be done.

current schedule. The database that contains the current plan information.

cyclic interval. The number of days in a cyclic period.

cyclic period. A period with a specific origin date and set frequency. A cyclic period can be broken down into two types:

- Those that include work and free days
- Those that include only work days.

Cyclic periods must always represent a fixed time period in days. For example, week (7 days).

D

daily plan. A set of plans that shows work that the operations department does on a particular day or shift. A list by day and application of all operations to be performed within the operations department.

default calendar. (1) A calendar that you have defined for TWS to use when you do not specify a calendar in an application description. (2) A calendar that TWS uses if you have neither specified a calendar in an application description, nor defined your own default calendar.

deadline. See deadline date and deadline date.

deadline date. The latest date by that an occurrence must be complete by.

deadline time. The latest time by that an occurrence must be complete by.

defined. An open day status that indicates that specific open time intervals exist for a workstation on a particular day.

dependency. A relationship between two operations where the first operation must successfully finish before the second operation can begin.

dialog. The user's online interface with TWS.

displacement. A number specifying 'Number of Days from Period Start' or 'Number of Days from Period End'. Sometimes called offset. See offset.

duration. The time an operation is active at a workstation.

E

edit. An ISPF/PDF dialog function that is used for editing text, collecting data, and modifying data.

end user. A person who uses the services of the data processing center.

ended in error (E). The TWS reporting status for an operation that has ended in error at a workstation.

error code. The system completion code or program return code for automatic reporting workstations. The code entered by the workstation operator for manually reporting workstations.

exclusive. The state of a special resource indicating that it is fully used by one operation and cannot be used simultaneously by other operations.

exclusive resource. A workstation resource that is solely used by one operation and cannot be shared with other operations.

expected arrival time. The time when an operation is expected to arrive at a workstation. It may be calculated by daily planning or specified in the long-term plan.

extend current period. a TWS function that allows the user to extend the current plan up to a maximum of 504 hours (21 days) from the current end date.

external dependency. A relationship between two occurrences where an operation in the first occurrence must successfully finish before an operation in the second occurrence can begin processing. See dependency.

external predecessor. The name given to the operation in the first occurrence of an external dependency that must finish before its external successor can begin processing.

external successor. The name given to the operation, in the second occurrence of an external dependency, that cannot begin until its external predecessor completes.

F

free day. A nonworking day.

free day rule. A rule that determines how TWS will treat free days when the application run day falls on a free day. The rule is as follows:

- **Excluded**: Free days excluded; only work days are taken into account.
- **Included**: Free days included; all days are taken into account, as follows:
  - **(1)** Run before the free day.
  - **(2)** Run after the free day.
  - **(3)** Run on the free day.
  - **(4)** Do not run on the free day.

G

general workstation. A workstation where activities, usually manual, and other than printing and processing, are carried out. Manual activities might be data entry or job setup. A general workstation reporting to TWS is usually manual, but can be automatic.

generic search argument. A portion of a key containing a generic search character that in TWS is an asterisk (*) or percent sign (%). The asterisk represents any string of characters and the percent sign any single character. Use with any portion of a key to search the database for items to be displayed as part of a listing. Examples: %ABC, A*C, A*.

H

host processor. * A processor that controls all or part of a user application network. * In a network, the processing unit that the access method for the network resides in.

highest return code. A numeric value from 0 to 4095. If this return code is exceeded during a job's processing, the job will be reported as ended in error.

I

incident log. An optional function available under the job completion checker.

input arrival. The user-defined date and time an operation or an application becomes ready for processing.

internal dependency. A relationship between two operations within an occurrence where the first operation must successfully finish before the second operation can begin.

internal predecessor. The name given to the operation of an internal dependency that must finish before its internal successor can begin processing.

internal successor. The name given to the operation of an internal dependency that cannot begin until its internal predecessor completes processing.

ISPF. Interactive System Productivity Facility.

interrupted (I). a TWS reporting status for an operation indicating that the operation has been interrupted while processing.

J

job. * A set of data that completely defines a unit of work for a computer. A job usually includes all necessary computer programs, linkages, files, and instructions to the operating system. In TWS, an operation performed at a CPU workstation.

job completion checker (JCC). An optional function of TWS that provides an extended checking capability of the results from CPU operations.

job control language (JCL). * A problem-oriented language designed to express statements in a job that are used to identify the job or describe its requirements to an operating system.

JES. Job Entry Subsystem.

job entry subsystem (JES). * A system facility for spooling, job queuing, and managing I/O.

job setup. The preparation of a set of JCL statements for a job at a TWS workstation you defined for this purpose.

job submission. a TWS process that presents jobs to MVS for running on a TWS defined workstation at a time specified in the daily plan.

JS. The JCL repository data set.

K

keyword. * A symbol that identifies a parameter. * A part of a command operand that consists of a specific character string (such as DSNAME=).

keyword parameter. * A parameter that consists of a keyword, followed by one or more values.

