CA VM:Schedule™

Administration Guide

Version 2.0, First Edition

9/25/2013

CA Technologies Product References

This document references the following CA Technologies products:

- CA Mainframe VM Product Manager™
- CA VM:Batch™
- CA VM:Secure™
- CA VM:Backup™

Contact CA Technologies

Contact CA Support

For your convenience, CA Technologies provides one site where you can access the information that you need for your Home Office, Small Business, and Enterprise CA Technologies products. At http://ca.com/support, you can access the following resources:

- Online and telephone contact information for technical assistance and customer services
- Information about user communities and forums
- Product and documentation downloads
- CA Support policies and guidelines
- Other helpful resources appropriate for your product

Providing Feedback About Product Documentation

If you have comments or questions about CA Technologies product documentation, you can send a message to techpubs@ca.com.

To provide feedback about CA Technologies product documentation, complete our short customer survey which is available on the CA Support website at http://ca.com/docs.

Documentation Changes

CA VM:Schedule 2.0, First Edition, 9/25/2013

- Global change—All references to the user ID for system administrator and maintenance functions changed from VMRMAINT to VMANAGER.
- Global change—Names and branding of related products were updated.
- Format Conventions for Code Syntax (see page 11)—Changed to new common standards.
- CA VM:Schedule Single System Image Mode (see page 103)—New chapter for new SSI mode feature.
- SSI: Activating Single System Image Mode Operation (see page 46)—New configuration file record to activate SSI mode.
- VMDRPT Utility (see page 78)—Added SYSNAME field.
- Database Utilities (see page 111)—Added new utilities. VMDCHSYS bulk-renames SYSNAME fields. VMDDBEXT extracts records from the database of requests.
- CMDCHECK User Exit (see page 95)—New user exit.
- RESID: Defining the CA VM:Schedule Resource ID (see page 67)—New configuration file record, which sets a resource ID.
- About CA VM:Schedule (see page 19)—Updated the section.
- Administration Overview (see page 27)—Updated information about passwords. Documentation reflects deletion of the AUTOLOG user exit and the addition of the CMDCHECK user exit.
- DUMP: Directing Dumps (see page 56)—Correction of DUMP record.
- Using Non-CA VM:Secure Security Packages (see page 101)—Chapter rewritten. New version explains how to set up CA VM:Schedule to XAUTOLOG virtual machines without passwords.
- Using CA VM:Schedule User Exits (see page 93)—Changes to return codes and operation of user exits.

Contents

Chapter 1: About This Book

The *CA VM:Schedule Administration Guide* contains information for configuring and administering the product. It contains information on configuration file records, user exits and utilities.

This document assumes you are familiar with the following:

- IBM z/VM, CP, and CMS standards, conventions, and operation
- The daily requirements of your VM environment

Note: For more information about using *CA VM:Schedule* commands see the *User Guide*. For more information on using *CA VM:Schedule* operator functions, see the *Operator Guide*. These books describe how to perform common tasks using the CA VM:Schedule screens and line–mode commands.

This section contains the following topics:

Format Conventions for Code Syntax

This section describes the format and conventions used to document commands, utilities, and user exits. Each convention provides examples, describing how to use commands, how to use options, or how the system responds to user entries.

> **Note:** The examples and instructions throughout this document use VMANAGER as the user ID for system administrator and maintenance functions. Also, this document uses VMSCHED as the user ID for the service virtual machine (svm). These user IDs are the default values. If you use non-default user IDs for the system administrator or the svm, replace the default values throughout this document with the values that you use.

Command Abbreviations

> When a command contains uppercase and lowercase letters, then the uppercase letters denote the shortest acceptable abbreviation that you can use to type the command. However, when a code item appears entirely in uppercase letters, you cannot abbreviate the item.
>
> You can type the code item in uppercase letters, lowercase letters, or any combination.

Example:

```
CMDNAme
```

In this example, you can enter CMDNA, CMDNAM, or CMDNAME in any combination of uppercase and lowercase letters.

Continuation

The code syntax or code fragment definitions can continue from one line to the next line. The following examples describe code continuation:

Example 1:

```
A | B C | D
```

This code is equivalent to the following code:

```
A
| B C
| D
```

Example 2:

```
{choice1 | choice2 | choice99}
```

This code is equivalent to the following code:

```
 { choice1
 | choice2
 | choice99 }
```

Default Values

An underlined code item denotes the default value. The system uses the default value unless you override it. You can override the default value by coding an option from the available list.

Example:

[parm1 | *parm2* | *parm3*]

In this example, the code item *parm1* is the default value, and this is used by the system when you do not specify any of the options. However, you can code *parm1*, *parm2*, or *parm3.*

Keywords and Constants

A keyword name or constant always appears in uppercase letters. Code the keyword name or constant exactly as shown in the following example:

Example:

STOP {*tracenumber* | * | [USER] *userid*}

This example displays the USER keyword.

Optional Choices

Defines optional code items—denoted by square brackets around a code item.

Example:

```
CMDName [parm1]
```

In this example, you can choose *parm1* or no parameter at all. However, when two or more items are enclosed in square brackets and separated by vertical bar characters, all of them are optional.

Multiple Optional Choices

When two or more items are enclosed in square brackets and separated by vertical bar characters, all of them are optional.

Example:

In this example, you can choose *parm1, parm2, parm3,* or nothing at all.

```
[parm1 | parm2 | parm3]
```

Positional Parameters

Commands with positional parameters are identified by nested square brackets. Each positional parameter requires the specification of all previous positional parameters. The following example describes the positional parameter:

Example:

```
CMDName [posparm1 [posparm2 [posparm3]]]
```

In this example, *posparm3* can be specified only when *posparm1* and *posparm2* are also specified.

Repeatable Choices

A list of code items enclosed in square brackets and followed by an ellipsis means that you can select more than one item or, in some cases, repeat a single item.

Example:

```
[value1 | value2 | value3] ...
```

In this example, you can choose a single value, more than one value, or none of the values.

Repetition

An ellipsis following a code item means that the code item can be repeated.

Example:

```
Repeat…
```

Required Choices

You must select one item from a list of items when they are enclosed in curly braces. The items are separated by a vertical bar character.

Example:

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

In this example, your choice results in CMDNAME A, CMDNAME B, or CMDNAME C.

Special Symbols

The following list describes the meaning of the special symbols used in codes:

- {} (encloses a list of operands, one of which is required).
- [] (enclose an optional operand or operands).
- " " (enclose the name of a syntax fragment)
- underline (identifies a default value)
- | (separates alternative operands)
- . . . (Indicates that the preceding item or group can be repeated).

Symbols

The following list displays symbols. These symbols should be coded exactly as they appear in the code syntax.

- * (Asterisk)
- : (Colon)
- , (Comma)
- = (Equal Sign)
- — (Hyphen)
- ( )(Parentheses)
- . (Period)

Syntax Fragments

Some codes use fragments, when the code syntax is too lengthy. The fragment name appears between double quotes in the code syntax.

The expanded fragment appears in the syntax after all other parameters or at the bottom of the code syntax. A heading with the fragment name identifies the expanded fragment.

Example:

```
CMDName "Parms"
Parms:

[A | B | C]
```

In this example, the fragment is named "Parms", and the expanded fragment appears at the bottom of the code syntax.

System Response

Uppercase characters represent system responses or prompts.

Example:

```
ENTER YOUR LOGON PASSWORD:
```

This example displays a system response.

User–Entered Commands or Records

User-entered commands are shown in lowercase letters even though you can enter commands in either upper or lower case.

Example:

```
vmsecure addentry writers tcom (noformat nowait
```

This example shows what a user-entered command looks like.

In this example "writers tcom" is the file name and file type of the directory entry you are adding.

However, if the entry is a record, it appears in uppercase letters.

Example:

```
ACCESS DRCT 1B0 U
```

This example shows a configuration record that is entered by a user.

Variables

Lowercase items in italics denote variables.

Example:

CMDNAme *varname*

In this example, *varname* represents a variable that you must specify when you code the command CMDNAME.

Related Documentation

Several guides that are associated with *CA VM:Schedule* make up the *CA VM:Schedule* library. Each guide addresses a different type of user depending on the tasks the user must perform. The *CA VM:Schedule* library includes the following books:

- *CA VM:Schedule User Guide* - presents concepts and procedures for tasks that end users typically perform. This guide also includes end-user commands and the format of those commands.
- *CA VM:Schedule Operator Guide* - presents concepts and procedures relevant to the day-to-day operation of *CA VM:Schedule*. The book also includes operator commands and the formats for those commands.
- *CA VM:Schedule Message Reference Guide* - provides the following information:
  - Lists all messages that *CA VM:Schedule* produces
  - Provides possible cause of the situation that generated the message, and any actions to take in response to the situation
  - Contains a cross-reference that lists the first 60 characters of each *CA VM:Schedule* message in alphabetical order.
- *CA VM:Schedule Administration Guide* - provides the following information:
  - Explanations of *CA VM:Schedule* and its administration
  - Instructions for customizing *CA VM:Schedule* to fit site-specific requirements
  - Descriptions of any special administrative operations that must be performed.
- *CA VM:Schedule Installation Guide* - gives information about installing and maintaining *CA VM:Schedule*.

Other CA Technologies product guides that are referenced in this book and to which you can refer include:

- *CA Mainframe VM Product Manager Reference Guide*
- *CA Mainframe VM Product Manager Interface Guide*
- *CA Mainframe VM Product Manager Generalized Report Writer Reference Guide*
- *CA VM:Batch User and Group Manager Guide*
- *CA VM:Batch Operator Guide*
- *CA VM:Batch Administration Guide*

Chapter 2: About CA VM:Schedule

The *CA VM:Schedule* product is a component of the *CA VM Manager* system management suite for z/VM or z/Linux. *CA VM:Schedule* is a comprehensive scheduling tool for system administration, operations, and end-user jobs.

A scheduled request consists of an EXEC, program, or command plus the scheduling and monitoring instructions that go with it. *CA VM:Schedule* requests can schedule the following actions:

- Print files
- Compile programs
- Issue CP and CMS commands
- Run other types of programs

Users can schedule requests to run once or to repeat at regular intervals. For example, an executable can be scheduled to run at the following times and intervals:

- At 1:00 p.m. this coming Friday
- At 6:00 a.m. every business day
- On the last day of every month

Users can schedule requests to run on their own or other user IDs. They can also schedule EXECs to run on the *CA VM:Schedule* service virtual machine.

When it is time for a request to run, *CA VM:Schedule* verifies whether the user is logged on. If the user is logged on, *CA VM:Schedule* reminds the user to log out. Once the user is logged off, *CA VM:Schedule* logs that user ID on in disconnected mode. (This type of logon is called "autologging".) The request then runs as if the user entered the command directly from the terminal. *CA VM:Schedule* monitors and records what happens during processing. When the request runs for the last time, or when an error occurs during processing, *CA VM:Schedule* sends a notification file to the user's reader.

Your site can use *CA VM:Schedule* to set up repeated system work in advance. Such work includes the following tasks:

- Backup jobs
- System administration tasks
- Performance and accounting reporting

Users can schedule time-consuming tasks, such as compiling programs, to run when they are out of the office. This way, the requests do not interrupt other work.

This section contains the following topics:

Authorizations

The person who performs the administration tasks in this book must have the following access rights:

- To run utilities:
  - Log on to the VMSCHED service machine or be able to SUSPEND or END the product.
  - Get access to its disks as needed (read only or write).
- To administer the product and requests:
  - USER and OPERATOR authorization.

Administering VM:Schedule

As a *CA VM:Schedule* system administrator, you have two types of tasks to perform:

- Updating configuration and running utilities. You perform these tasks from the CMS command line, on VMSCHED or another user ID.
- Executing user or operator commands. You execute these commands with the VMSCHED command, on a user ID other than VMSCHED.

Configuring CA VM:Schedule And Running Database Utilities

You have several different ways to configure the product and run database utilities. The way that you use is related in part to the status of *CA VM:Schedule* and your authorization. The following table describes the relationships between performing administration tasks, product status, and authorization.

***CA VM:Schedule* Status**	**Administration Task**	**Notes**
Down	Manually edit configuration file records and run utilities.	You must be able to log on to the VMSCHED svm. Otherwise, you must gain the appropriate access to the config disk (usually VMSCHED 191) and the database disk (usually VMSCHED 1B0). The appropriate access depends on the task that you want to do.
Running	Edit configuration file records and end or reinitialize the product. Use the VMADMIN command with the CONFIG, END, or REINIT function.	For more information about VMADMIN, see the *CA Mainframe VM Product Manager* documentation.
Suspended (SUSPEND command)	Run utilities on the database of requests.	You must LINK to the database disk. For most use cases, you must also make a stable copy of the VMSCHED IRBDB file. The VMDRPT report utility does not need this copy in some use cases. **Warning:** Do not change the VMSCHED IRBDB file without ending and reinitializing the product.

Executing User and Operator Commands

To execute *CA VM:Schedule* commands, access the product with authorization as a USER or OPERATOR. You can access *CA VM:Schedule* in full-screen menus or in line mode. The screens are easier to use, especially if you are unfamiliar with *CA VM:Schedule* or you do not schedule requests often. The line-mode commands let you use *CA VM:Schedule* within EXECs or from a program.

To open the full-screen menu for operator commands, enter this command from CMS:

```
vmsched oper
```

To open the full-screen menu for user commands, enter this command from CMS:

```
vmsched user
```

Note: VMSCHED is the default name of the *CA VM:Schedule* service virtual machine. If the *CA VM:Schedule* service virtual machine at your site has a different name, use that name instead.

