Stonebranch Solutions

Version 4.2.0

Universal Event Monitor

Reference Guide

uem-ref-4200

Universal Event Monitor

Reference Guide

Stonebranch Solutions 4.2.0

Document Name	Universal Event Monitor 4.2.0 Reference Guide				
Document ID	**uem-ref-4200**				
Components	**z/OS**	**UNIX**	**Windows**	**IBM i**	**HP NonStop**
Universal Event Monitor Manager	√	√	√		
Universal Event Monitor Server		√	√		
UEMLoad		√	√		

Stonebranch Documentation Policy

Summary of Changes

Changes for Universal Event Monitor 4.2.0 Reference Guide (uem-ref-4200) August 6, 2010

Universal Event Monitor 4.2.0

- Moved information from Universal Event Monitor 4.1.0 User Guide into Universal Event Monitor 4.2.0 Reference Guide.

  Information on component features and examples was moved to the Indesca and Infitran 4.2.0 User Guides.
- Added information about supplemental group memberships for HP-UX 11.00 (and later) in Section 9.2.1 Execution Context of 9 Universal Event Monitor Server for UNIX.
- Added Section 10.9 KEEPALIVE_INTERVAL in 10 UEM Server Configuration Options.
- Added KEEPALIVE_INTERVAL in the following sections:
  - Network Category Options in Section 8.4.1 Configuration Options
  - Network Category Options in Section 9.4.1 Configuration Options

Changes for Universal Event Monitor 4.1.0 Reference Guide (uem-ref-4100) February 10, 2010

- No changes were required for this version of the document.

Changes for Universal Event Monitor 3.2.0 Reference Guide (uem-ref-3203) September 8, 2009

- Specified behavior when the MAX_OCCURRENCE_COUNT configuration option specifies that no event occurrences are to be monitored (value=0).

Universal Event Monitor 3.2.0.2

- Added the following code pages in Section 17.2 Character Code Pages:
  - IBM875
  - IBM4971

Changes for Universal Event Monitor 3.2.0 Reference Guide (uem-ref-3203) December 17, 2008

- Corrected the environment variable name for the Universal Event Monitor Manager SYSTEM_ID option.

Changes for Universal Event Monitor 3.2.0 Reference Guide (uem-ref-3202) October 17, 2008

- Specified the maximum valid value for date and time in:
  - Universal Event Monitor Manager INACTIVE_DATE_TIME option
  - UEMLoad ACTIVE_DATE_TIME option

Changes for Universal Event Monitor 3.2.0 Reference Guide (uem-ref-3201) September 5, 2008

- Added 12 Universal Event Monitor UACL Entries.
- Added toll-free telephone number for North America in C Customer Support.

Changes for Universal Event Monitor 3.2.0 Reference Guide (uem-ref-320) May 16, 2008

Universal Event Monitor 3.2.0

- Added the following configuration options in 6 UEM Manager Configuration Options:

    - BIF_DIRECTORY
    - CONNECT_TIMEOUT
    - DNS_EXPAND
    - HANDLER_TYPE
    - HOST_SELECTION
    - INSTALLATION_DIRECTORY
    - NLS_ DIRECTORY
    - OPTIONS
    - PLF_DIRECTORY
    - SYSTEM_ID
- Modified the REMOTE_HOST configuration option in 6 UEM Manager Configuration Options.
- Added the NLS_ DIRECTORY configuration option in 10 UEM Server Configuration Options.
- Added the following configuration options in 16 UEMLoad Utility Configuration Options:
    - HANDLER_TYPE
    - OPTIONS
- Added 11 Universal Event Monitor Component Definition Options.
- Deleted the following specification methods for all configuration options in 10 UEM Server Configuration Options:
    - Command Line, Short Form
    - Command Line, Long Form
    - Environment Variable
- Added Configuration File Keyword as a specification method for Windows configuration options in:
    - 6 UEM Manager Configuration Options
    - 10 UEM Server Configuration Options

Contents

List of Figures

List of Tables

Preface

Document Structure

This document is written using specific conventions for text formatting and according to a specific document structure in order to make it as useful as possible for the largest audience. The following sections describe the document formatting conventions and organization.

Cross-Reference Links

This document contains cross-reference links to and from other Stonebranch Solutions documentation.

In order for the links to work correctly:

- Place the documents in the same folder.
- In Adobe Reader / Adobe Acrobat, de-select **Open cross-document link in same window** in the **General** category of your **Preferences** dialog (selected from the **Edit** menu).

Conventions

Specific text formatting conventions are used within this document to represent different information. The following conventions are used.

Typeface and Fonts

This document provides tables that identify how information is used. These tables identify values and/or rules that are either pre-defined or user-defined:

- *Italics* denotes user-supplied information.
- **Boldface** indicates pre-defined information.

Elsewhere in this document, **`This Font`** identifies specific names of different types of information, such as file names or directories (for example, **`\abc\123\help.txt`**).

Command Line Syntax Diagrams

Command line syntax diagrams use the following conventions:

Convention	Description
`bold monospace font`	Specifies values to be typed verbatim, such as file / data set names.
`italic monospace font`	Specifies values to be supplied by the user.
[ ]	Encloses configuration options or values that are optional.
{ }	Encloses configuration options or values of which one must be chosen.
|	Separates a list of possible choices.
. . .	Specifies that the previous item may be repeated one or more times.
BOLD UPPER CASE	Specifies a group of options or values that are defined elsewhere.

Table P.1 Command Line Syntax

Operating System-Specific Text

Most of this document describes the product in the context of all supported operating systems. At times, it is necessary to refer to operating system-specific information. This information is introduced with a special header, which is followed by the operating system-specific text in a different font size from the normal text.

z/OS

This text pertains specifically to the z/OS line of operating systems.

This text resumes the information pertaining to all operating systems.

Tips from the Stoneman

Look to the Stoneman for suggestions
or for any other information
that requires special attention.

Stoneman's Tip

Vendor References

References are made throughout this document to a variety of vendor operating systems. We attempt to use the most current product names when referencing vendor software.

The following names are used within this document:

- **z/OS** is synonymous with IBM z/OS and IBM OS/390 line of operating systems.
- **Windows** is synonymous with Microsoft's Windows 2000 / 2003 / 2008, Windows XP, Windows Vista, and Windows 7 lines of operating systems. Any differences between the different systems will be noted.
- **UNIX** is synonymous with operating systems based on AT&T and BSD origins and the Linux operating system.
- **IBM i** is synonymous with IBM i/5, IBM OS/400, and OS/400 operating systems.
- **IBM System i** is synonymous with IBM i Power Systems, IBM iSeries, IBM AS/400, and AS/400 systems.