L

LTP. Long-term plan.

last operation. (1) An operation in an occurrence that has no internal successor. (2) The terminating node in a network.

latest start. The latest start day and time (calculated by TWS) for an operation that will allow all occurrences to meet their deadline.

layout ID. A unique name that identifies a specific ready list layout.

limit for feedback. See feedback limit.

local. * Synonym for channel-attached.

local processor. * In a complex of processors under JES3, a processor that executes users' jobs and that can assume global functions in the event of failure of the global processor. In TWS, a processor in the same installation that communicates with the controlling TWS processor through shared DASD communication.

long-term plan. A high-level schedule of processing activities for the forthcoming weeks and months. The scope of a long-term plan can be from one day to four years.

The long-term planning function produces a list of application occurrences identified by name, date, and run time for a specified planning period.

M

manual reporting workstation. A type of workstation reporting where events, once they have taken place, are manually reported to TWS. This type of reporting requires that some action be taken by a workstation operator. Manual reporting is usually performed from a list of ready operations.

mass updating. A function of the application description dialog where a large update to the application database can be requested.

modify current plan. a TWS dialog function used to dynamically change the contents of the current schedule to respond to changes in the operation environment. Examples of special events that would cause alteration of the current schedule are: a rerun, a deadline change, or the arrival of an unplanned application.

most critical application occurrences. Those unfinished applications that have a latest start time that is less than or equal to the current time.

N

node. * In a network, a point where one or more functional units interconnect transmission lines.

noncyclic period. A period that has a varying frequency that you must define each origin date for. Examples: month, payroll period, and quarterly.

nonreporting. A reporting attribute of a workstation that indicates that information is not fed back to TWS.

O

OPC/ESA. Operations Planning and Control/Enterprise Systems Architecture

occurrence. Each instance of an application in the long-term plan and current plan is called an occurrence.

An application occurrence is one attempt to process that application. Occurrences are distinguished from one another by run date, input arrival time, and application ID. For example, one application that runs four times a day is said to have four occurrences a day.

offset. A maximum of 12 positive and 12 negative values in the ranges 1 through 999 and –1 through –999 that indicate which days of a calendar period an application shall run on. See displacement.

TWS host. The processor where TWS updates the current plan database.

TWS local processor. A processor that connects to the TWS host or remote processor through shared event data sets.

open time interval. The time interval during which a workstation is active and can process work.

operation. An operation is a unit of work that is part of an occurrence and is processed at a workstation.

operation waiting for arrival. The status of an operation that indicates that the necessary input has not arrived at a workstation so that the operation can begin processing. This status is applicable only for operations without predecessors.

operation status. The status of an operation at a workstation.

An operation's status can be one of the following:

A	Waiting for input to arrive.
R	Ready for processing. All predecessors are complete.
*****	Ready for processing. There is a nonreporting predecessor. All predecessors are complete but one or more predecessors were executed at a nonreporting workstation.

S	Started.
I	Interrupted operation.
C	Complete.
E	Operation ended in error.
W	Waiting for predecessor to complete.
U	Undecided. The status is not known.

operator. * (ISO) A symbol that represents the action to be performed in a mathematical operation. * In the description of a process, that which indicates the action to be performed on operands. * A person who operates a machine.

option. A selection item on a menu panel in the TWS dialog.

origin date. The date that a period (cyclic or noncyclic) starts on.

P

panel. * A particular arrangement of presentation windows used to show information to the user. TWS uses only fixed-format panels.

parallel operations. Operations at workstations that are not dependent on one another and therefore can be performed simultaneously.

parallel server. The function that processes operations at a workstation, especially when there is more than one such function. See server.

parameter. * (ISO) A variable that is given a constant value for a specified application and that may denote the application. * A name in a procedure that is used to refer to an argument passed to that procedure.

pending application description. An application description that is incomplete and not ready for use in planning or scheduling.

period. A business processing cycle. A time period defined in the TWS calendar. They are used to describe when, and how often, applications are to run.

period name. A name of a period. Examples are week, month, quarter and fiscal period end.

period type. Periods are of two types: cyclic or noncyclic.

PDF. program development facility.

predecessor. An operation of an internal or external dependency that must finish successfully before its successor operation can begin.

printout routing. The ddname of the daily planning printout data set.

print workstation. A workstation that prints output and usually reports status to TWS automatically.

priority. A digit from 1 to 9 (where 1 = low, 8 = high, and 9 = urgent) that determines how TWS schedules applications to run. A number from 1 (low priority) to 9 (high priority) that establishes the importance of an application relative to other applications.

processor. * (ISO) In a computer, a functional unit that interprets and executes instructions. * A functional unit or part of another unit (such as a terminal or a processing unit) that interprets and executes instructions.

program interface. a TWS interface that allows a user-written program to issue various types of requests to the TWS subsystem.

Q

QCP. Query current plan.