Note: When you first access *CA VM:Schedule*, a copyright screen can appear. To clear the screen, press ENTER or wait 15 seconds. To prevent this screen from reappearing every time *CA VM:Schedule* is used, erase or rename the VMSCHED COPYRGHT file on the public minidisk. After the copyright screen clears, the main menu for your user level displays.

CA VM:Schedule displays the main menu for your user level. Type the number of the task you want to perform and press ENTER. If you prefer, you can enter the command name or its abbreviation instead. The product displays a fill-in-the-blank screen for that task. Where appropriate, default values for fields are shown. When you complete the task, press PF12 to submit your instructions. If you forget to fill in a required field, *CA VM:Schedule* prompts you to enter the information.

If you want further explanation for any item on a screen, press Tab until your cursor is on the item. Then press PF1. *CA VM:Schedule* displays an explanation.

To leave *CA VM:Schedule* full-screen menu processing, press PF3 from the command screens. You return to the main menu. Press PF3 again until you return to CMS.

For more information about full-screen menus, see the *User Guide*.

To execute a command in line mode, enter the name of the service virtual machine (default is **VMSCHED**) followed by the command name and any operands. When a line mode command completes, a return code is presented to the issuing user.

For example, to transfer the ACCOUNTS request from user ALICE to user LOUISE through a line-mode command, enter from CMS:

```
vmsched transfer accounts alice louise
```

Any error messages are edited according to your virtual machine's current EMSG setting.

The complete formats for operator line-mode commands are listed in the *CA VM:Schedule Operator Guide*. The complete formats for user line-mode commands are listed in the *CA VM:Schedule User Guide*.

Help with CA VM:Schedule Commands

You can review explanations of *CA VM:Schedule* commands at your terminal. To get help for commands, enter:

```
help vmsched menu
```

A help menu displays. Move the cursor to the command you need help with and press ENTER.

CA VM:Schedule Messages

While using *CA VM:Schedule,* you will receive messages that do one of the following:

- Inform you of the status and progress of the function or task
- Prompt you for information about your system
- Indicate if an error occurred
- Provide instructions for resolving errors

Example: Command Too Long Error

The following message appears when a command that exceeds the 256 character limit is entered:

```
VMDCOM1091E COMMAND LENGTH CANNOT EXCEED 256 CHARACTERS
```

Like all messages, this message consists of a message identifier (VMDCOM1091E) and a message body. The message identifier in turn consists of several parts, encoding the following information:

VMD

The product code

COM

The routine or module that issued the message

1091

The message number

E

The severity code, in this case meaning "error"

Severity Codes

You can tell the type of message by its last letter, which is its *severity code*. The severity code can be one of the following:

A

Action message

You need to perform an action, for example, move the cursor or enter information.

E

Error message

The function or task can continue normally if you correct the error. The function or task may end.

I

Information message

CA VM:Schedule initialization has completed or a task that *CA VM:Schedule* is executing has completed. When an informational message appears immediately after a severe error message (severity code S), it tells you how to correct the problem that the severe error message identified.

R

Response message

CA VM:Schedule requests information. The function or task continues after you supply the required information.

S

Severe error message

An error preventing the function or task from continuing has occurred. You are returned to CMS and must restart the procedure after correcting the error.

T

Terminal error message

A serious internal error has occurred, preventing the *CA VM:Schedule* service virtual machine from continuing execution.

W

Warning message

An abnormal condition has occurred. The function or task continues, although the results may be affected by the abnormal condition.

Help With Messages

CA VM:Schedule provides two sources of information about messages:

- The CMS HELP facility
- The *CA VM:Schedule Message Reference Guide*

These sources list all messages *CA VM:Schedule* can generate, explain the possible causes of each message, and tell you how to respond to that message.

To use the CMS HELP facility for a *CA VM:Schedule* message, type the following command:

1. Type **help** followed by a space.
2. Type the 3-character product identifier (**vmd**)...
3. ...the message number...
4. ...and the severity code.

   Note: Do not type the 3-letter routine or module code that is displayed after the product identifier.

For example, to get help on the message VMDCOM1091E, enter the following command from CMS:

```
help vmd1091e
```

When you have finished reading the message help, press PF3 to exit the CMS HELP facility.

For more information about using the CMS HELP facility, refer to the CMS User's Guide for your system.

Chapter 3: Administration Overview

This chapter describes the *CA VM:Schedule* system administrator tasks, and indicates where you can find further information.

- **Review pre-implementation steps**. To prepare *CA VM:Schedule* for implementation and customization, read Chapter 3, *Configuration File Records for Setup*.
- **Set up authorizations**. Use the AUTHORIZ configuration file record to control which users can perform the following tasks:
  - Issue CLASS and LOG options of the SCHEDULE command
  - Issue operator commands
  - Use EXECs that are set up to run on *CA VM:Schedule*
  - Control other users' requests without providing a password
  - Control their own requests without providing a password
- **Set up initiations and timing**. To control resources available to each request class, define request run periods, and control when requests can run, use these configuration records:
  - CLASS
  - BUSINESS and WEEKEND (define business and weekend days)
  - SHIFT and RANGE (define time and date ranges)
  - QUARTERS (defines calendar quarters)
  - DELAY, RETRY, and WITHIN (control when a request begins executing)
- **Set up password handling**. To control whether *CA VM:Schedule* prompts for passwords or allows them on the command line, and to specify how to handle invalid password attempts, use the JOURNAL configuration file record.
- **Set up request tracking**. Use the MONITOR configuration file record to set up request tracking, including the following tasks:
  - Track requests and the resulting return codes
  - Spool users' consoles to a specified user ID
  - Request a dump and determine its format
- **Set up utilities**. Set up the *CA VM:Schedule* utilities to report on requests in the database (VMDRPT utility).
- **Set up CA VM:Schedule EXECs**. Set up *CA VM:Schedule* EXECs to do the following actions:
  - Send messages without autologging a user ID (VMDMSG EXEC)
  - Punch DOS files to a DOS guest virtual machine (VMDDOS EXEC)

- **Set up user exits**. Set up *CA VM:Schedule* user exits to do the following actions:
  - Monitor a request's resource limits *(*user exit: MONITOR; requires the configuration record USEREXIT MONITOR)
  - Intercept and control user commands *(*user exit: COMMAND; requires the configuration record USEREXIT COMMAND)
  - Intercept and control commands from *CA VM:Schedule* menus (user exit: CMDCHECK; requires the configuration file record USEREXIT CMDCHECK)
  - Examine passwords (user exit: PASSWORD; requires the configuration record USEREXIT PASSWORD)

The following list summarizes the configuration records you use to implement and customize *CA VM:Schedule*.

AUTHORIZ

Grants authorizations to specified user IDs

BUSINESS

Defines business days

CLASS

Defines classes

CLOCKCHK

Guards against incorrect CPU clock setting

DELAY

Sets a default period between request scheduling and execution

DUMP

Specifies whether a dump is taken and where it is routed if CA VM:Schedule abends

JOURNAL

Controls password prompting

MESSAGE

Specifies the CP command that is used to send messages

MSGCASE

Specifies whether *CA VM:Schedule* messages are in all upper case or mixed case

MONITOR

Configures the request-monitoring facility

PRODUCT

Identifies and activates the interface with another CA product

PURGE

Specifies how long a completed or canceled request remains in the *CA VM:Schedule* database

QUARTERS

Defines a calendar quarter in which requests can be run

RANGE

Identifies by name a range of dates or times in which requests can be run

RELEASE

Specifies a virtual machine to be autologged and a request to be run during *CA VM:Schedule* initialization

RETRY

Specifies how long *CA VM:Schedule* waits between autolog attempts

SHIFT

Identifies by name a span of times during which requests can be run

SPOOLTO

Specifies an alternative user ID to receive spooled message files

SYSOPER

Identifies the *CA VM:Schedule* operator

USEREXIT

Specifies user exits to receive control at various points in *CA VM:Schedule* operation

WEEKEND

Defines weekend days

WITHIN

Specifies the default time within which *CA VM:Schedule* must successfully autolog a virtual machine

Chapter 4: Configuration File Records for Setup

The Installation and deployment process for CA VM:Schedule populates its product minidisks and creates its initial configuration files. The product is ready for use. For information about installing and deploying this product, see the CA VM:Schedule *Installation Guide*.

Do not change the name of the VMSCHED CONFIG file regardless of the user ID of the *CA VM:Schedule* service virtual machine.

After *CA VM:Schedule* is installed, customize it for efficient use at your site:

1. Modify the *CA VM:Schedule* configuration file, VMSCHED CONFIG.
2. Modify the PROFILE EXEC on your AUTOLOG1 user ID to issue the CP AUTOLOG command for user ID VMSCHED. This command automatically initializes *CA VM:Schedule* each time your VM system is IPLed.

3. Make sure that users check their PROFILE EXEC files for the following restrictions:

   a. The PROFILE EXEC of users who schedule requests must not prompt for parameters. The PROFILE EXEC also must not contain any commands that alter or destroy the parameters that *CA VM:Schedule* places in the console stack.

      This restriction is due to the way CA VM:Schedule autologs user IDs. *CA VM:Schedule* uses the standard CP XAUTOLOG command, which passes the request command parameters to the console stack of the user ID. These parameters are sent to the virtual machine by CP on the first read to the console. If the PROFILE EXEC contains commands that alter or destroy these parameters, or the first read to the console is issued prematurely, the scheduled request can fail to execute correctly.

   b. For *CA VM:Schedule* to be able to initialize their requests, users must have one of the following:

      - Either SET AUTOREAD ON or SET RUN ON in their PROFILE EXECs
      - AUTOCR in the IPL statement in their directory entries

   c. Users must not have SET AUTOREAD OFF in their PROFILE EXECs. If they do, *CA VM:Schedule* does not execute their commands.

4. Either reinitialize *CA VM:Schedule* by entering the PROFILE command, or initialize *CA VM:Schedule* in a suspended state by entering the VMDSYS SUSPEND command to. When *CA VM:Schedule* is suspended, you can perform special maintenance tasks. Run the *CA VM:Schedule* report utility VMDRPT immediately after any initialization.

5. To allow *CA VM:Schedule* to run disconnected, enter the #CP DISC command.

Note: For more information about installing *CA VM:Schedule*, see the *CA VM:Schedule Installation Guide*.

This section contains the following topics:

AUTHORIZ: Activating the CA VM:Batch Interface

To enable the interface between *CA VM:Schedule* and *CA VM:Batch,* use the AUTHORIZ BATCH record, together with the PRODUCT VMBATCH record. This interface allows users to schedule batch jobs.

Note: For more information about the interface, consult the *CA Mainframe VM Product Manager Product Interface Guide*.

Note: You must also have a PRODUCT VMSCHEDULE record in the *CA VM:Batch* configuration file to enable this interface.

You can access the interface between *CA VM:Schedule* and *CA VM:Batch* only from *CA VM:Batch*. For complete information about using the interface, see the *CA VM:Batch User's and Group Manager's Guide*.

`AUTHORIZ BATCH` *`userids`*

userids

Specifies the user IDs of the service virtual machines that are running *CA VM:Batch.*

AUTHORIZ: Allowing Users to Specify Request Classes

To control whether users can use the CLASS option of the SCHEDULE command, use the AUTHORIZ CLASS/NOCLASS records. Every *CA VM:Schedule* user must have either CLASS or NOCLASS authorization, or both. There is no default.

Using CLASS on scheduled requests allows system control over when requests run. Requests assigned to a CLASS only run when the class is started.

For example, your site can dedicate selected virtual machines for production work. A production control person can schedule work for these machines by using classes. Operators can then run all the requests at the appropriate time using the CLASS START command.

`AUTHORIZ {CLASS | NOCLASS} userids`

CLASS

Allows the specified user IDs to use the CLASS option of the SCHEDULE command. Unless they are also specified on a AUTHORIZ NOCLASS record, the user IDs specified here must use the CLASS option of the SCHEDULE command. These user IDs cannot use the EXEC command, because it has no CLASS option.

NOCLASS

Allows the specified user IDs to schedule requests without using the CLASS option of the SCHEDULE command.

userids

Specifies a blank-separated list of user IDs that receive the authorization. An asterisk indicates that all users are authorized. An asterisk can also be used to indicate front or trailing pattern matching for user IDs in the list.

Examples

- To allow all users to schedule requests without using the CLASS option, specify this record:

  ```
  AUTHORIZ NOCLASS *
  ```

- To allow all users to use the CLASS option of the SCHEDULE command, specify this record:

  ```
  AUTHORIZ CLASS *
  ```

- To allow all users the choice to use the CLASS scheduling options, specify these records:

  ```
  AUTHORIZ NOCLASS *
  AUTHORIZ CLASS *
  ```

AUTHORIZ: Allowing Users to Issue CA VM:Schedule EXECs

To allow users to run administrator-supplied EXECs on the *CA VM:Schedule* service virtual machine, use the AUTHORIZ *execname* record. As a security feature, these users can only schedule, not modify or copy, the specified EXEC.

```
AUTHORIZ execname userids
```

execname

Specifies the name of the EXEC that the specified user IDs are authorized to use.

Note: VMDMSG is an EXEC supplied with *CA VM:Schedule* that could be specified here.

userids

Specifies the user IDs you wish to authorize to run EXECs. An asterisk indicates that all users are authorized. An asterisk can also be used to indicate front or trailing pattern matching for user IDs in the list.