Note: These names do not imply software support in any manner. For a detailed list of supported operating systems, see the Stonebranch Solutions 4.2.0 Installation Guide.

Universal Event Monitor Terminology

A complete description of the features and capabilities of Universal Event Monitor requires the use of specific terms, which appear throughout this document. These terms, along with their meaning, are listed below for easy reference.

Demand-driven Server – A UEM Server process that is started by a Universal Broker at the request of a UEM Manager or the UEMLoad utility. In this situation, the UEM Server is said to have started upon demand. When a demand-driven UEM Server is started by a UEM Manager, it will run only until monitoring of a single event is completed. When a demand-driven Server is started by the UEMLoad utility, it will only run until the requested database operation is finished. (See also ***Event-driven Server***.)

Event definition – The parameters that define a system occurrence to UEM. These parameters may come from the command line of a UEM Manager or from a record stored in the event definition database.

Event handler **–** A response to an event occurrence, an event handler specifies the action to take when a system event monitored by UEM satisfies certain conditions. The action taken may be a system command or a script.

Event-driven Server – A UEM Server process that is started automatically by a Universal Broker, without a request from a client. An event-driven Server monitors only those system events that have been assigned to it via an event definition record. (See also ***Demand-driven Server***.)

Event occurrence – Each instance of a system event detected by Universal Event Monitor. Any event monitored by a given UEM Server may be defined so that any number of distinct system occurrences may satisfy the event criteria. Each system occurrence that satisfies the event criteria will be detected by UEM as an event occurrence. It follows, then, that each event monitored by UEM will result in the detection of 0 (zero) or more event occurrences.

Expired – Describes an event state set by UEM. It also refers to the event handler that is executed whenever this state is set.

Unlike the other two states that can be set by UEM, ***Triggered*** and ***Rejected***, `expired` refers to an event definition itself, and not to a specific event occurrence.

Only one test is conducted before an event is set to an `expired` state. If the event becomes inactive and no occurrence of that event was detected by UEM, it will be set to an expired state.

The event definition may specify an event handler to execute when an event expires, but the assignment of an expired handler is optional. The state of an event does not depend upon whether this assignment was made.

Handler process – The system command or set of script statements that is executed by UEM on behalf of a given event handler.

Rejected – Describes the state of an event occurrence. It also refers to the event handler that is executed whenever an event occurrence enters this state.

(See also ***Expired*** and ***Triggered***.)

If UEM determines that an event occurrence has not completed within the allotted time (that is, before the event is scheduled to become inactive), that occurrence is set to a **`rejected`** state.

If the event definition specifies an event handler to execute when an event occurrence is rejected, that handler is referred to as the rejected handler. As with a triggered handler, the assignment of a rejected handler is optional, and the state of the event occurrence does not depend upon whether this assignment is made.

Tracking – Describes the state of a system event detected and monitored by UEM that has not met the completion criteria specified in the event definition.

Triggered – Describes the state of an event occurrence. It also refers to the event handler that is executed whenever an event occurrence enters this state.

(See also ***Expired*** and ***Rejected***.)

When an event occurrence is detected, UEM performs tests on that occurrence, based on criteria specified in the event definition, to determine when it completes. When UEM determines that an event occurrence is complete, it sets that occurrence to a triggered state.

The event definition may also specify an event handler to execute when an occurrence of an event becomes triggered. This assignment of a triggered handler is optional, and the state of the event occurrence does not depend upon whether this assignment is made.

Document Organization

The document is organized into the following chapters:

- Overview (Chapter 1)
  General architectural and functional overview of Universal Event Monitor.
- Universal Event Monitor Manager (Chapter 2)
  General description of the UEM Manager functionality.
- Universal Event Monitor Manager for z/OS (Chapter 3)
  Description of the UEM Manager, plus procedures and options specific for z/OS operating systems.
- Universal Event Monitor Manager for Windows (Chapter 4)
  Description of the UEM Manager, plus procedures and options specific for Windows operating systems.
- Universal Event Monitor Manager for UNIX (Chapter 5)
  Description of the UEM Manager, plus procedures and options specific for UNIX operating systems.
- UEM Manager Configuration Options (Chapter 6)
  Detailed information about the configuration options used with the UEM Manager component.
- Universal Event Monitor Server (Chapter 7)
  Description of the UEM Server, plus procedures and options specific for Windows and UNIX platforms
- Universal Event Monitor Server for Windows (Chapter 8)
  Description of the UEM Server, plus procedures and options specific for Windows operating systems.
- Universal Event Monitor Server for UNIX (Chapter 9)
  Description of the UEM Server, plus procedures and options specific for UNIX operating systems.
- UEM Server Configuration Options (Chapter 10)
  Detailed information about the configuration options used with the UEM Server component.
- Universal Event Monitor Component Definition Options (Chapter 11)
  Detailed information about the UEM component definition options.
- Universal Event Monitor UACL Entries (Chapter 12)
  Detailed information about the Universal Access Control List (UACL) entries available for use with Universal Event Monitor.
- UEMLoad Utility (Chapter 13)
  General description of the UEMLoad Utility.
- UEMLoad Utility for Windows (Chapter 14)
  Description of the UEMLoad Utility, plus procedures and options specific to Windows operating systems.
- UEMLoad Utility for UNIX (Chapter 15)
  Description of the UEMLoad Utility, plus procedures and options specific to UNIX operating systems.
- UEMLoad Utility Configuration Options (Chapter 16)
  Detailed information about the configuration options used with the UEMLoad utility.

- Additional Information for Universal Event Monitor (Chapter 17)
  Additional technical information used by or specific to UEM.
- Standard I/O Redirection and Event Handler Processes (Appendix A)
- Environment Variables Set by Universal Event Monitor (Appendix B)
- Customer Support (Appendix C)
  Customer support contact information for Universal Event Monitor.

CHAPTER 1

Overview

1.1 Introduction

Universal Event Monitor (UEM) provides a consistent, platform-independent means of monitoring one or more local or remote system events. It also can execute a system command or script based on the outcome of the events that it monitors.

One or more system events can be monitored by UEM at any given time. UEM requires only that any event of interest be presented to it in a well-defined manner.

The methods available for defining an event and its associated actions, and then presenting those definitions to UEM, are described in the following sections.

This document provides operating system-specific detailed technical information for Universal Event Monitor and UEMLoad:

- Usage
- Configuration Options
- Command line syntax
- Command references
- Component Definition options
- Universal Access Control List entries

For information how Universal Event Monitor is utilized, see the Indesca User Guide and/or Infitran User Guide.