R

RACF. Resource Access Control Facility.

read authority. A type of access authority that allows a user to read the contents of a data set, file, or storage area, but not to change it.

ready (R). The status of an operation indicating that predecessor operations are complete and that the operation is ready for processing.

ready list. A display list of all the operations ready to be processed at a workstation. Ready lists are the means by which workstation operators manually report on the progress of work.

recovery. See automatic job recovery.

remote processor. A processor connected to the TWS host processor by a VTAM network.

remote job tracking. The function of tracking jobs on remote processors connected by VTAM links to a TWS controlling processor. This function enables a central site to control the submitting, scheduling, and tracking of jobs at remote sites.

replan current period. a TWS function that recalculates planned start times for all occurrences to reflect the actual situation.

reporting attribute. A code that specifies how a workstation will report events to TWS.

rerun. a TWS function where an application or part of an application that ended in error can be run again.

rescale factor. A value from 0 to 100 used to reduce the new duration value by a given percentage amount.

return code. An error code issued by TWS for automatic reporting workstations.

row command. A dialog command used to manipulate data in a table.

run cycle period. A time frame defining the effective period and run days of a calendar period.

run day. The date that an application is to run on. It is expressed as a number relative to the start or the end of a run cycle period.

S

SAF. System Authorization Facility.

search argument. A value that is used to search the database for an item that is to be part of a displayed listing.

selection criteria. Search arguments entered on a list criteria panel in the dialog that limit the contents of a listing.

server. A program or device set up for a workstation to perform a service for that particular type of workstation. For example, an initiator is a server for a computer workstation. A printer is a server for a print workstation.

service functions. Functions of TWS that let the user deal with exceptional conditions such as investigating problems, preparing APAR tapes, and testing TWS during implementation.

shared DASD. Direct access storage device that can be accessed from more than one processor.

shared resource. A special or workstation resource that can be used simultaneously by more than one operation while the operation is processed at a workstation.

slack. Used to refer to ‘spare’ time. Can be calculated for the critical path by taking ‘Deadline less the Input Arrival less the Sum of Operation Durations’.

smoothing factor. A value between 0 and 100 that controls the extent to which actual durations are fed back into the application description database.

SMP. System Modification Program.

special resource. Resources that are not associated with a particular workstation but are needed to process work there.

splittable. Refers to an operation that can be interrupted while processing at a workstation.

standard. User specified open time intervals for a typical day at a workstation.

status. The current state of an operation or an occurrence.

started (S). a TWS reporting status of an operation or an application indicating that an operation or an occurrence is started.

submit/release data set. A data set shared between the TWS host and a local TWS processor that is used to send job stream data and job release commands from the host to the local processor.

subresources. A set of resource names and rules for the construction of resource names. TWS uses these names when checking a user’s authority to access individual TWS records.

subsystem. * A secondary or subordinate system, usually capable of operating independently of, or asynchronously with, a controlling system.

successor. An operation in an internal or external dependency that cannot begin until its predecessor completes processing.

sysout class. * An indicator used in data definition statements to signify that a data set is to be written on a system output unit. It applies only to print workstations.

T

temporary operator instructions. Operator instructions that have a specific time limit during which they are valid. They will be displayed to the workstation operator only during that time period.

TME 10 OPC. TME 10 Operations Planning and Control

tracker. The portion of TME 10 OPC or OPC/ESA that runs on every system in your complex. It acts as the communication link between the MVS system that it runs on and the controller.

tracking event log. A log of job tracking events and updates to the current schedule.

transport time. The time allotted for transporting materials from the workstation where the preceding operation took place, to the workstation where the current operation is to occur.

TSO. Time Sharing Option.

time zone support. A feature of TWS that allows applications to be planned and run with respect to the local time of the processor that runs the application. Some networks may have processors in different time zones. The controlling processor will make allowance for differences in time during planning activities, for example the input arrival time of predecessor applications, to make sure that interacting activities are correctly coordinated.

turnover. A subfunction of job tracking that is activated when job tracking creates an updated version of the current schedule.

U

undecided (U). a TWS reporting status for an operation or an application indicating that the status is not known.

update authority. Access authority given to a user by RACF to use the ISPF/PDF edit functions of the TWS dialog. Access authority to modify a master file or data set with the current information.

V

validity period. The time interval defined by an origin date and an end date within which a run cycle or an application description is valid.

versions. Applications with the same ID but different validity dates.

VSAM. Virtual Sequential Access Method.

VTAM. Virtual Telecommunication Access Method.

W

waiting (W). a TWS reporting status (for an application) indicating that it is waiting for a predecessor operation to complete.

waiting list. A list of submitted jobs that are waiting to be processed.

work day end time. The time at which TWS will consider a work day to have ended when that work day immediately precedes a free day. For example, if you specify Saturday to be a free day, you could specify 08.00 hours. Saturday morning as the end of Friday's work day. TWS can then plan work to be done from 00.00 to 08.00 Saturday morning, as if that time was actually part of Friday.

workstation. A unit, place, or group that performs a specific data processing function. A logical place where work occurs in an operations department.

TWS requires that you define the following characteristics for each workstation: the type of work it does, the quantity of work it can handle at any particular time, and the times it is active. The activity that occurs at each workstation is called an operation.

workstation description database. a TWS database containing descriptions of the workstations in the operations department.

workstation resources. Limited resources defined for each workstation that an operation requires a certain amount of to process work.

workstation type. Each workstation can be one of three types: computer, print, or general.

work day. A day that applications can normally be scheduled to start on.

Index

Program Number: 5698-SA3

Printed in USA

SC34-2579-00