More information:

Using CA VM:Schedule EXECs (see page 91)

Examples

- To authorize users CURLY, MOE, and LARRY to use the VMDMSG EXEC, specify this record:

  ```
  AUTHORIZ VMDMSG CURLY MOE LARRY
  ```

- To authorize all users to use the VMDMSG EXEC, specify this record:

  ```
  AUTHORIZ VMDMSG *
  ```

- To authorize all users that start with the letters ‘SYS’ to use the VMDMSG EXEC, specify this record:

  ```
  AUTHORIZ VMDMSG SYS*
  ```

AUTHORIZ: Controlling Password Use and Prompting

To authorize users to schedule, cancel, and modify their own requests without specifying their logon passwords, use the AUTHORIZ SKIPPASS record . Use this authorization when using *CA VM:Schedule* in automated tasks, to prevent an unexpected prompt in the automated task. Without this authorization, *CA VM:Schedule* issues a password prompt to verify the user's identity whenever the user schedules, modifies, or cancels one of their requests.

WARNING! These records are not recommended for sites with high security needs, or where logged on terminals might be left unattended.

To allow administrative staff to work with other users scheduled requests without knowing the other users' logon passwords, use the AUTHORIZ NOPASS record. Staff with this authorization can schedule requests to run on other users' machines. They can also CHANGE or CANCEL all requests without a password.

`AUTHORIZ {NOPASS | SKIPPASS}` *`userids`*

NOPASS

Gives users the authorization to schedule, modify, and cancel requests for all user IDs without providing a logon password.

SKIPPASS

Gives users the authorization to schedule, modify, and cancel their own requests without providing their logon password.

userids

Specifies a blank-separated list of user IDs that receive the authorization. An asterisk indicates that all users are authorized. An asterisk can also be used to indicate front or trailing pattern matching for user IDs in the list.

Security Considerations

SKIPPASS and NOPASS authorizations can compromise a virtual machine's security. Any user can schedule requests if they access a terminal that is logged on with one of these authorizations.

Users with NOPASS authorization have the same authorization as users who know the *CA VM:Schedule* password. These users can schedule commands to run on *CA VM:Schedule*, possibly compromising the security of your system.

NOPASS authorization overrides *CA VM:Secure* rules. For example, with NOPASS authorization, you can issue the SCHEDULE command for another user ID without entering a password, even if that user ID has a *CA VM:Secure* rule rejecting the command. In addition, users with no special authorization can issue the QUERY and WHEN commands for their own user IDs, in spite of any *CA VM:Secure* rules rejecting these commands.

AUTHORIZ: Allowing Users to Get Request Logs

To authorize users to request a status file when their request is run, use the AUTHORIZ LOG record. This file can be useful in determining why requests did not execute successfully. If this feature is not used, users receive status information from *CA VM:Schedule* only in the following cases:

- When a request does not run
- When a request stops because of an error initiating it
- When a request runs for the last time
- When an exception occurs (for example, if *CA VM:Schedule* skips a scheduled request).

Errors (non-zero return codes) or abends that occur for the request itself are not indicated to the user unless they request that all events be logged through the LOG option.

`AUTHORIZ LOG` *`userids`*

LOG

Gives the user authorization to use the LOG option to produce a status file from *CA VM:Schedule* for all executions of a request. Without this option specified, a user only gets this information when one of the request exceptions listed above occurs.

userids

Specifies a blank-separated list of user IDs that receive the authorization. An asterisk indicates that all users are authorized. An asterisk can also be used to indicate front or trailing pattern matching for user IDs in the list.

AUTHORIZ: Authorizing Operator Commands

To allow selected users to act as the *CA VM:Schedule* operator, use the AUTHORIZ OPERATOR record.

Operator privileges enable users to use all operator commands except the TRANSFER command. This command requires separate authorization, through the AUTHORIZ TRANSFER record. Users automatically have the AUTHORIZ TRANSFER authorization if they are designated as system operators for *CA VM:Schedule* through the SYSOPER configuration file record.

`AUTHORIZ OPERATOR` *`userids`*

OPERATOR

Gives authorization for the users to act as a *CA VM:Schedule* operator. These users can issue any *CA VM:Schedule* commands that require OPERATOR authorization.

userids

Specifies the user IDs who are allowed to issue all operator commands (except TRANSFER) and access operator full-screen menus.

Example

To allow users JIMMY and LOIS to enter *CA VM:Schedule* operator commands, specify this record:

```
AUTHORIZ OPERATOR JIMMY LOIS
```

AUTHORIZ: Allowing Users to Transfer Request Ownership

To allow users such as general directory managers to transfer scheduled requests from one user to another, use the AUTHORIZ TRANSFER record. For example, use the AUTHORIZ TRANSFER record when a user changes departments or is deleted from the directory.

`AUTHORIZ TRANSFER` *`userids`*

TRANSFER

Gives authorization to allow the users to issue the *CA VM:Schedule* TRANSFER command.

userids

The user IDs to be allowed to transfer ownership of a scheduled request from one user ID to another user ID.

PRODUCT VMSISERV: Preset at Installation

When you install this product, the installation procedure automatically configures this product by building a PRODUCT VMSISERV record in the VMSCHED CONFIG file. You can add or modify configuration records by editing this file and reinitializing *CA VM:Schedule*. However, do not alter the PRODUCT VMSISERV record unless otherwise noted.

The PRODUCT VMSISERV record authorizes the VMANAGER to perform component administrative tasks.

Important: Do not delete or change this record.

```
PRODUCT VMSISERV VMANAGER
```

JOURNAL: Password Control

To tell *CA VM:Schedule* whether to allow passwords as an option on the command line, use the JOURNAL record. This record also tells the product what action to take if a user makes too many consecutive invalid password attempts. This feature helps prevent unauthorized changes to scheduled activities, particularly from unattended terminals. If you omit the JOURNAL record, *CA VM:Schedule* prompts users for logon passwords that commands require and it does not report invalid password attempts.

```
JOURNAL {PROMPT | NOPROMPT} threshold [max]
```

PROMPT

Prompts for passwords and does not allow them on the command line. If not specified, PROMPT is the default.

NOPROMPT

Allows passwords as options on the command line.

threshold

Specifies the number of invalid password attempts a user can make for a target user ID before *CA VM:Schedule* alerts the operator.

max

Specifies the maximum number of consecutive invalid password attempts a user can make for the target user ID. If the user exceeds this number, that user is prevented from issuing any more commands requiring a password. If you omit this value or the JOURNAL record altogether, *CA VM:Schedule* does not enforce a maximum.

Example

To prevent users from specifying passwords on the command line, to have *CA VM:Schedule* notify the operator if any user makes more than three consecutive invalid password attempts, and to prevent further commands requiring a password for a target user ID after five consecutive invalid attempts, specify the following record:

```
JOURNAL PROMPT 3 5
```

PRODUCT: Activating the CA VM:Backup Interface

To enable the interface between this product and *CA VM:Backup,* use the PRODUCT VMBACKUP record. The interface allows *CA VM:Backup* to quiesce the *CA VM:Schedule* minidisks during backups. This quiescence ensures a successful backup of the data files, particularly the *CA VM:Schedule* database file.

To specify that *CA VM:Backup* is to quiesce *CA VM:Schedule* minidisks during backups, use the *CA VM:Backup* exception file management screens. For more information about the interface, see the *CA Mainframe VM Product Manager Interface Guide*.

```
PRODUCT VMBACKUP userid [MASTER]
```

VMBACKUP

Enables the *CA VM:Backup* interface.

userid

Specifies the user ID of the service virtual machine running *CA VM:Backup*.

MASTER

Designates the master *CA VM:Backup* system if you use more than one PRODUCT VMBACKUP record.

PRODUCT: Activating the CA VM:Batch Interface

To enable the interface between *CA VM:Schedule* and *CA VM:Batch,* use the PRODUCT VMBATCH record, along with the AUTHORIZ BATCH record. Interfacing these products allows you to schedule batch jobs.

For more information about the interface, see the *CA Mainframe VM Product Manager Interface Guide*.

Note: You must also have a PRODUCT VMSCHEDULE record in the *CA VM:Batch* configuration file to enable this interface.

You can access the interface between *CA VM:Schedule* and *CA VM:Batch* only from *CA VM:Batch*. For complete information about using the interface, see the *CA VM:Batch User and Group Manager Guide*.

`PRODUCT VMBATCH` *`userid`* `[MASTER]`

VMBATCH

Enables the *CA VM:Batch* product interface.

userid

Specifies the user ID of the service virtual machine running *CA VM:Batch*.

MASTER

Designates the master *CA VM:Batch* system if you use more than one PRODUCT VMBATCH record.

PRODUCT: Activating the CA VM:Secure Interface

To enable the interface between *CA VM:Schedule* and *CA VM:Secure,* use the PRODUCT VMSECURE record. Interfacing these products increases data security.

Note: For more information, see the *CA Mainframe VM Product Manager Interface Guide*.

You must also have a PRODUCT VMSCHEDULE record in the *CA VM:Secure* PRODUCT CONFIG configuration file to enable this interface.

```
PRODUCT VMSECURE userid
```

VMSECURE

Enables the *CA VM:Secure* product interface.

userid

Specifies the user ID of the service virtual machine running *CA VM:Secure*.

SSI: Activating Single System Image Mode Operation

Use the SSI record to activate the CA VM:Schedule Single System Image Mode operation.

```
SSI
```

More information:

CA VM:Schedule Single System Image Mode (see page 103)

Description

The SSI record activates the Single System Image Mode in the product. This record allows *CA VM:Schedule* to manage scheduled requests for all members of an SSI cluster.

Important: Your z/VM SSI cluster configuration must include a DISTRIBUTE IUCV TOLERATE statement. If this statement is absent, add it to your z/VM configuration, then IPL every system in your cluster. Without this z/VM system configuration change, IDENTITY users cannot communicate to a *CA VM:Schedule* service machine on a different member of the cluster. For more information on the DISTRIBUTE statement, see the IBM *CP Planning and Administration* guide.

More information:

CA VM:Schedule Single System Image Mode (see page 103)

SYSOPER: Designating the CA VM:Schedule Operator

To define the user IDs that receive *CA VM:Schedule* messages that require immediate attention, use the SYSOPER record.

Users that are designated as message recipients on this record also have OPERATOR authorization. This authorization allows these users to issue all operator commands except for TRANSFER, which requires separate authorization.

If none of the specified operator user IDs is logged on, CA VM:Schedule sends messages to the *CA VM:Schedule* system log.

You can specify multiple SYSOPER records in the VMSCHED CONFIG file.

SYSOPER *userids*

userids

Specifies the user IDs that receive *CA VM:Schedule* operator messages. If you specify the *CA VM:Schedule* service virtual machine, *CA VM:Schedule* sends messages to the *CA VM:Schedule* console.

Chapter 5: Customizing Configuration File Records

This chapter describes the records you need to customize scheduling at your site and to control resource usage.

This section contains the following topics:

BUSINESS, WEEKEND: Defining Business and Weekend Days

To define business and weekend days for request scheduling, use the BUSINESS and WEEKEND records.

- If you use *only one* of these records, the days that are not defined on that record default to the other day type. For example, if you specify only BUSINESS MON TUE WED THU, then Friday, Saturday, and Sunday are treated as weekend days.
- If you omit *both* records, *CA VM:Schedule* considers business days to be Monday through Friday and weekend days to be Saturday and Sunday.
- *CA VM:Schedule* does not consider as business days any dates defined as HOLIDAY on the RANGE record. See *RANGE: Defining Named Date and Time Ranges* (see page 63) for more information about this record.

`{BUSINESS | WEEKEND}` *`days`*

BUSINESS

Specifies which days of the week the AGAIN n BDAYS, BDAILY, and B options of the SCHEDULE and EXEC commands represent.

WEEKEND

Specifies which days of the week the WEEKEND option represents on the SCHEDULE and EXEC screens. Also specifies which days of the week the WE option represents in the SCHEDULE and EXEC commands.

days

Specifies days as business days (in BUSINESS *days*) or weekend days (in WEEKEND *days*). The days can be either spelled out or abbreviated with the three-character abbreviations MON, TUE, WED, THU, FRI, SAT, and SUN.

CLASS: Defining Classes

To create the request processing classes that correspond to the CLASS option on the SCHEDULE command, use the CLASS record. By implementing classes in *CA VM:Schedule*, you can assign a class to requests that you only want to run during certain operations windows. These requests run only when that *CA VM:Schedule* class is started.

This record designates a class and defines the maximum number of virtual machines that *CA VM:Schedule* can autolog at any one time for that class. This record can also define the resource limits for requests that the class processes. For more information, see Resource Limits for Executing Requests (see page 52).

CA VM:Schedule does not try to autolog users who are logged on when a CLASS request is scheduled to run. CLASS requests for logged-on users are immediately requeued. This requeuing prevents the CLASS command processing from reaching its *maxstart* limit with requests that cannot be logged on. With the ATTEMPTS parameter, you can specify how many times *CA VM:Schedule* tries to start the same request. With the WARN/NOWARN parameter, you can warn users that *CA VM:Schedule* is trying to start their CLASS request.

```
CLASS char maxstart [maxcpu [maxsio[maxuio]]] "Parms"

Parms:

[WARN | NOWARN] [ATTEMPTS nnnn | *]
```

char

Identifies this class. Can be any digit or letter, or an asterisk to set global class limits.

maxstart

The maximum number of virtual machines *CA VM:Schedule* is allowed to run simultaneously.

maxcpu

Defines the maximum total (virtual plus overhead) CPU seconds users can specify on the CPU option of the SCHEDULE command. *maxcpu* is the default CPU option.

maxsio

Defines the maximum disk or tape I/Os users can specify on the SIO option of the SCHEDULE command for requests in this class. *maxsio* is the default SIO option.

maxuio

Defines the maximum unit record (reader, printer, punch) I/Os users can specify on the UIO option of the SCHEDULE command. *maxuio* is the default UIO option.