1.2 Usage

The configuration options in this document are organized in the following way:

- Options are listed in separate chapters for separate UEM components:
  - UEM Manager
  - UEM Server
  - UEMLoad Utility
- Options are listed alphabetically in each chapter.
- Each option indicates for which operating systems (one or more) that it can be used.
- Each option identifies any operating-specific data or usage relative to that option

1.3 UEM Functionality

Use the Universal Event Monitor (UEM) Manager to monitor a single local or remote system event.

The UEM Manager (**`uem`**) may provide all of the parameters necessary to define a system event, or it may specify the ID of a database record that contains the event definition. In either case, the UEM Manager passes the event definition to a local or remote UEM Server (**`uemsrv`**), which uses that information to look for an occurrence of the event and test for its completion.

The UEM Manager may also provide all of the parameters necessary to define an event handler to the UEM Server, or it may specify the ID of a database record that contains the event handler. An event handler is a command or script that UEM Server executes, based on the outcome of the event occurrence.

A UEM Server may monitor several local system events simultaneously using records stored in its event definition database. An event-driven UEM Server executes in this manner. An event-driven UEM Server does not require a UEM Manager to initiate a monitoring request, and you may configure it to start automatically whenever the local Universal Broker starts. During start-up, an event-driven UEM Server retrieves a list of its assigned event definitions from the local Universal Broker. UEM Server monitors each event until it is no longer active, or until the event-driven Server ends.

The UEMLoad utility (**`uemload`**) enables you to add event definition and event handler records to their respective databases

UEMLoad handles all event definition and event handler database management tasks, including adds, updates, deletes, and lists / exports. UEMLoad forwards a database request to a UEM Server, which validates the information. The UEM Server then sends a request to a local Universal Broker to apply the requested operation to the appropriate UEM database file.

Figure 1.1, below, illustrates the interaction of the various components that make up Universal Event Monitor.

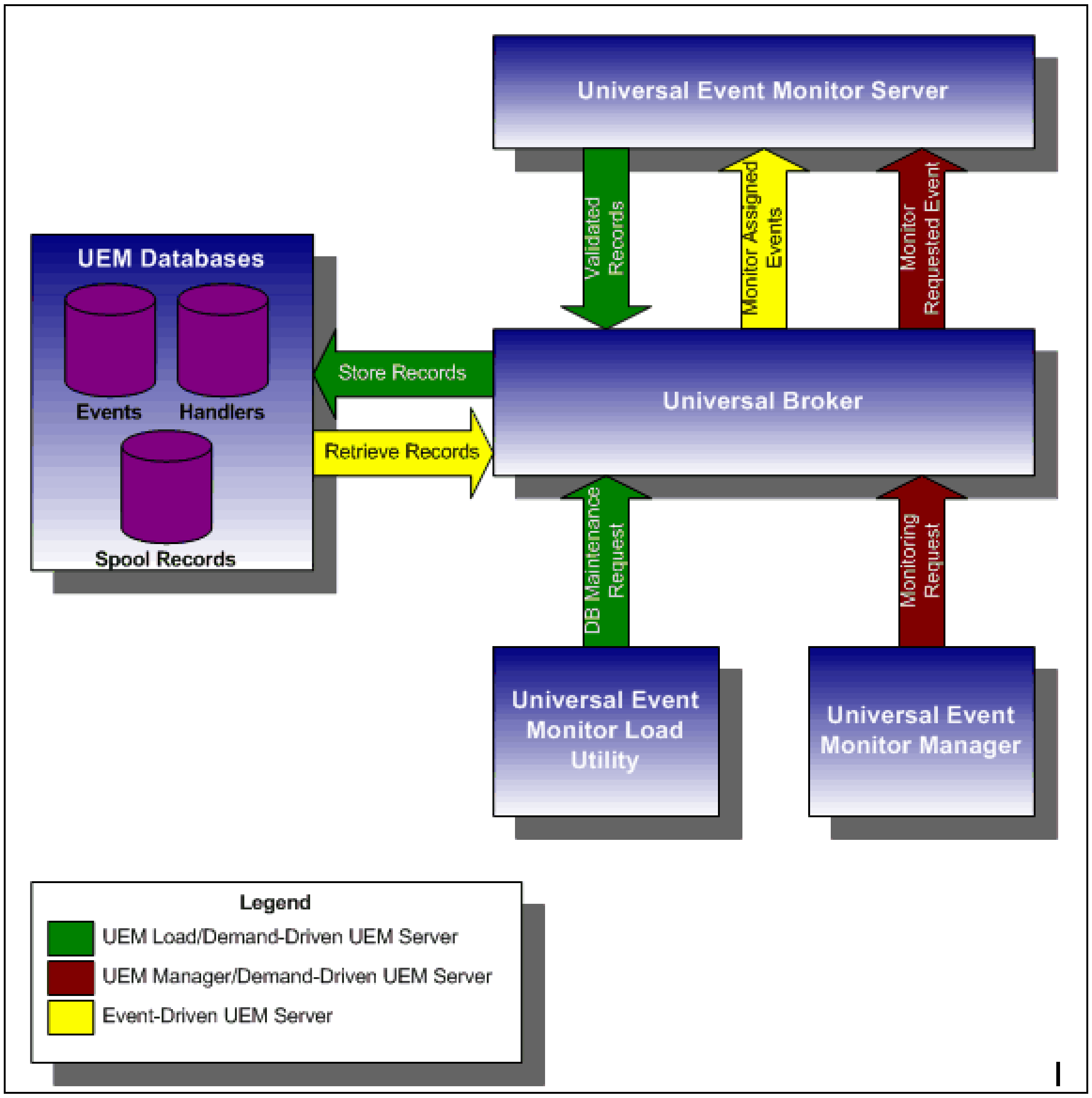


Figure 1.1 High-Level Interaction of UEM Components

1.4 Storing Event Definitions and Event Handlers

Event definitions and event handlers can be stored in separate BerkeleyDB database files. When an event definition or event handler record is added to its respective database, a unique identifier must be specified. Whenever UEM is required to monitor an event or execute an event handler, only this ID needs to be referenced in order for UEM to obtain the corresponding event definition or event handler parameters.

UEMLoad initiates all UEM-related database requests. UEMLoad is a command line application that can be used to:

- Add, update, and delete event definition and/or event handlers from their respective databases
- List the entire contents of the event definition and/or event handler databases
- List the parameters of a single event definition and/or event handler
- Export the contents of the event definition and/or event handler databases to a file that can be used to re-initialize the database or populate a new database on another system.

When UEMLoad is started, it sends a request to a Universal Broker running on the local system to start a UEM Server process. Because a client application (that is, UEMLoad) initiates the request, the UEM Server that is started is a demand-driven Server.

UEMLoad forwards the database request to the UEM Server, which validates it and supplies default values for any required parameters (based upon the type of request) that were not specified from the UEMLoad command line. When a set of complete, valid parameters is available, the UEM Server sends a request to the Universal Broker, which is responsible for actually performing the requested database operation.

Universal Broker reports the success or failure of all database maintenance requests (add, update, delete) to the UEM Server. The UEM Server then passes any errors back to UEMLoad.

For a database query request (list, export), Universal Broker will return the contents of each requested event definition or event handler record to the UEM Server, which then is responsible for forwarding the records to the UEMLoad.

Figure 1.2, below, illustrates the interaction of the Universal Broker and the Universal Event Monitor Server components involved during the execution of UEMLoad.

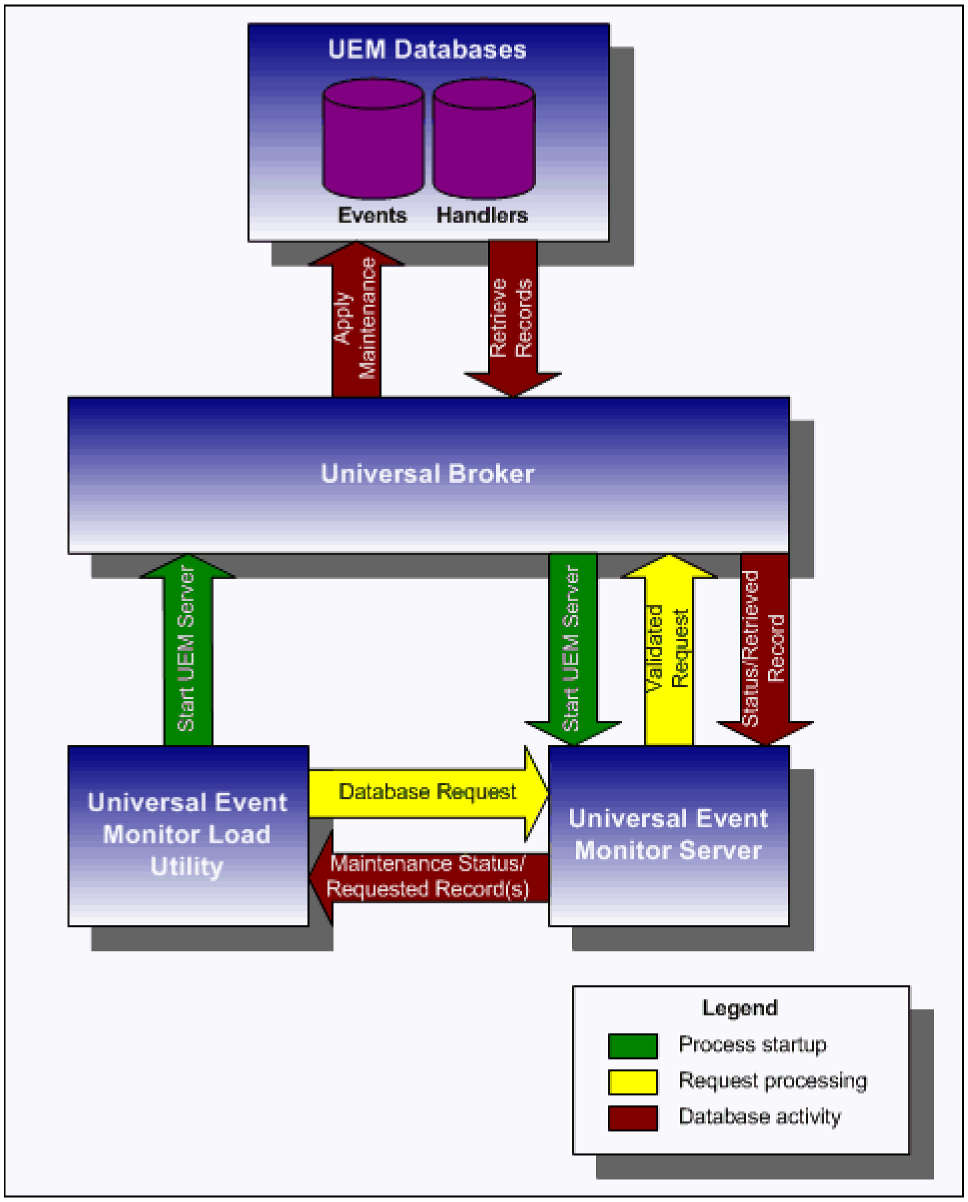


Figure 1.2 UEMLoad Utility Overview

1.5 Monitoring a Single Event

A single event can be monitored using the UEM Manager. The UEM Manager provides a command line interface from which all parameters required to define an event and its associated event handlers can be specified. In addition, the ID of a stored event definition or event handler can be used as an alternative to specifying all parameters explicitly.

When a UEM Manager is started, it sends a request to the specified local or remote Universal Broker to start a UEM Server. Because the request to start the UEM Server comes from a client application (that is, UEM Manager), it is a *demand-driven* Server that is started.

The UEM Manager sends the monitoring request to the UEM Server. The UEM Server validates the request and supplies default values for any required parameters that were not specified from the command line.

The UEM Manager command line provides for the assignment of an event handler to execute whenever the UEM Server sets the state of an event occurrence or state of the event itself. The UEM Server then is responsible for executing the assigned event handlers which are appropriate for the state change.

The UEM Server will monitor the event until either of the following conditions is satisfied:

- Required number of expected event occurrences has been detected
- Inactive date and time specified for the event definition elapses.

When either of these occurs, the event becomes inactive and the UEM Server stops monitoring it. The UEM Server then ends after informing the UEM Manager of the result of the monitoring request. The UEM Manager will set its exit code based on this information. This is the default behavior.

However, if an option was set in the UEM Manager instructing it to not wait on the UEM Server, the UEM Manager will end as soon as the UEM Server acknowledges its receipt of a valid monitoring request.

Figure 1.3, below, illustrates the interaction of the Universal Broker and the Universal Event Monitor components involved when a UEM Manager is executed.