WARN | NOWARN

Indicates whether you want to send a warning message to a logged on user ID that *CA VM:Schedule* is trying to autolog. Warning messages are not sent to user IDs running disconnected. The default is WARN.

ATTEMPTS {*nnnn* | *}

Indicates the number of times *CA VM:Schedule* attempts to initialize a CLASS request. If you specify ATTEMPTS * (the default), *CA VM:Schedule* retries the autolog until it is successful. If you specify ATTEMPTS 0, *CA VM:Schedule* skips the request if the first attempt is unsuccessful. If you specify ATTEMPTS 3, *CA VM:Schedule* tries to autolog the request three times before skipping it.

Resource Limits for Executing Requests

You can limit the resources that can be used to execute a request. The following resources can be limited:

- The total CPU seconds
- The maximum number of disk, tape, and unit record I/Os
- The storage size of the virtual machine that executes the request (SCHEDULE command only)

The resource limits for requests can be set in three locations:

- The USEREXIT MONITOR record. These limits (if set) are the default resource limits.
- The CLASS record (classed requests only). These limits are the maximum resource limits for the request processing class. These limits override any maximum limits that are set on the USEREXIT MONITOR record.
- Options on the SCHEDULE command. Limits that are set here override any maximum limits that are set on the USEREXIT MONITOR record. For classed requests, limits that are set here cannot exceed the limits (if any) set in the CLASS record. However, they can be more restrictive than those limits.

More information:

MONITOR User Exit (see page 97)

Example

The following record specifies these limits for class B:

- Autolog a maximum of five virtual machines.
- Limited to 20 CPU seconds.
- Limited to 10,000 SIOs and 2,000 UIOs.
- Skip any requests that do not start within ten attempts.

```
CLASS B 5 20 10000 2000 ATTEMPTS 10
```

CLOCKCHK: Setting Up CPU Clock Verification

To ensure that requests run at the proper time, use the CLOCKCHK record. This record invokes a clock verification routine during *CA VM:Schedule* initialization.

If the last *CA VM:Schedule* internal clock checkpoint is ahead or behind the CPU clock by more than the *hours* value specified in the record, *CA VM:Schedule* prompts the operator. *CA VM:Schedule* also suspends initialization until the operator replies.

`CLOCKCHK hours`

hours

Specifies the number of hours the CPU clock must be ahead or behind before the clock verification routine executes. The default is 24. Specifying 0 hours disables clock verification.

Recommendation

If your system runs unattended for long periods of time (for example, over a weekend), set the CLOCKCHK hours to a number high enough to cover the amount of time the system runs unattended.

DELAY: Setting a Default Delay For Initiations

To provide a default delay between the time a user schedules a request and the time *CA VM:Schedule* starts the request, use the DELAY record. This record creates a default interval of time (up to 23:59:59) between the scheduling and first initiation of user requests. When a user schedules a request without specifying a start time, *CA VM:Schedule* uses this value.

The delay time applies only to requests scheduled to run the same day.

DELAY *hh:mm:ss*

hh:mm:ss

Specifies the interval of time between the scheduling and first initiation of a user request. The default is one hour (01:00:00).

Example

To delay for 2 minutes the initiation of all requests that are scheduled to run today, specify the following record:

```
DELAY 00:02:00
```

This record only affects requests that have no specified start time.

Requests scheduled to start another day without a start time are scheduled on that day at the time the request was submitted. For example, if at 11 a.m. you schedule a request to run tomorrow, Tuesday, without specifying a start time, *CA VM:Schedule* schedules the request for 11:00 a.m. Tuesday rather than 11:02 a.m. Tuesday.

DUMP: Directing Dumps

Use the DUMP record to direct dumps to a preferred location or to turn them off.

This record specifies whether *CA VM:Schedule* produces a VMDUMP when it abends. You can also specify the user ID to which the dump is spooled. If no DUMP record is specified, *CA VM:Schedule* produces a dump when it abends. If no *userid* parameter is specified, *CA VM:Schedule* sends this dump to the default user ID. This default user ID is specified by the DUMP parameter in the System_Userids statement in the z/VM configuration file.

```
DUMP  {VMDUMP [userid] | OFF}
```

VMDUMP

Specifies that *CA VM:Schedule* creates a VMDUMP. If you do not create a DUMP record, *CA VM:Schedule* creates a VMDUMP by default. VMDUMP is the recommended setting.

userid

Specifies the user ID to which the VMDUMP is spooled.

OFF

Specifies that if *CA VM:Schedule* abends, it does not create a dump.

Important: If you create a DUMP OFF record and CA VM:Schedule abends, you have no record of the abend. To correct the problem that caused the abend, a record of the abend can be useful. To separate *CA VM:Schedule* dumps from other system dumps without eliminating them entirely, specify the record DUMP VMDUMP VMANAGER. This record places dumps in the VMANAGER reader.

Example: Spooling a VMDUMP to VMANAGER

To separate *CA VM:Schedule* dumps from other system dumps, specify the following record:

```
DUMP VMDUMP VMANAGER
```

MESSAGE, MSGCASE: Configuring Message Appearance

To change message appearance to simplify messages, use the MESSAGE and MSGCASE records.

The MESSAGE record allows you to specify whether messages sent by *CA VM:Schedule* have headers. *CA VM:Schedule* uses CP MSG or CP MSGNOH commands for messages (but see **Note**). The choice of using MSG or MSGNOH, set in the MESSAGE record, determines whether there are headers. See the *CA VM:Schedule Message Reference Guide* for explanations of system messages.

Note: You cannot set whether the first warning message that *CA VM:Schedule* sends to a logged-on virtual machine has headers. *CA VM:Schedule* uses CP WARNING for the first message when it tries to initiate a request. *CA VM:Schedule* uses CP MSG or CP MSGNOH for all subsequent messages.

The MSGCASE record allows you to specify whether messages sent by *CA VM:Schedule* are displayed in mixed case or all upper case.

```
MESSAGE {MSG | MSGNOH}
```

MSG

Indicates that the message displays with the header. This setting is the default.

MSGNOH

Specifies that the message has no header. The *CA VM:Schedule* service virtual machine requires CP privilege class A or B or their equivalent for this option.

```
MSGCASE {LOWER | UPPER}
```

LOWER

Displays all messages in mixed case. This is the default option.

UPPER

Displays all messages in upper case.

MONITOR: Tracking Requests

To allow you to run requests under control of the *CA VM:Schedule* request execution monitor, use the MONITOR record. The request execution monitor produces status information not otherwise available. This information is useful when determining why a request did not complete execution. In addition, users can avoid having to issue the CP LOGOFF command from an EXEC or program after their scheduled request completes.

This record activates request execution monitoring, allowing you to:

- Spool users' consoles to a specified user ID
- Request a CPDUMP or VMDUMP in case a request abends
- Have the user ID automatically log off after the request completes
- Send a request completion time and command return code to the *CA VM:Schedule* service virtual machine and a specified user ID.

```
MONITOR {DEFLTYES | DEFLTNO | YES | NO} "Options"


Options:

[CONSOLE [TO] {userid | *}]
[{CPDUMP | VMDUMP} [TO] {userid | *}]
[LOGOFF]
```

DEFLTYES

Invokes request monitoring. Users can still override.

DEFLTNO

Bypasses request monitoring. Users can still invoke.

YES

Always invokes request monitoring. Users cannot override.

NO

Never invokes request monitoring. Users cannot invoke.

CONSOLE [TO] {*userid* | *}

Spools users' consoles to the specified user ID. If you specify * as the user ID, consoles are spooled to the requesting user ID.

{CPDUMP | VMDUMP} [TO] {*userid* | *}

Generates a dump to the specified user ID if the request abends. If you specify * as the user ID, the dump is sent to the requesting user ID.

LOGOFF

Logs off the requesting user ID when the request completes. By default, the user ID remains logged on.

More information:

MONITOR User Exit (see page 97)

Examples

- To invoke request monitoring for every request and spool users' consoles to them, specify the following record:

  ```
  MONITOR YES CONSOLE *
  ```

- To bypass request monitoring (unless overridden by a user) and, if a request abends, generate a VMDUMP and send it to user ID ACCTPOOL, specify the following record:

  ```
  MONITOR DEFLTNO VMDUMP TO ACCTPOOL
  ```

Restrictions

The *CA VM:Schedule* request execution monitor runs as a nucleus extension and uses bit 26 in control register 0. The monitor sets this bit when a request begins executing on the virtual machine. If the bit is still set, the request is assumed to be running. Otherwise, the request is assumed to have terminated abnormally. If your site has software installed that has a conflicting use for bit 26, contact CA Technical Support.

If your site monitors requests, warn users to make sure their user IDs have access to the *CA VM:Schedule* communications module (VMSCHED MODULE). If *CA VM:Schedule* autologs a user ID and cannot find this module, the request does not run. The user ID sits idle for 15 minutes until the system forces it off. (After the user ID is forced off, the QUERY command shows the status UNKNOWN TERMINATION.)

PURGE: Purging Completed or Canceled Requests

To prevent the *CA VM:Schedule* database from becoming cluttered, use the PURGE record.

This record removes completed and canceled requests from the *CA VM:Schedule* database.

PURGE *days*

days

Specifies the number of days *CA VM:Schedule* waits before purging completed and canceled requests. If the PURGE record is not specified, the default PURGE value is three days.

Examples

- To purge completed and canceled requests at midnight after two days, specify the following record:

  ```
  PURGE 2
  ```

- To purge completed and canceled requests at midnight, or the next time CA VM:Schedule initializes--whichever happens first, specify the following record:

  ```
  PURGE 0
  ```

QUARTERS: Defining Calendar Quarters

Use the QUARTERS record to define the following time periods:

- For the QUARTERS option on the SCHEDULE entry screen.
- For the AGAIN QTRLY option of the line-mode SCHEDULE command.

The QUARTERS record defines the four quarters of the year by specifying the first date of each quarter. If your fiscal year changes, you can modify the QUARTERS record to reschedule automatically all quarterly scheduled EXECs after the next time they run.

QUARTERS 1-*mm/dd* 2-*mm/dd* 3-*mm/dd* 4-*mm/dd*

n-mm/dd

The n part of the parameter indicates the quarter being defined. The mm/dd part defines the month and day on which each quarter begins. If you do not specify this record, default quarter dates are 1-01/01, 2-04/01, 3-07/01, and 4-10/01.

Important: These dates must be unique and they must create four ranges that do not overlap or span more than 12 months.

Example: Starting a Year With a Month Other Than January

If you define the first quarter to start with a month other than January, *CA VM:Schedule* interprets subsequent quarters defined by earlier months as belonging to the next year. For example, you include this QUARTERS record at the beginning of 2015:

```
QUARTERS 1-07/01 2-10/01 3-01/01 4-04/01
```

CA VM:Schedule interprets it to mean these quarters:

Quarter	Calendar time
1	July 1, 2015
2	October 1, 2015
3	January 1, 2016
4	April 1, 2016

Example: Starting the First Quarter With a Month That Passed

If you define the first quarter to start with a month that has already passed for the current year, *CA VM:Schedule* interprets that quarter as next year's. For example, you include this QUARTERS record on July 31, 2015:

```
QUARTERS 1-01/01 2-04/01 3-07/01 4-10/01
```

CA VM:Schedule interprets it to mean these quarters:

Quarter	Calendar time
1	January 1, 2016
2	April 1, 2016
3	July 1, 2015
4	October 1, 2015

RANGE: Defining Named Date and Time Ranges

To define keywords for time periods inside or outside of which users can schedule requests, use the RANGE record. You can define any number of such keywords. These keywords can refer to date or date and time ranges.

This record creates time and date ranges (shifts) that users can call by name when using the INSIDE and OUTSIDE options of the SCHEDULE and EXEC commands. If you omit RANGE (and SHIFT) records, users cannot specify the INSIDE and OUTSIDE options. Dates must be in chronological order and must be satisfiable.

```
RANGE rangename {"ParmA"} ["ParmB"]

ParmA:

date1 [time1] | day1 [time1] | time1


ParmB:

date2 [time2] | day2 [time2] | time2
```

rangename

Defines the name of the range.

date1

Specifies the beginning date of the range, in the format *mm*/*dd*/*yy*.

day1

Specifies the first day of the range, either spelled out or abbreviated to three letters (MON, TUE, WED, THU, FRI, SAT, SUN).

time1

Specifies the start time of the range, in the format *hh*:*mm*:*ss*. This operand can be used with the date and day specifications or used alone.

date2

Specifies the end date of the range, in the format *mm*/*dd*/*yy*.

day2

Specifies the last day of the range, either spelled out or abbreviated to three letters (MON, TUE, WED, THU, FRI, SAT, SUN).

time2

Specifies the end time of the range, in the format *hh*:*mm*:*ss*. This operand can be used with the date and day specifications or used alone.

Assigning Multiple Time Periods to a Single Range Name

You can assign more than one time period to a single range by using more than one RANGE record. You can define multiple ranges by dates or by clock times. All ranges that are associated with the same range name are logically considered together (they are cumulative).