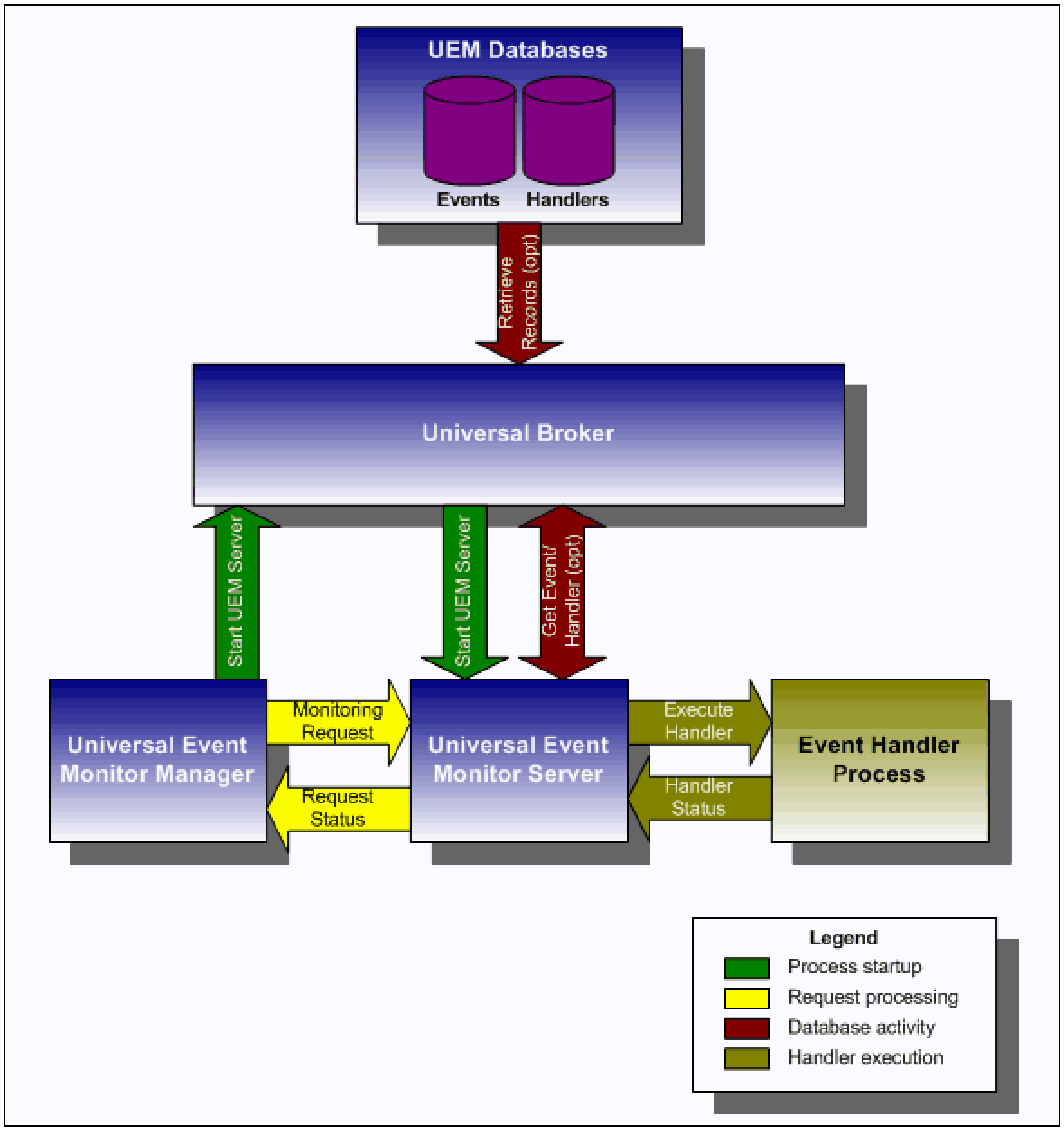


Figure 1.3 UEM Manager Overview

1.6 Monitoring Multiple Events

An *event-driven* Universal Event Monitor Server can be used to monitor multiple events at the same time. An event-driven UEM Server uses the records stored in the event definition database file to identify the events it is responsible for monitoring.

An event-driven UEM Server can be executed automatically during start-up of a Universal Broker. While it requires no interaction from a UEM client application, however, an event-driven UEM Server can be started at any time using Universal Control.

Unless it is stopped manually (using Universal Control), the event-driven UEM Server will continue to run as long as the Broker remains active. When the Broker stops, it will send a stop request to the UEM Server, instructing it to shut itself down.

When an event-driven UEM Server starts, it sends a request to the Broker asking for all of the event definitions residing in the event definition database that are assigned to that event-driven UEM Server. (This assignment was made when the event definition record was added to the database with UEMLoad.) The Server checks the active and inactive dates and times of the event definitions that it receives. It then begins monitoring the active events.

Each event definition provides for the assignment of an event handler to execute when an event occurrence is triggered or rejected. The assignment of an event handler to execute when an event expires also is made within the event definition. The UEM Server is responsible for executing appropriate event handlers based upon the states it sets for detected event occurrences and/or the event themselves.

Figure 1.4, below, illustrates the interaction of the Universal Broker and an event-driven UEM Server.

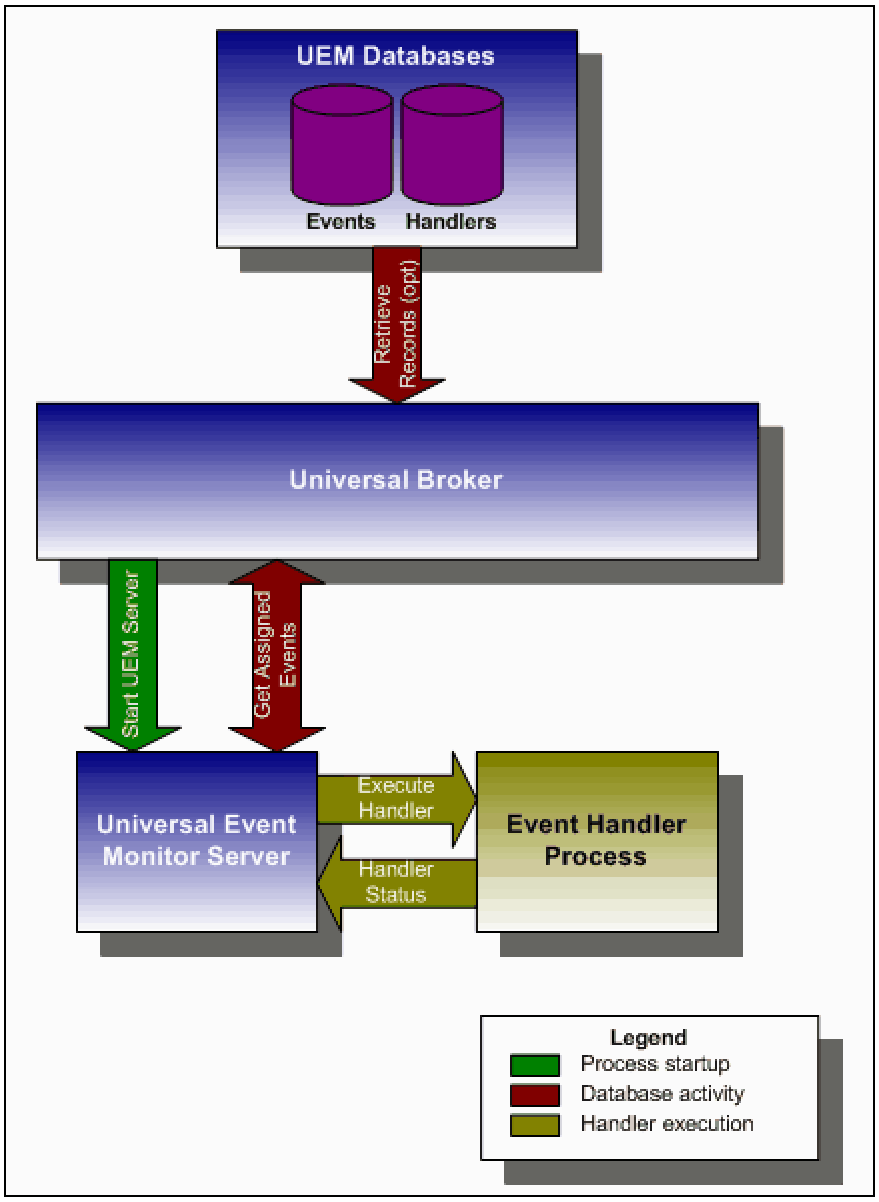


Figure 1.4 UEM Server Overview

CHAPTER 2

Universal Event Monitor Manager

2.1 Overview

The Universal Event Monitor Manager (**uem**) is provided for monitoring a single event. The parameters which define the system event and any actions that should be taken when the event satisfies certain conditions are specified via the UEM Manager's command options.

When a UEM Manager is started, it sends a request to the specified local or remote Universal Broker to start a Universal Event Monitor Server (**uemsrv**). The UEM Server that is started is a *demand-driven* Server, because it is executed *on demand* by a UEM Manager.

The UEM Manager sends the monitoring request to the UEM Server, which validates the provided parameters and supplies default values for any required parameters not specified.

After the request is validated and its receipt acknowledged by the UEM Server, the UEM Manager will wait (by default) for the UEM Server process to finish. This will occur when the following conditions are satisfied:

1. Event becomes inactive, which occurs either when:
   a. Required number of expected event occurrences are detected.
   b. Inactive date and time specified for the event elapses.
2. Any handler processes executed by the UEM Server have completed.

As noted in the conditions listed above, one or more handler processes can be executed by the UEM Server. The scenarios under which these processes are executed come from information provided by the UEM Manager when the actions to take for **trigger** or **reject** event occurrences are specified.

The actions to take when an event is set to an **expired** state also can be specified in the UEM Manager command statement.

Before it ends, the UEM Server may inform the UEM Manager of the result of the monitoring request. If it does, the UEM Manager will set its exit code based on this information. This is the default behavior. However, if an option was set in the UEM Manager instructing it to not wait on the UEM Server, the UEM Manager will end as soon as the UEM Server acknowledges its receipt of a valid monitoring request.

Detailed operating-specific usage information for UEM Manager, as well as detailed information on all UEM configuration options, is included in this document.

CHAPTER 3

Universal Event Monitor Manager for z/OS

3.1 Overview

This chapter provides information on Universal Event Monitor (UEM) Manager specific to the z/OS operating system.

UEM Manager monitors a single event on any computer running a UEM Server component. The event to monitor is described using an existing record in the event definition database or values supplied via command parameters. Likewise, parameters describing event handlers also can be supplied from an existing event handler record or from command line options.

A UEM Manager causes a demand-driven UEM Server component to be started on the target system. The UEM Server is classified as demand-driven because it is started upon demand by a UEM Manager.

A UEM Server is classified as event-driven when it is started automatically by the Universal Broker, without a request from a UEM Manager.

It is the demand-driven UEM Server that is responsible for monitoring the event and executing any processes on behalf of the specified event handlers. The UEM Manager may finish as soon as the UEM Server begins monitoring the event, or it may wait until the UEM Server completes, in which case the UEM Manager will receive status messages regarding monitoring activity.

3.2 Usage

UEM Manager for z/OS executes as a batch job.

This section describes the JCL, configuration and configuration options, and command line syntax of UEM Manager for z/OS.

3.2.1 JCL Procedure

Figure 3.1, below, illustrates the Universal Event Monitor for z/OS JCL procedure (**UEMPRC**, located in the **SUNVSAMP** library) that is provided to simplify the execution JCL and future maintenance.

```
//UEMPRC  PROC UPARM=,               -- UEM options
//             UEMPRE=#SHLQ.UNV
//*
//PS1      EXEC PGM=UEM,PARM='ENVAR(TZ=EST5EDT)/&UPARM'
//STEPLIB  DD  DISP=SHR,DSN=&UEMPRE..SUNVLOAD
//*
//UNVNLS   DD  DISP=SHR,DSN=&UEMPRE..SUNVNLS
//UNVTRACE DD  SYSOUT=*
//*
//SYSPRINT DD  SYSOUT=*
//SYSOUT   DD  SYSOUT=*
//CEEDUMP  DD  SYSOUT=*
```

Figure 3.1 UEM Manager for z/OS – JCL Procedure

The symbolic parameter **UPARM** is provided to allow EXEC PARM keyword values to be specified for the UEM program. The PARM values to the left of the slash (**/**) character are IBM Language Environment (LE) parameters. (See the Stonebranch Solutions 4.2.0 Installation Guide for information regarding the customization of Language Environment parameters.)

The **CONFIG** symbolic parameter can be used to specify the name of the PDS member in which persistent configuration options for UEM Manager reside.

The **UEMPRE** symbolic parameter specifies the data set name prefix of Stonebranch Solutions installation data sets.

3.2.2 DD Statements used in JCL Procedure

Table 3.1, below, describes the DD statements used in the Universal Event Manager for z/OS JCL procedure illustrated in Figure 3.1.

ddname	DCB Attributes *	Mode	Description
STEPLIB	DSORG=PO, RECFM=U	input	Stonebranch Solutions load library which contains the program to execute.
UNVNLS	DSORG=PO, RECFM=(F, FB, V, VB)	input	Stonebranch Solutions national language support library. Contains message catalogs and code page translation tables.
UNVTRACE	DSORG=PS, RECFM=(F, FB, V, VB)	output	UEM trace output.
SYSPRINT	DSORG=PS, RECFM=(F, FB, V, VB)	output	Standard output file for the UEM program. UEM does not write any messages to SYSPRINT.
SYSOUT	DSORG=PS, RECFM=(F, FB, V, VB)	output	Standard error file for the UEM program. UEM writes its messages to SYSOUT.
* The C runtime library determines the default DCB attributes. Refer to the IBM manual *OS/390 C/C++ Programming Guide* for details on default DCB attributes for stream I/O			

Table 3.1 UEM Manager for z/OS – DD Statements in JCL Procedure

DD Statement Categories

UEM Manager for z/OS DD statements are organized into the following categories:

- Runtime specifications
  STEPLIB, UNVCONF, and UNVNLS are in this category.
- UEM message and command files
  SYSPRINT, SYSOUT, and UNVTRACE are in this category.
- Command files
  User-defined DD statements to which Universal Event Monitor commands refer.