For example, the following records define different parts of the same range, called PARTTIME. The first record defines Monday as belonging to the time range PARTTIME; the second record defines Wednesday as belonging to the same time range:

```
RANGE PARTTIME MON
RANGE PARTTIME WED
```

Requests scheduled with the OUTSIDE option of the SCHEDULE and EXEC user commands do not run on either day. Requests scheduled with the INSIDE option run on both of these days.

Restrictions

For time ranges defined by clock time, *CA VM:Schedule* considers 00:00:00 to be the beginning of a day and 23:59:59 to be the end. *CA VM:Schedule* considers 24:00:00 to be equivalent to 23:59:59, and always converts it to 23:59:59. If you want to end a time range at midnight rather that one second before midnight, specify the end time as 00:00:00 of the following day instead of 24:00:00.

You cannot define the beginning hour of a time range as 24:00:00. Ranges that are defined without an end time are assumed to end at 23:59:59.

Example: Defining a Time Range

To define the time range LATE to begin at 6 p.m. and end at midnight, inclusive, specify this record:

```
RANGE LATE 18:00:00 00:00:00
```

Example: Defining a Date and Time Range

To define a range called SPECIAL for the date September 25, 2022, from 6 p.m. through midnight, inclusive, specify this record:

```
RANGE SPECIAL 09/25/22 18:00:00 09/26/22 00:00:00
```

Shifts and Ranges Together

Shifts (defined by the SHIFT record) and ranges that have the same name are logically considered together. For example, the following records define the times when reduced rates are in effect on your system:

```
SHIFT CHEAP 17:00:00 23:59:59 BUSINESS
RANGE CHEAP SAT SUN
```

HOLIDAY as a Range Name

If you specify dates for a special range called HOLIDAY, *CA VM:Schedule* considers those dates to be nonbusiness days when processing requests.

For example, the following record designates December 25 and 26, 2015, as part of the range called HOLIDAY. No regular business processing happens on those days.

```
RANGE HOLIDAY 12/25/15 12/26/15
```

This record means that all day December 25 and all day December 26 are treated as holidays.

However, if you need regular business processing to happen on parts of those days, use a range name other than HOLIDAY. Other range names let you treat only parts of those days as holidays. You can then specify both the dates and the times that this range defines:

```
RANGE XMAS 12/25/15 08:00:00 12/26/15 08:00:00
```

RELEASE: Initiating a Request at CA VM:Schedule Initialization

Use the RELEASE record to initiate a request when *CA VM:Schedule* initializes.

The configuration file can contain zero or more RELEASE records.

RELEASE *userid requestname*

userid

Specifies the user ID of the virtual machine that owns the request to be initiated.

requestname

Specifies the name of the request to be initiated.

RESID: Defining the CA VM:Schedule Resource ID

To switch the protocol used to communicate between the product command MODULE and the server virtual machine from IUCV to APPC, use the RESID record.

```
RESID resourcename GLOBAL [SSIONLY]
```

resourcename

Specifies the name of an APPC resource. This resource name must also be specified on an IUCV*IDENT statement in the product server directory entry.

GLOBAL

Specifies that the named resource is known globally throughout the TSAF or ISFC collection. The GLOBAL parameter must also be specified on an IUCV *IDENT statement in the product server directory entry, if this operand is present.

SSIONLY

Specifies that APPC connections to the RESID are only accepted from VM systems residing in the SSI cluster as the VM system hosting the product server virtual machine.

More information:

CA VM:Schedule Single System Image Mode (see page 103)

Description

The RESID record is one of the common product configuration statements. It is not currently required for any specific purpose in *CA VM:Schedule*.

RETRY, WITHIN: Limiting Autolog Attempts

To give users time to log off their sessions before *CA VM:Schedule* tries to autolog their virtual machines, use the RETRY and WITHIN records. RETRY and WITHIN also give requests that cannot run at their scheduled times a chance to run. (If the requests could not run because they were on hold, the request status must change. If they could not run because an operator stopped request initiations, the operator must start request initiations.)

The RETRY record specifies how often *CA VM:Schedule* attempts to autolog a virtual machine to start a request. The WITHIN record limits how long *CA VM:Schedule* attempts to autolog a user to run a request.

Make the WITHIN value several times greater than the RETRY value. Doing so allows several autolog attempts. Also make WITHIN slightly greater than your site's average downtime.

CA VM:Schedule skips requests that do not initiate before the WITHIN period ends and cancels single execution requests.

```
RETRY  seconds
WITHIN hh:mm:ss
```

RETRY *seconds*

Defines the number of seconds *CA VM:Schedule* waits before retrying an initiation. The default is 60 and the maximum is 65535.

WITHIN *hh:mm:ss*

Defines how long the autolog attempts are to continue. If the WITHIN record is not specified, the default WITHIN time is 5 minutes (00:05:00). The maximum WITHIN time that you can specify is just under 100 hours (99:59:59).

Examples

- To tell *CA VM:Schedule* to try autologging every five minutes (300 seconds), specify this record:

  ```
  RETRY 300
  ```

- To tell *CA VM:Schedule* to spend no more than two hours attempting to autolog, specify this record:

  ```
  WITHIN 02:00:00
  ```

SHIFT: Defining Named Time Ranges

To define keywords for time periods inside or outside of which users can schedule requests, use the SHIFT record.

This record creates time ranges that users can call by name when using the INSIDE and OUTSIDE options of the SCHEDULE and EXEC commands. If you omit SHIFT (and RANGE) records, users cannot specify the INSIDE and OUTSIDE options.

```
SHIFT shiftname starttime [endtime] ["Parms"]

Parms:

  firstday [lastday]
| BUSINESS
| WEEKEND
| EVERYDAY
```

shiftname

Specifies the name to give the shift.

starttime

Defines the shift's begin time, in the format *hh*:*mm*:*ss*.

endtime

Defines the shift's end time, in the format *hh*:*mm*:*ss*.

firstday

Defines the first day of the shift, either spelled out or abbreviated to three letters (MON, TUE, WED, THU, FRI, SAT, SUN).

lastday

Defines the last day of the shift, either spelled out or abbreviated to three letters (MON, TUE, WED, THU, FRI, SAT, SUN). If you do not specify *lastday*, the shift is only one day long-specified by *firstday*.

BUSINESS

Indicates that the shift applies to business days (Monday through Friday).

WEEKEND

Indicates that the shift applies to weekend days (Saturday and Sunday).

EVERYDAY

Indicates that the shift applies to all days.

Assigning Multiple Time Periods to a Single Shift Name

You can assign more than one time period to a single shift by using more than one SHIFT record. All shifts that are associated with the same shift name are logically considered together (they are cumulative).

For example, you can create a shift called STAFF to correspond to your staff meetings, which are held from 9 a.m. to 10 a.m. Mondays, Wednesdays, and Fridays:

```
SHIFT STAFF 09:00:00 10:00:00 MON
SHIFT STAFF 09:00:00 10:00:00 WED
SHIFT STAFF 09:00:00 10:00:00 FRI
```

Order of Shift Times

The begin and end times of a shift must be in chronological order and must be satisfiable. For example:

```
SHIFT 6AM211PM 06:00:00 23:00:00
```

Shifts That Span Midnight

To define a shift that spans midnight, use two SHIFT records or the RANGE record. For example, you can define a shift from 11 p.m. to 6 a.m. either of these ways:

Two SHIFT records define a shift that spans midnight:

```
SHIFT THIRD 23:00:00 23:59:59
SHIFT THIRD 00:00:00 06:00:00
```

One RANGE record defines a range that spans midnight:

```
RANGE THIRD 23:00:00 06:00:00
```

CA VM:Schedule considers 00:00:00 to be the beginning of a day and 23:59:59 to be the end. *CA VM:Schedule* considers 24:00:00 to be equivalent to 23:59:59, and always converts it to 23:59:59. If you want to end a shift at midnight rather than one second before midnight, specify the end time as 00:00:00 of the following day rather than as 24:00:00.

To end a shift at midnight, define it to include 00:00:00 of the following day. For example, to define the LATE shift to be from 6 p.m. through midnight, Monday through Friday, use the following records:

```
SHIFT LATE 18:00:00 23:59:59 MON FRI
SHIFT LATE 00:00:00 00:00:00 TUE SAT
```

Using Shifts and Ranges Together

Shifts and ranges (defined by the RANGE record) that have the same name are logically considered together. For example, the following records define the times (evenings and weekends) when reduced rates are in effect on your system:

```
SHIFT CHEAP 17:00:00 23:59:59 BUSINESS
RANGE CHEAP SAT SUN
```

Examples

To define your system's prime time shift to be from 9 a.m. to 5 p.m., Monday through Friday, use either of the following records:

```
SHIFT PEAK 09:00:00 17:00:00 MONDAY FRIDAY
```

or

```
SHIFT PEAK 09:00:00 17:00:00 BUSINESS
```

SPOOLTO: Specifying Spooled File Destinations

To redirect spooled message files, use the SPOOLTO record. *CA VM:Schedule* redirects message files when the user ID that was specified on the SPOOL option of the SCHEDULE or EXEC command does not exist. If you omit this record, *CA VM:Schedule* discards messages that it cannot send to the user ID that is specified in the SPOOL option.

SP0OLT0 *userid*

userid

Defines the user ID to whom spooled messages are redirected if the user ID that is specified in the SPOOL option is unavailable.

Recommendation

Use the SPOOLTO record when one or more users have LOG authorization.

USEREXIT: Passing Control to User Exits

Use the USEREXIT record to provide your site with additional control over:

- Runaway requests
- Access to *CA VM:Schedule* commands
- Password validation

The USEREXIT record has the following syntax:

```
USEREXIT {"Parms"}

Parms:

  CMDCHECK filename
| COMMAND filename
| MONITOR filename interval [cpu[sio[uio]]]
| PASSWORD filename
```

CMDCHECK *filename*

Passes control to the exit routine specified by *filename* on receipt of any *CA VM:Schedule* MENU-driven command. The user exit must have a filetype of EXEC. VMDEXIT5 XEXEC is provided as a sample CMDCHECK user exit.

COMMAND *filename*

Passes control to the exit routine specified by *filename* on receipt of any *CA VM:Schedule* line-mode command before command parameter and option validation begins. The user exit file must have a filetype of EXEC. VMDEXIT3 XEXEC is provided as a sample COMMAND user exit.

MONITOR *filename*

Monitors the execution of virtual machines that *CA VM:Schedule* autologs. Whenever one of these virtual machines exceeds its resource limits, *CA VM:Schedule* passes control to the exit routine specified by *filename*. The user exit file must have a filetype of EXEC. VMDEXIT1 XEXEC is provided as a sample MONITOR user exit.

interval

Specifies the number of seconds *CA VM:Schedule* is to wait between resource checks.

[*cpu*[*sio*[*uio*]]]

Specifies the total (virtual plus overhead) CPU time, disk and tape I/O, and unit record (reader, printer, punch) I/O limits assigned to requests.

PASSWORD *filename*

Passes control to the exit routine specified by *filename* every time a password is processed, immediately before normal *CA VM:Schedule* password verification begins. This user exit does not receive control if the user issuing the command is specified on an AUTHORIZ NOPASS record. The user exit file must have a filetype of EXEC. VMDEXIT4 XEXEC is provided as a sample PASSWORD user exit.

More information:

Using CA VM:Schedule User Exits (see page 93)
Required Configuration File Record (see page 99)

Examples

- The following record specifies the following monitoring conditions:
  - Each request is monitored every 30 seconds.
  - If a request exceeds 60 CPU seconds, *CA VM:Schedule* calls the MONITOR user exit MON1 EXEC.

  ```
  USEREXIT MONITOR MON1 30 60
  ```
- To pass control to the COMMAND user exit COMM1 EXEC just before *CA VM:Schedule* processes a command, specify this record:

  ```
  USEREXIT COMMAND COMM1
  ```
- To pass control to the PASSWORD user exit JOURN1 EXEC just before *CA VM:Schedule* verifies a password, specify this record:

  ```
  USEREXIT PASSWORD JOURN1
  ```

Chapter 6: Creating Request Database Reports

To provide reports about scheduled requests to your systems staff or other users or departments, use the VMDRPT utility. VMDRPT capabilities include:

- Producing a report that lists some or all requests in the *CA VM:Schedule* request database.

  This report can use a selection file that controls request selection and ordering. You select the report contents by changing VMDRPT parameters and running the utility.

- Including specific requests in the report.

  You can specify which requests to include in a report by entering SELECT records in the selection file. See Including Requests in the Report (see page 81) for information about the SELECT record.

- Excluding specific requests from the report.

  You can specify which requests to omit from a report by entering IGNORE records in the selection file. See Excluding Requests From the Report (see page 82) for information about the IGNORE record.

- Sorting and positioning requests in report fields.

  You can determine the order in which requests are included in a report by entering a SORT record in the selection file. See Sorting Requests for the Report (see page 83) for information about the SORT record.

This section contains the following topics:

VMDRPT Utility

```
VMDRPT [irbdb_filename [selection_filename]] [("Options"]


Options:

SORT field1 [field2 [field3]]
```

irbdb_filename

Specifies the name of the *CA VM:Schedule* database file. The filetype must be IRBDB. If not specified, the default filename is VMSCHED.

selection_filename

Specifies the name of the selection file you want to use for processing. The filetype must be SELECT; the filemode must be A. The default file id is VMDRPT SELECT A. If you do not specify a selection file and a VMDRPT SELECT file does not exist on your 'A' disk, all *CA VM:Schedule* requests are included in the report.

SORT

Identifies parameters that follow as sort fields. The following fields are valid:

- CLASS
- CPU
- LASTRUN
- NEXTRUN
- REQUEST
- SIO
- STATUS
- UIO
- USERID

Note: Sorting specified by VMDRPT overrides sorting specified by a SORT record in the selection file.

field1

The first sort field. The default is USERID.

field2

The second sort field. The default is REQUEST.

field3

The third sort field. There is no default.

Example

To create a report sorted by userid, then by status, and then by class, specify the following:

```
VMDRPT VMSCHED (SORT USERID STATUS CLASS
```

Report Contents

The following explains the fields in default order in VMDRPT output:

USERID

Specifies the user ID that owns the request.

REQUEST

Specifies name of the request.

LAST DATE/TIME

Specifies day, date, and time at which the request was last initiated. If an asterisk follows the time, it means the last run of this scheduled request was initiated with the RELEASE command.

NEXT DATE/TIME

Specifies day, date, and time of the next scheduled request initiation.

CPU

Specifies the threshold for the number of real CPU seconds (TTIME) the virtual machine can use while executing the request; if this threshold is exceeded, *CA VM:Schedule* calls the MONITOR user exit.

SIO

Specifies the threshold for the number of disk and tape I/Os the virtual machine can perform while executing the request. If this threshold is exceeded, *CA VM:Schedule* calls the MONITOR user exit.

UIO

Specifies the threshold for the number of unit record (reader, printer, punch) I/Os the virtual machine can perform while executing the request. If this threshold is exceeded, *CA VM:Schedule* calls the MONITOR user exit.

CLASS

Specifies execution class for the request.

STORAGE

Specifies virtual machine storage size in which *CA VM:Schedule* executes the request.

EXECUTION STATUS

Displays a summary of the results of the last initiation results.

Note: For more information about the execution statuses, see the *CA VM:Schedule User Guide*.

Using VMDRPT at Your Site

To use VMDRPT utility functions, you must do the following:

- Select a user ID on which to run VMDRPT. VMDRPT cannot run on the *CA VM:Schedule* service virtual machine when *CA VM:Schedule* is operating because it runs in the user area.
- Establish a read-only link to the *CA VM:Schedule* service virtual machine database disk. This disk is usually the 1B0 disk. This link provides access to the request database.
- Suspend *CA VM:Schedule* and reaccess the VMSCHED database minidisk. This prevents any updates to the request database and gets the most current data from the database.

To run VMDRPT without suspending *CA VM:Schedule,* follow these steps:

1. Initialize *CA VM:Schedule* daily and issue the VMDRPT command from the *CA VM:Schedule* service virtual machine PROFILE EXEC (before the VMDSYS command).
2. SUSPEND *CA VM:Schedule* and copy the VMSCHED IRBDB file to the user ID that is about to run the utility. You can RESUME *CA VM:Schedule* processing once the database has been copied to another disk.

Including Requests in the Report (SELECT Record)

To specify a criterion for selecting the requests that you want to include in VMDRPT reports, use the SELECT record. You can have as many SELECT records in the VMDRPT selection file as you need, so select requests to meet the desired output results. For a request to be included, it must meet at least one of the SELECT criteria and none of the IGNORE criteria.

SELECT *field value*

field

Specifies a field in the request database.

value

Specifies the value or values that must be contained in the named field for that request to be included in the report.

More information:

Excluding Requests From the Report (IGNORE Record) (see page 82)
SELECT and IGNORE Fields and Values (see page 86)

Examples

- To generate a report of all requests with a request name beginning with TEST, include the following record in the selection file:

  ```
  SELECT REQUEST TEST*
  ```

- To generate a report containing canceled and error requests, include this record in the selection file:

  ```
  SELECT STATUS CANCEL ERROR
  ```

- To generate a report containing information on requests belonging to user IDs LUCY and CHARLES, include the following record in the selection file:

  ```
  SELECT USERID LUCY CHARLES
  ```

Excluding Requests From the Report (IGNORE Record)

To specify a criterion for identifying the requests that you want to exclude from VMDRPT reports, use the IGNORE record. You can have many IGNORE records in the VMDRPT selection file (default name VMDRPT SELECT A). Therefore, you can omit requests that meet any of several criteria. For a request to be excluded, it must meet at least one of the IGNORE criteria.

```
IGNORE field value
```

field

Specifies a field in the request database.

value

Specifies the value or values that must be contained in the named field for that request to be excluded from the report.

More information:

SELECT and IGNORE Fields and Values (see page 86)

Examples

- To generate a report excluding all requests with a user ID beginning with TEST, specify the following record in the selection file:

  ```
  IGNORE USERID TEST*
  ```

- To generate a report for requests for user IDs BELLE and BLUE, excluding requests for those users that last ran in June of 2002, specify the following records in the selection file:

  ```
  SELECT USERID BELLE BLUE
  IGNORE LASTRUN 06/01/02 THRU 06/30/02
  ```

Sorting Requests for the Report (SORT Record)

To sort reported requests in ascending order and to position fields on the VMDRPT reports, use the SORT record. The selection file can contain only one SORT record.

If you include a SORT record, do not use the SORT parameter on VMDRPT; the parameter overrides any sort specifications that you make through this record.

SORT *field1* [*field2* [*field3*]]

***field1* [*field2* [*field3*]]**

Specifies the fields on which to sort the report. The following fields are valid:

- CLASS
- CPU
- LASTRUN
- NEXTRUN
- REQUEST
- SIO
- STATUS
- SYSNAME
- UIO
- USERID

field1

Specifies the first field on which to sort the report (appears as the first field on the report). The default first sort field is USERID.

field2

Specifies the second field on which to sort the report (appears as the second field on the report). The default second sort field is REQUEST.

field3

Specifies the third field on which to sort the report (appears as the third field on the report). There is no default third sort field.

Running VMDRPT Automatically

To run VMDRPT automatically at regular intervals, set up an EXEC that suspends *CA VM:Schedule* and runs the VMDRPT utility. Alternatively, schedule the utility with the SCHEDULE command.

If you run VMDRPT through an EXEC, that EXEC must be on the user ID that runs the utility. The user that submits the EXEC must have OPERATOR authorization to suspend *CA VM:Schedule*.

Setting Up and Scheduling an EXEC: Example

To automate reporting on the *CA VM:Schedule* database, follow these steps:

1. Create an EXEC that runs the VMDRPT utility. An example of such an EXEC is shown below as the REPORT EXEC.
2. Create a CA VM:Schedule request that runs this exec at 2:00 AM on a daily basis. Use the following example as a model:

   ```
   vmsched schedule dailyrpt report (at 02:00 again daily
   ```

These VMDRPT defaults are in effect in the REPORT EXEC sample that follows:

- The *CA VM:Schedule* service virtual machine is VMSCHED
- The selection file is VMDRPT SELECT
- Requests are sorted first by user ID and then by request name
- The first two data fields on the report are USERID and REQUEST.

The following report is the REPORT EXEC example for running the VMDRPT utility:

```
/*  Create a VM:Schedule database report using VMDRPT.     */

address ''

qrc = 0

 /*  SUSPEND VM:Schedule for a bit to capture a stable      */

/*   database for reporting.                                */

 Do until qrc = 117

      'VMSCHED SUSPEND'

      'CP SLEEP 3 SEC'

      'VMSCHED QUERY'

      qrc = rc

End


/* VM:Schedule is suspended at this point.                */

/* LINK and ACCESS the software and database disk, run    */

/* VMDRPT to create the report, detach the disks and      */

/* RESUME VM:Schedule                                     */
```

```
'CP LINK VMSCHED 192 992 RR'

'CP LINK VMSCHED 1B0 9B0 RR'

'ACCESS 992 D'

'ACCESS 9B0 E'

'VMDRPT'

'RELEASE D (DET'

'RELEASE E (DET'

'VMSCHED RESUME'

exit 0
```

Scheduling VMDRPT with the SCHEDULE Command: Example

The following SCHEDULE command schedules a report that runs every Monday morning and lists the requests scheduled to initiate during the upcoming week:

```
vmsched schedule currweek curwkrpt (on mon at 07:00 again weekly
```

The CURWKRPT EXEC is similar to the EXEC above, except the VMDRPT command line is:

```
vmdrpt vmsched weekly
```

where the WEEKLY SELECT file contains the following record:

```
SELECT NEXTRUN * THRU + 6
```

SELECT and IGNORE Fields and Values

Identify the requests that are included and excluded in the report by specifying the appropriate fields in each record. Include the values that those fields must contain. The following table lists the relevant information.

Field	Value Format	Include on Record?	Contents of the Resulting Report
CLASS	class1 [... classn]	SELECT	Includes all requests of the specified classes

Field	Value Format	Include on Record?	Contents of the Resulting Report
		IGNORE	Omits all requests of the specified classes
CPU	limit1 [THRU limit2]	SELECT	Includes all requests whose execution falls within the specified CPU range
		IGNORE	Omits all requests whose execution falls within the specified CPU range
LASTRUN	mm/dd/yy [hh:mm:ss] [THRU mm/dd/yy [hh:mm:ss]] or mm/dd/yy [hh:mm:ss] [THRU +integer]	SELECT	Includes all requests whose last run date (and time) falls within the specified range
		IGNORE	Omits all requests whose last run date (and time) falls within the specified range
NEXTRUN	mm/dd/yy [hh:mm:ss] [THRU mm/dd/yy [hh:mm:ss]] or mm/dd/yy [hh:mm:ss] [THRU +integer]	SELECT	Includes all requests whose next run date (and time) falls within the specified range
		IGNORE	Omits all requests whose next run date (and time) falls within the specified range
REQUEST	name1 [...namen]	SELECT	Includes all requests of the specified names
		IGNORE	Omits all requests of the specified names
SIO	limit1 [THRU limit2]	SELECT	Includes all requests whose execution falls within the specified SIO range
		IGNORE	Omits all requests whose execution falls within the specified SIO range
STATUS	statusfld1 [... statusfldn]	SELECT	Includes all requests whose statuses match the specified statuses
		IGNORE	Omits all requests whose statuses match the specified statuses

Field	Value Format	Include on Record?	Contents of the Resulting Report
STORAGE	limit1K [THRU limit2K] or limit1M [THRU limit2M]	SELECT	Includes all requests whose storage falls within the specified range
		IGNORE	Omits all requests whose storage falls within the specified range
SYSNAME	Systemname1 [systemnamen...]	SELECT	Includes all requests whose system name field matches what was specified. Use SYSTEM * to SELECT requests that run on the same system that CA VM:Schedule runs on.
		IGNORE	Omits all requests whose system name field matches what was specified. Use SYSTEM * to IGNORE requests that run on the same system that CA VM:Schedule runs on.
UIO	limit1 [THRU limit2]	SELECT	Includes all requests whose execution falls within the specified UIO range
		IGNORE	Omits all requests whose execution falls within the specified UIO range
USERID	userid1 [... useridn]]	SELECT	Includes all requests belonging to the specified userids
		IGNORE	Omits all requests belonging to the specified userids

[statusfld values are presented in Values For the STATUS Field (statusfld) (see page 89).

] You can specify as many as will fit on an 80-character record.

Values For the STATUS Field (statusfld)

Values for the STATUS field can be any of the values listed in the following table. This table also shows which requests these values select.

Status Field Value	Requests Selected
ADJUSTED	Selects requests whose STATUS is "Date adjusted"
CANCEL	Selects requests whose STATUS is "Request canceled" or whose STATUS is "Purge scheduled" and NEXTRUN is "Request canceled"
COMPLETE	Selects requests whose STATUS is "Not rescheduled" or "Purge scheduled" or "Initiation failed" or "Request canceled" or whose STATUS is "Purge scheduled" and NEXTRUN is "Skipped to completion"
DELAYED	Selects requests whose STATUS is "Initiation delayed"
ERROR	Selects requests whose NEXTRUN is "Execution error" or whose STATUS is "Execution error"
FAILED	Selects requests whose STATUS is "Initiation failed" or whose NEXTRUN is "Canceled-autolog error"
LOGGEDON	Selects requests whose STATUS is "User was logged on"
HOLD	Selects requests that were placed on hold by the HOLD command or by an automatic method
NEW	Selects requests whose LASTRUN is "Has not executed"
NOHOLD	Selects requests that were removed from a previous hold
NORMAL	Selects requests whose STATUS is "Normal initiation"
PURGE	Selects requests whose STATUS is "Purge scheduled"
RUN	Selects requests whose STATUS is "Execution complete"
SKIPPED	Selects requests whose STATUS is "Initiation skipped" or whose NEXTRUN is "Skipped to completion"
UNKNOWN	Selects requests whose STATUS is "Unknown completion"

Chapter 7: Using CA VM:Schedule EXECs

If you write your own EXECs to run on *CA VM:Schedule*, keep in mind the following points:

- EXEC requests run on the *CA VM:Schedule* service virtual machine (svm), not on an autologged user. This fact leads to the other points.
- The EXECs should be short. While the programs are running, this product cannot respond to other user requests.
- *CA VM:Schedule* passes to the EXEC the input parameters that were specified when the request was scheduled. In addition, *CA VM:Schedule* passes the user ID that scheduled the EXEC request on the program stack. By passing the requesting user ID, *CA VM:Schedule* enables the programmer to execute selective logic that is based on that user ID.
- To use the EXECs, users must be authorized in the VMSCHED CONFIG file.

More information:

AUTHORIZ: Allowing Users to Issue CA VM:Schedule EXECs (see page 36)

This section contains the following topics:

Punching DOS Files

The VMDDOS EXEC punches a DOS/VSE file to a DOS/VSE guest virtual machine reader. The EXEC is loaded from the distribution tape during installation.