3.2.3 JCL

Figure 3.2, below, illustrates the Universal Event Monitor for z/OS JCL using the **UEMPRC** procedure illustrated in Figure 3.1.

```
//jobname JOB CLASS=A,MSGCLASS=X
//STEP1 EXEC UEMPRC
//ENCFILE DD DISP=SHR,DSN=UEM.UENCRYPT.USERINFO(MYHOST)
//SYSIN   DD *
 -event_type file -filespec “myfile*.dat” -host myhost
 -encryptedfile ENCFILE
/*
```

Figure 3.2 UEM Manager for z/OS – JCL

Job step STEP1 executes the procedure **UEMPRC**.

The command options are specified on the SYSIN DD.

In this example, a UEM Server is executed on the remote host **myhost** in order to detect the occurrence of a file that matches the file specification of **myfile*.dat**. The contents of **ENCFILE** include an encrypted user ID and password that are validated on the remote host.

3.2.4 Configuration

Configuration consists of:

- Setting default options and preferences for all executions of UEM Manager.
- Setting options and preferences for a single execution of UEM Manager.

UEM for z/OS receives its configuration options from the following sources:

1. PARM keyword
2. SYSIN DD statement
3. Command file
4. Configuration file

The order of precedence is the same as the list above; PARM keyword options being the highest and configuration file being the lowest. That is, options specified via a PARM keyword override options specified via a SYSIN ddname, and so on.

Detailed information on these methods of configuration can be found in the Configuration Management chapters of the Indesca and Infitran 4.2.0 User Guides.

Configuration File

The configuration file provides the simplest method of specifying configuration options whose values will not change with each command invocation. These default values are used if the options are not read from one or more other sources.

Some options only can be specified in the configuration file; they have no corresponding command line equivalent. Other options cannot be specified in the configuration file; they must be specified via one or more other sources for a single execution of UEM Manager.

For UEM Manager for z/OS, configuration options are placed in the configuration file that is referenced in the **`UEMPRC`** JCL procedure by the **`UNVCONF`** DD statement, member **`UEMCFG00`**.

The short and long forms of configuration options are used when an option is specified in the SYSIN DD statement.

- Long form consists of two or more case-insensitive characters; it is available for all command options.
- Short form consists of a single case-sensitive character; it is available for some command options.

3.2.5 Configuration Options

This section describes the configuration options used to execute UEM Manager for z/OS.

Configuration Options Categories

Table 3.2, below, categorizes the configuration options according to function.

Category	Description
Event Definition	System event to monitor.
Event Handler	Actions that should be taken when an event occurrence is triggered or rejected, or when an event expires.
Local	Options required for local broker registration.
Message	Universal Event Monitor message options.
Miscellaneous	Display command help and program versions.
Monitoring	Options that control how an event is monitored by the remote UEM Server.
Network	Transferring data between the local and remote systems.
Options	Alternative methods available for specifying command options.
Remote	Network address of the remote system and connection options.
User	Identify the user account with which monitoring activity is conducted on the remote system.

Table 3.2 UEM Manager for z/OS - Configuration Options Categories

The UEM Manager configuration options for each category are summarized in the following tables. Each **Option Name** is a link to detailed information about that option.

Event Definition Category Options

Option Name	Description
EVENT_ID	ID of a stored event definition record.
EVENT_TYPE	Type of event to monitor.
HANDLER_OPTIONS	Options that are passed as command line arguments to all processes executed on behalf of an event handler. See OPTIONS for handler-specific command line options.
INACTIVE_DATE_TIME	Date and time at which the state of the monitored event should be made “inactive.”
TRACKING_INTERVAL	Frequency, in seconds, with which a tracked event occurrence is tested for completeness.

Event Definition Category Options - Type-Specific

These options are specific to event definitions with an EVENT_TYPE of `FILE`.

Option Name	Description
FILE_SPECIFICATION	Name or pattern of the file whose creation should be detected and tracked for completion.
MINIMUM_FILE_SIZE	Smallest size a file may be in order for it to be considered complete.
RENAME_FILE	Flag that indicates whether or not a completed file should be renamed.
RENAME_FILE_SPECIFICATION	Name or pattern to use when a file is renamed.

Event Handler Category Options

Option Name	Description
EVENT_STATE	Event state that, when encountered, will result in the execution of the associated event handler.
HANDLER_ID	ID of a stored event handler record.
HANDLER_TYPE	Type of process to execute. Reserved for future integration with other Stonebranch Solutions applications.
MAXIMUM_RETURN_CODE	Highest return code that an event handler can exit with to be considered as having executed successfully.
OPTIONS	Values that are passed as command line arguments to a particular handler specified for a given EVENT_STATE.
USER_COMMAND	Complete path to an application file or remote script that should be executed on behalf of the event handler.
USER_SCRIPT	USER_SCRIPT DD statement that contains one or more system commands that should be executed on behalf of the event handler
USER_SCRIPT_TYPE	Type of script interpreter used to evaluate and execute the commands contained in USER_SCRIPT.

Local Category Options

Option Name	Description
SYSTEM_ID	Local Universal Broker with which the UEM Manager must register.

Message Category Options

Option Name	Description
MESSAGE_LANGUAGE	Language of messages written.
MESSAGE_LEVEL	Level of messages written.
TRACE_FILE_LINES	Maximum number of lines written to a trace file before it wraps around.
TRACE_TABLE	Memory trace table specification.

Miscellaneous Category Options

Option Name	Description
HELP	Write command option help.
VERSION	Write program version.

Monitoring Category Options

Option Name	Description
MAX_OCCURRENCE_COUNT	Maximum number of event occurrences to monitor.
POLLING_INTERVAL	Frequency with which the UEM Server will detect new occurrences of the system event. The UEM Server will also check at this time to see if the monitored event should be made inactive.
WAIT	Forces the UEM Manager to wait for the completion of the UEM Server.