The following assumptions are made in punching DOS files:

- You do not have any device defined at virtual address 00F.
- The virtual punch is spooled **CLASS A NOCONT NOHOLD** to the VSE virtual machine.

Modifications

Make these modifications to the VMDDOS EXEC following the introductory comments:

- Change the FILETYPE variable to provide a default filetype for the VSE files.
- Change the DOSVM variable to provide a default VSE service virtual machine.
- If desired, change the FILEMODE variable to specify a filemode other than the default * (any accessed disk).

After you modify the VMDDOS EXEC, move it to a public minidisk.

Debugging

The EXEC starts a console log, but automatically purges it if there are no errors.

Sending Messages Without Autologging

The VMDMSG EXEC lets you schedule and send routine messages without autologging a virtual machine. The VMDMSG EXEC is loaded from the distribution tape during *CA VM:Schedule* installation. This EXEC resides on the *CA VM:Schedule* service virtual machine minidisk. You can execute it through the EXEC command.

Required Authorizations for Using CA VM:Schedule EXECs

Users can issue the VMDMSG EXEC only if you have authorized them on the AUTHORIZ VMDMSG configuration record. You can authorize all users by specifying **AUTHORIZ VMDMSG ***.

For MSGNOH, the *CA VM:Schedule* service virtual machine requires CP privilege class A or B, or their equivalent.

Warning! This warning applies if the *CA VM:Schedule* service virtual machine has CP privilege class A: By authorizing users to use EXECs on *CA VM:Schedule*, you authorize them to be able to broadcast messages to all other logged-on user IDs.

Example

Create a request called **pammsg** that schedules the VMDMSG EXEC to send ANDREW a message every Monday at 9:00 a.m. The text of the message is *meeting at 10*.

```
vmsched exec pammsg vmdmsg andrew meeting at 10 (at 9:00 on mon again
weekly
```

Chapter 8: Using CA VM:Schedule User Exits

CA VM:Schedule supports four user exits that must be written in REXX. These exits are provided with the product installation files as sample REXX user exits. Before you use these sample user exits, have a REXX programmer review and edit them as needed for your site.

CMDCHECK

Examines, accepts, or rejects commands that are generated from *CA VM:Schedule* fullscreen menus.

Sample user exit: VMDEXIT5 XEXEC

COMMAND

Examines, modifies, or rejects user commands.

Sample user exit: VMDEXIT3 XEXEC

MONITOR

Is passed control when an autologged virtual machine exceeds resource thresholds while executing a request.

Sample user exit: VMDEXIT1 XEXEC

PASSWORD

Examines passwords before normal *CA VM:Schedule* verification.

Sample user exit: VMDEXIT4 XEXEC

This section contains the following topics:

How to Implement User Exits

1. Have a REXX programmer review and modify the particular sample user exit for your site. Rename the sample to have a filetype of EXEC. The Filename can be anything you like.
2. Add the appropriate USEREXIT record to the *CA VM:Schedule* configuration file.

   Example: This record specifies that you are using the MONITOR user exit with the file name VMDEXIT1:

   ```
   USEREXIT MONITOR VMDEXIT1
   ```
3. Make sure the *CA VM:Schedule* service machine has access to the user exit EXEC.

CMDCHECK User Exit

To examine, accept and reject commands that are generated from full screen menus, use the CMDCHECK user exit. You cannot modify commands coming from full screen menus. The following parameter list is passed to the CMDCHECK user exit:

userid command command-parameters

userid

Specifies the user ID issuing the menu drive command.

command

Specifies the command being requested from a full screen menu.

command-parameters

Specifies the parameters that are set through the full screen menu for this command.

Required Configuration File Record

The following record must be present in the VMSCHED CONFIG file in order to use the CMDCHECK user exit:

USEREXIT CMDCHECK *execname*

More information:

USEREXIT: Passing Control to User Exits (see page 74)

Return Codes

The CMDCHECK user exit supports the following return codes:

0

Accepts the menu-generated command.

8

Rejects the menu-generated command and displays any message that is pushed on the program stack.

COMMAND User Exit

To examine, reject, or modify commands that *CA VM:Schedule* is about to process, use the COMMAND user exit. The following parameter list is passed to the COMMAND user exit:

userid commandline

userid

Specifies the user ID issuing the command.

commandline

Specifies the command being issued, including all parameters.

Required Configuration File Record

The following record must be present in the VMSCHED CONFIG file in order to use the COMMAND user exit:

USEREXIT COMMAND *execname*

More information:

USEREXIT: Passing Control to User Exits (see page 74)

Return Codes

The COMMAND user exit supports the following return codes:

0

Accepts the command as-is.

4

Accepts the command with any changes to the command line. The COMMAND user exit pushes these changes back on the program stack. The user ID cannot be changed. If more than one push command is used, *CA VM:Schedule* accepts the contents of only the last push command. If the user exit returns code 4 but does not push any changes on the stack, *CA VM:Schedule* disregards the user exit.

8

Rejects the command and display to the user any message the user exit pushes on the program stack.

MONITOR User Exit

To pass control when an autologged virtual machine exceeds a resource threshold while executing a request, use the MONITOR user exit. When this user exit is called, it performs actions specific to your site. The following parameter list is passed to the MONITOR user exit:

```
MCBuser MCBrname MCBtype  MCBcpuct MCBsioct
MCBuioct MCBxcpu MCBxsio MCBxuio MCBlcpu MCBlcpu
MCBlsio MCBluio MCBscpu MCBssio MCBsuio
MCBetime MCBflag MCBclass
```

MCBuser

Specifies the user ID being monitored.

MCBrname

Specifies the name of request being executed.

MCBtype

Specifies the type of exception detected.

MCBcpuct

Specifies the CPU exception count (from 0 through 255 inclusive).

MCBsioct

Specifies the SIO exception count (from 0 through 255 inclusive).

MCBuioct

Specifies the UIO exception count (from 0 through 255 inclusive).

MCBxcpu

Specifies the number of CPU seconds consumed.

MCBxsio

Specifies the number of disk/tape SIOs performed.

MCBxuio

Specifies the number of unit record I/Os performed.

MCBlcpu

Specifies the real CPU time limit.

MCBlsio

Specifies the disk/tape SIO limit.

MCBluio

Specifies the unit record I/O limit.

MCBscpu

Specifies the last sample CPU time.

MCBssio

Specifies the last sample SIO count.

MCBsuio

Specifies the last sample UIO count.

MCBetime

Specifies the final elapsed run time.

MCBflag

Specifies the character flag: Y indicates a classed request; N indicates an unclassed request.

MCBclass

Specifies the batched request class if MCB*flag* is Y.

More information:

USEREXIT: Passing Control to User Exits (see page 74)
Resource Limits for Executing Requests (see page 52)

Required Configuration File Record

The following record must be present in the VMSCHED CONFIG file in order to use the MONITOR user exit:

`USEREXIT MONITOR execname`

More information:

USEREXIT: Passing Control to User Exits (see page 74)

Return Codes

Return codes do not have any significance for the MONITOR user exit. It is expected the user exit will perform all functions necessary for dealing with a request that has exceeded its resource limits.

Return codes less than zero and greater than 800 are used internally by *CA VM:Schedule* and should not be used. All other return codes are ignored by the MONITOR user exit.

PASSWORD User Exit

To allow your site to examine all password specifications before they are accepted by normal *CA VM:Schedule* logic, use the PASSWORD user exit. The following parameter list is passed to the PASSWORD user exit:

PASSWORD *requser objuser suppass*

requser

Specifies the user ID that issued the command.

objuser

Specifies the user ID that *requser* specifies on the command line.

suppass

Specifies the password entered by *requser.*

Required Configuration File Record

The following record must be present in the VMSCHED CONFIG file in order to use the MONITOR user exit:

USEREXIT PASSWORD *execname*

More Information:

USEREXIT: Passing Control to User Exits (see page 74)

Return Codes

The PASSWORD user exit supports the following return codes:

0

Password accepted and *CA VM:Schedule* performs no further checking.

4

Password rejected.

8

CA VM:Schedule must validate the password using DMSPASS.

Chapter 9: Using Non-CA VM:Secure Security Packages

CA VM:Schedule requires authorizations or ways to do the following actions when an external security manager is installed:

- XAUTOLOG user IDs to run scheduled requests
- Validate the usage of this product through a user password.

This section contains the following topics:

Allowing CA VM:Schedule to XAUTOLOG Virtual Machines

This product does not access logon passwords. Therefore, this product must be authorized to XAUTOLOG user IDs without passwords.

How to Allow CA VM:Schedule to XAUTOLOG Without a Password

You can allow autologging with or without a security manager. Use one of the following methods:

- Grant to this product the appropriate security manager authorizations to XAUTOLOG without a password.
- If you do not use a security manager, you can use CP privilege class assignment to allow this product to XAUTOLOG without a password.

Allowing Password Validation for CA VM:Schedule Commands

This product uses the IBM CSL routine DMSPASS to validate passwords. If your security manager does not support DMSPASS (specifically Diagnose X'88'), use the SKIPPASS and NOPASS configuration file authorization records.

Password Validation Using DMSPASS

The *CA VM:Schedule* service virtual machine has the following requirements to use the IBM DMSPASS routine for validating passwords:

- ESM authorization to use DMSPASS, specifically to use the Diagnose X'88' password/passphrase checker.
- The record OPTION DIAG88 in the *CA VM:Schedule* directory entry.

DMSPASS is the recommended method of password validation.

Note: Some security packages require an IUCV ALLOW record in the service virtual machine directory entry. Check with your security package vendor for their requirements.

Use SKIPPASS and NOPASS Authorization Records

If your security manager does not support DMSPASS, use the SKIPPASS and NOPASS configuration file authorization records to use this product without passwords. SKIPPASS authorization allows a user to use this product without a password for their own requests. NOPASS authorization allows a user to use this product without a password for requests that other users own.

To use the SKIPPASS and NOPASS configuration file authorization records, open the VMSCHED CONFIG file and create AUTHORIZ records. These records allow system management user IDs to access requests owned by other users without passwords. These records also allow all users to access their own records without passwords. Base the AUTHORIZ records on the following records, but with parameters appropriate for your site.

```
AUTHORIZ NOPASS MAINT OPERATOR VMANAGER
AUTHORIZ SKIPPASS *
```

Chapter 10: CA VM:Schedule Single System Image Mode

In Single System Image Mode, one *CA VM:Schedule* service virtual machine satisfies all scheduled requests for all members of the Single System Image cluster.

This means:

- Only one *CA VM:Schedule* service virtual machine to define and service
- Only one *CA VM:Schedule* service virtual machine to administer
- A central place to manage all your SSI cluster scheduled events
- *CA VM:Schedule* can be initialized on any member of the cluster. Therefore, you can continue running scheduled requests even if you shut down the system that the product is currently running on.

When SSI Mode is configured, the product accepts a SYSNAME option on the CHANGE, QUERY, SCHEDULE, and DISPLAY commands. Using SYSNAME allows you to specify that the request runs on a specific SSI member or to add selection criteria to QUERY and DISPLAY. If you try to use the SYSNAME option and SSI is not configured, you receive an error and the command does not complete.

Important: Requests that are not assigned to a specific SSI member run on the system where the *CA VM:Schedule* service virtual machine is running. Make sure these requests access only resources that are common to all members of the SSI cluster.

This section contains the following topics:

Requirements

The following items are required to activate *CA VM:Schedule*'s SSI mode operation:

- The product is installed on a Single System Image cluster.
- Make sure that your VMSCHED service virtual machine (svm) is defined as a USER entry rather than an IDENTITY. As a USER entry, only one definition exists for this user ID. This user ID can run on any system in the SSI cluster but can only run on one member at a time.
- Your z/VM SSI cluster configuration must include a DISTRIBUTE IUCV TOLERATE statement. This statement enables IDENTITY users to communicate to the VM:Schedule svm on another member of the SSI cluster. Such communication must be enabled for IDENTITY users to interact with *CA VM:Schedule* or have requests run on them.

  If you change your z/VM configuration to include this statement, IPL all systems in the cluster before turning on *CA VM:Schedule* SSI mode. IPLing a system puts the new configuration into effect on that system. This configuration must match on all members of the SSI cluster, therefore IPL all systems in the cluster.

  Note: For more information on the z/VM system configuration DISTRIBUTE statement, see the IBM *CP Planning and Administration* guide.
- Add an SSI record to the *CA VM:Schedule* configuration file VMSCHED CONFIG. For more information, see SSI: Activating Single System Image Mode Operation (see page 46).
- If you are running z/VM 6.3.0 or below, make sure that you have applied IBM apar VM65269 on all systems that are members of the SSI cluster. This apar allows the product to autolog user IDs properly for requests and to start the requested command.

Preparing for Single System Image Mode Activation

If you have a new *CA VM:Schedule* installation, you can turn on Single System Image Mode without preparing for the installation. If you want to interface *CA VM:Batch* with *CA VM:Schedule*, install a separate *CA VM:Schedule* service machine user ID on the *CA VM:Batch* local system.

If you have an ongoing implementation of *CA VM:Schedule*, follow these steps:

1. If *CA VM:Schedule* is integrated with *CA VM:Batch*, move the *CA VM:Batch* requests to a *CA VM:Schedule* local service virtual machine (svm). See How to Move CA VM:Batch Requests to a Local CA VM:Schedule SVM (see page 105).
2. Prepare your current request databases for merging into one central request database. When you put the databases together, each request must be unique by request name and the user ID that owns the request. See How to Prepare Request Databases for Merging (see page 107).
3. Before you go to production with the new configuration, test this procedure multiple times.

For production systems, perform this procedure when as few requests as possible are scheduled to run. Doing so lessens the possibility that requests are skipped because they are not able to run inside their WITHIN time. If necessary, contact the users who own the requests that run in the window of time in which you do this work. Give them the opportunity to DELAY their requests or otherwise change their scheduled time to be outside of your conversion window.

How to Move CA VM:Batch Requests to a Local CA VM:Schedule SVM

In SSI-enabled mode, *CA VM:Schedule* cannot interface with *CA VM:Batch*. The *CA VM:Batch* requests must be on a *CA VM:Schedule* service virtual machine (svm) that is local to the *CA VM:Batch* svm. To extract the *CA VM:Batch* requests and place them on a local *CA VM:Schedule* service machine, complete the following procedures:

1. Create the local *CA VM:Schedule* service virtual machines. Back up the request database and check request schedules. See Prepare to Move CA VM:Batch Requests (see page 106).
2. Extract all *CA VM:Batch* requests from your current database. For each system that hosts *CA VM:Batch*, move the *CA VM:Batch* requests to a *CA VM:Schedule* service virtual machine (svm) local to the *CA VM:Batch* svm. See Extract CA VM:Batch Requests and Move Them to the Local CA VM:Schedule Service Virtual Machine (see page 106).
3. Repopulate the SSI Mode CA VM:Schedule Service Virtual Machine (see page 107).

Prepare to Move CA VM:Batch Requests

To prepare to move *CA VM:Batch* requests to a local *CA VM:Schedule* service virtual machine, follow these steps:

1. Create a *CA VM:Schedule* service machine for each system that runs scheduled *CA VM:Batch* requests.

   Note: For more information about defining and deploying a new service virtual machine, see the *CA VM:Schedule Installation Guide.*

2. Find a time of day when no *CA VM:Batch* requests are running. Use the *CA VM:Schedule* DISPLAY command and/or the VMDRPT utility (see page 77).

3. Bring down the *CA VM:Schedule* and *CA VM:Batch* machines that are involved in this move. By bringing down both machines, you have control of the databases and know that they will not change while this work is performed.

4. Back up the VMSCHED 1B0 so you can restore the current state of the database if necessary.

5. Make a copy of the VMSCHED IRBDB file on the VMSCHED 1B0 disk. Copy the file to another disk that can hold two copies of that file. The output from the VMDDBEXT extraction utility goes to the same disk as the input file.

Extract CA VM:Batch Requests and Move Them to the Local CA VM:Schedule Service Virtual Machine

To extract the *CA VM:Batch* requests and move them to the new local *CA VM:Schedule* service virtual machine, follow these steps:

1. Run the VMDDBEXT utility on the VMSCHED IRBDB file. Have the utility separate *CA VM:Batch* requests from other scheduled requests. For more information, see VMDDBEXT Utility (see page 115).

2. After the utility runs, take the output file that contains the *CA VM:Batch* requests. Put it on the 1B0 disk of the local *CA VM:Schedule* service machine that you created for *CA VM:Batch* requests.

3. Update or place a PRODUCT VMBATCH (see page 45) record and an AUTHORIZ BATCH (see page 33) record in the VMSCHED CONFIG for the local service virtual machine. Doing so allows the interface to the *CA VM:Batch* machine.

4. Update the VMBATCH CONFIG file for the *CA VM:Batch* service virtual machine. Add a PRODUCT VMSCHEDULE record that points to the local *CA VM:Schedule* service machine.

5. Initialize the local *CA VM:Schedule* and *CA VM:Batch* machines.

   These machines are now complete and can run all requests.

Repopulate the SSI Mode CA VM:Schedule Service Virtual Machine

After CA VM:Batch is integrated with a local CA VM:Schedule service virtual machine (svm), repopulate the SSI mode *CA VM:Schedule* svm.

Follow these steps:

1. Take the DISCARDS-filetype output file from the VMDDBEXT utility. This file contains the non-VMBATCH requests. Put this file on the 1B0 disk of the *CA VM:Schedule* service virtual machine that runs in SSI mode. Rename the file as the VMSCHED IRBDB file.
2. Remove the PRODUCT VMBATCH and the AUTHORIZ BATCH records from the VMSCHED CONFIG file.
3. To continue normal operations for the scheduled requests, initialize the SSI mode *CA VM:Schedule*.

How to Prepare Request Databases for Merging

A cluster's individual *CA VM:Schedule* request databases can be merged into one by appending each SSI member's VMSCHED IRBDB file to another. However, each request must be unique by request name and owning user ID. Therefore, examine each database before merging them. Make sure that all requests in the resultant merged file are unique. Rename each non-unique request on their original *CA VM:Schedule* service virtual machine before attempting to merge databases together.

After you rename requests, you can assign requests to run on specific SSI members. You do not have to do so at this time unless it is convenient.

To prepare request databases for merging, complete these procedures:

1. Get a Report on Each VMSCHED IRBDB File (see page 108).
2. Rename Non-Unique Requests (see page 108).
3. Identify Requests To Run on a Specific SSI Cluster Member (see page 109).

   Note: This last procedure is not required for preparing request databases for merging. However, the merge preparation process provides an opportunity to identify and assign requests to a specific cluster member. You can also assign requests to a specific SSI cluster member after you merge databases and activate SSI operation mode.

Get a Report on Each VMSCHED IRBDB File

On each SSI member system where you are currently using *CA VM:Schedule*, follow these steps:

1. Log on to a user ID that is authorized to SUSPEND and RESUME the product and LINK to its RUNTIME and database disks.
2. Issue the VMSCHED SUSPEND command. Suspending *CA VM:Schedule* stops updates to the VMSCHED IRBDB file. This command lets you run a report against the database without it changing.
3. Link to the VMSCHED 1B0 and 192 disks. This link gives you access to the VMSCHED IRBDB database and the VMDRPT utility.
4. To get a report of your database file, run the VMDRPT utility (see page 77). Sort the report by request and owner ID.
5. Issue the VMSCHED RESUME command. This command resumes operation of your current *CA VM:Schedule* production machine.

   You now have a list of all requests, sorted by request name and owner ID. Your *CA VM:Schedule* production machine, which you suspended, is running again.

Rename Non-Unique Requests

When you have a VMDRPT report for each VMSCHED IRBDB file, rename any non-unique requests. Follow these steps:

1. Search the VMDRPT output for requests that are not unique by request name and owning user ID.
2. Rename each non-unique request. To rename a request, use the SCHEDULE command full screen with the COPY function. This command makes a copy of the request with a new name. Repeat this step on each system where the non-unique request exists. You can keep the original name of a duplicate request on one system.
3. Delete the original request.

   All your requests are now unique by name and owner ID.

Identify Requests To Run on a Specific SSI Cluster Member

Any requests that are not assigned to a specific SSI member run on the system where the *CA VM:Schedule* service virtual machine is running. However, *CA VM:Schedule* can be initialized on any member of the SSI cluster. Therefore, requests that are not assigned to a specific SSI member must only use resources that are common to all members.

If you have any requests that use resources that are specific to an SSI member, assign those requests to run on only that member.

Follow these steps:

1. Extract the records and place them in a file. Use the VMDDBEXT Utility (see page 115).
2. Run the VMDCHSYS utility (see page 112) against these requests. This utility lets you set an SSI member association (SYSNAME) for each request. Merge the utility output files carefully. Make sure you end up with the same requests as in your original file.

Note: You can also assign requests to a specific SSI cluster member after you merge databases and activate SSI operation mode. Use the CHANGE command. (See the *CA VM:Schedule User Guide*.)

Chapter 11: Database Utilities

CA VM:Schedule has two utilities other than the VMDRPT reporting utility. They are VMDCHSYS and VMDDBEXT EXECs. VMDCHSYS allows you to do a batch update of the database with system information to prepare for SSI mode operation. VMDDBEXT allows you to extract database records to move to another *CA VM:Schedule* user ID.

This section contains the following topics:

VMDCHSYS Utility

To make mass SYSNAME field changes to a *CA VM:Schedule* database file, use the VMDCHSYS utility.

VMDCHSYS *filename filemode old-sysname new-sysname*

filename

Specify the filename of the input database file. This file must have a filetype of IRBDB.

filemode

Specify the filemode where the input file resides. This disk or SFS directory must have enough space to contain two copies of the input file. The output from the utility is placed on this filemode.

old-sysname

Specify the system name that you want to change. If you want to change all requests that currently do not have a system name assignment, specify *.

new-sysname

Specify the system name that you want to change to. If you want to remove system name assignments from all requests that currently have one, specify *.

Description

The VMDCHSYS utility allows you to make mass system name changes to your *CA VM:Schedule* database file, the VMSCHED IRBDB. Use this utility to change to new system names when you are renaming SSI members. Also use this utility to prepare a database for merging with others when converting *CA VM:Schedule* to Single System Image mode.

The system names supplied to this utility are not validated. Therefore, you can change system names in the database before changing your SSI member system names. Because the names are not validated, make sure that you are using the correct new system name.

The VMDCHSYS utility resides on the product's RUNTIME disk, usually the 192, so you must link to that disk to have access to the utility. You can run the utility on the VMSCHED service machine but the product must be shut down to do so.

The input to the utility must exist on a R/W disk that can contain two copies of it. Its filetype must be IRBDB but the filename can be anything. This flexibility of filename allows you to make a renamed copy of the original VMSCHED IRBDB file.

If you are running this utility on a copy of the VMSCHED IRBDB file, make sure that you SUSPEND the product before copying the file. Once the file is copied, use the RESUME command to continue processing.

Keep in mind that as long as the product is running, the database can be updated. Therefore, to change records that will be moved into production environments, bring the product down then initialize it after the changes are complete.

The utility creates an interim work file with the input file's filename and the IRBDBOUT filetype. (The IRBDBOUT file cannot exist before you run the utility.) When the utility completes the requested change, it changes the input file's filetype to IRBDBOLD. The output file has the IRBDB filetype.

Note: The final service virtual machine database file must be named VMSCHED IRBDB. However, you can use the utility to process an input database file with any file name as long as the filetype is IRBDB. This flexibility enables you to have IRBDB files from different systems and use filenames to keep track of their origins.

Examples

Run the VMDCHSYS utility to change all requests in a VMSCHED IRBDB file on a R/W B disk that have a system name of SYSTEMA to have a system name of PRODS2:

```
vmdchsys vmsched b systema prods2
```

Run the VMDCHSYS utility to change all requests in *CA VM:Schedule* database VMDNOSYS on the B disk that don’t have a system name associated to have one of PRODS1:

```
vmdchsys vmdnosys b * prods1
```

VMDDBEXT Utility

To extract records from a *CA VM:Schedule* database file, use the VMDDBEXT utility. For example, use this utility to extract your *CA VM:Batch* requests to run on a local *CA VM:Schedule* when you want to turn on SSI mode.

VMDDBEXT *filename filemode* {TYPE *recordtype* | *fieldname value*}

filename

Specify the filename of the input database file. The file must have a filetype of IRBDB.

filemode

Specify the filemode where the input file resides. This disk or SFS directory must have enough space to contain two copies of the input file. The output from the utility is placed on this filemode.

TYPE *recordtype*

To extract records based on the type of record, specify this operand. *Recordtype* indicates the type of record and can be one of the following values:

- VMBATCH – Extract *CA VM:Batch* scheduled requests
- EXEC – Extract EXEC requests that run on the *CA VM:Schedule* service virtual machine.
- REGULAR – Extract requests that are not *CA VM:Batch* requests or EXEC requests.

Fieldname value

To select records based on the value in a specific field in the database file, specify this operand.

Currently, the only supported fieldname you can use is OWNER. This fieldname specifies that selection is based on the owner of the scheduled request. Set the value to the user ID whom you want to select requests for.

Description

The VMDDBEXT utility allows you to extract selected records from a *CA VM:Schedule* database.

The VMDDBEXT utility resides on the product's RUNTIME disk, usually the 192, so you must link to that disk to have access to the utility. You can run the utility on the VMSCHED service machine but the product must be shut down to do so.

The input to the utility must exist on a R/W disk that can contain two copies of it. Its filetype must be IRBDB but the filename can be anything. This flexibility of filename allows you to make a renamed copy of the original VMSCHED IRBDB file.

If you are running this utility on a copy of the VMSCHED IRBDB file, make sure that you SUSPEND the product before copying the file. Once the file is copied use the RESUME command to continue processing.

Keep in mind that as long as the product is running, the database can be updated. Therefore, to extract records that will be moved into production environments, bring the product down then initialize it after your extraction work is done.

The utility creates two output files. The input to the utility is not changed. The records that you select for extraction are put in a file that has the IRBDBEXT filetype and the same filename as the input file. The rest of the records are put in a file that has the DISCARDS filetype and the same filename as the input file. These files cannot exist before you run the utility.

Note: The final service virtual machine database file must be named VMSCHED IRBDB. However, you can use the utility to process an input database file with any file name as long as the filetype is IRBDB. This flexibility enables you to have IRBDB files from different systems and use filenames to keep track of their origins.

Examples

Run the VMDDBEXT utility against the VMSCHED IRBDB file on a 'B' disk that is accessed R/W to extract *CA VM:Batch* scheduled requests:

```
vmddbext vmsched b type vmbatch
```

Run the VMDDBEXT utility against a SYSTEMQ IRBDB file on a R/W 'A' disk to extract all requests that are owned by user ID VMANAGER:

```
vmddbext systemq a owner vmanager
```

Index