Network Category Options

Option Name	Description
CODE_PAGE	Code page used for text translation.
CTL_SSL_CIPHER_LIST	SSL cipher list for the control session established between the UEM Manager and Server.
NETWORK_DELAY	Maximum number of seconds to wait for data communications.

Options Category Options

Option Name	Description
COMMAND_FILE_ENCRYPTED	Encrypted file that contains some command options.
COMMAND_FILE_PLAIN	Plain text file that contains some command options.
COMMAND_ID	ID that identifies unit of work represented by the UEM Manager and its associated UEM Server
ENCRYPTION_KEY	Optional encryption key used to decrypt the encrypted command file specified by the COMMAND_FILE_ENCRYPTED option.

Remote Category Options

Option Name	Description
CONNECT_TIMEOUT	Amount of time that a UEM Manager will wait for a connection to a remote Universal Broker to complete.
DNS_EXPAND	Number of IP addresses returned to UEM Manager following a DNS query issued to resolve a host name.
HOST_SELECTION	Host in the REMOTE_HOST list that the UEM Manager will choose to begin its attempts to connect to a remote Universal Broker.
HOSTNAME_RETRY_COUNT	Maximum number of attempts that will be made to establish a connection with the remote host.
OUTBOUND_IP	Host or IP address to use for all outgoing IP connections.
REMOTE_HOST	List of one or more hosts upon which a command may run.
REMOTE_PORT	TCP/IP port number on which the Universal Broker is accepting connections.

User Category Options

Option Name	Description
LOGIN	Instruction for the UEM Server to establish an execution environment for a user account.
USER_ID	ID of a remote user account that the UEM Server uses to establish the security context in which event monitoring is performed.
USER_PASSWORD	Password associated with USER_ID.

3.2.6 Command Line Syntax

Figure 3.3, below, illustrates the command line syntax — using the command line, long form of the configuration options — of UEM Manager for z/OS.

```
uem
-host hostlist
[-connect_timeout seconds]
[-dns_expand {yes|no}]
[-host_selection {sequential|random}]
[-port port]
[-system_id ID]
[-userid user [-pwd password] ]
[-login {yes|no}]
[-codepage codepage]
[-file dataset | -encryptedfile ddname [-key key] ]  *
[-cmdid id]
[-ctl_ssl_cipher_list cipherlist]
{-event_id id | -event_type type -filespec filespecification
   [-min_file_size size[{b|k|m|g}] ] [-rename_file {yes|no}]
   [-rename_filespec renamespecification] }
[ {-triggered | -rejected | -expired
   {-handler_id id | -cmd command | -script ddname [-script_type type] }
   [-options options] [-handler_type {cmd|script}] [-maxrc returncode] } ]
[-handler_opts options]
[-hostname_retry_count count]
[-inact_date_time time]
[-max_count count]
[-lang language]
[-level {trace|audit|info|warn|error}[,{time|notime}] ]
[-delay seconds]
[-outboundip host]
[-polling_int seconds]
[-tracefilelines lines]
[-trace_table size[{b|k|m|g}] [,{error|always|never}] ]
[-tracking_int seconds]
[-wait {yes|no}]

uem
{-help | -version}
```

* The command file (-file or -encryptedfile) can contain some or all required and/or optional configuration options, including -host. If a command file is specified on the command line, and it contains the required -host option, that option does not have to be specified additionally on the command line.

Figure 3.3 Universal Event Monitor Manager for z/OS - Command Line Syntax)

CHAPTER 4

Universal Event Monitor Manager for Windows

4.1 Overview

This chapter provides information on Universal Event Monitor (UEM) Manager specific to the Windows operating system.

A UEM Manager monitors a single event on any computer running a UEM Server component. The event to monitor is described using an existing record in the event definition database or values supplied via command line parameters. Likewise, parameters describing event handlers also can be supplied from an existing event handler record or from command line options.

A UEM Manager causes a demand-driven UEM Server component to be started on the target system. The UEM Server is classified as demand-driven because it is started upon demand by a UEM Manager. A UEM Server is classified as event-driven when it is started automatically by the Universal Broker, without a request from a UEM Manager.

It is the demand-driven UEM Server that is responsible for monitoring the event and executing any processes on behalf of the event handlers. The UEM Manager can finish as soon as the UEM Server begins monitoring the event, or it can wait until the UEM Server completes, in which case the UEM Manager will receive status messages regarding monitoring activity.

4.2 Usage

UEM Manager for Windows executes as a command line application.

This section describes the configuration, configuration options, and command line syntax of UEM Manager for Windows.

4.2.1 Configuration

Configuration consists of:

- Setting default options and preferences for all executions of UEM Manager.
- Setting options and preferences for a single execution of UEM Manager.

UEM for Windows receives its configuration options from the following sources:

1. Command line
2. Command file
3. Environment variables
4. Configuration file

The order of precedence is the same as the list above; command line options being the highest and configuration file being the lowest. That is, options specified via a command line override options specified a via command file, and so on.

Detailed information on these methods of configuration can be found in the Configuration Management chapters of the Indesca and Infitran 4.2.0 User Guides.

Configuration File

The configuration file provides the simplest method of specifying configuration options whose values will not change with each command invocation. These default values are used if the options are not read from one or more other sources.

Some options only can be specified in the configuration file; they have no corresponding command line equivalent. Other options cannot be specified in the configuration file; they must be specified via one or more other sources for a single execution of UEM Manager.

4.2.2 Configuration Options

This section describes the configuration options used to execute UEM Manager for Windows.

Configuration Options Categories

Table 4.1, below, categorizes the configuration options according to function.

Category	Description
Event Definition	Describe the system event to monitor.
Event Handler	Describe the actions that should be taken when an event occurrence is triggered or rejected, or when an event is set to an expired state.
Installation	Options that specify installation requirements, such as directory locations
Local	Options required for local broker registration.
Message	UEM message options.
Miscellaneous	Display command help and program versions.
Monitoring	Control how an event is monitored by the remote UEM Server.
Network	Transferring data between the remote and local systems.
Options	Alternative methods available for specifying command options.
Remote	Network address of the remote system and connection options.
User	User account with which monitoring activity is conducted on the remote system.

Table 4.1 UEM Manager for Windows - Configuration Options Categories

The UEM Manager configuration options for each category are summarized in the following tables. Each **Option Name** is a link to detailed information about that option.

Event Definition Category Options

Option Name	Description
EVENT_ID	ID of a stored event definition record.
EVENT_TYPE	Type of event to monitor.
HANDLER_OPTIONS	Options that are passed as command line arguments to all processes executed on behalf of an event handler. See OPTIONS for handler-specific command line options.
HANDLER_OPTIONS	Date and time at which the state of the monitored event should be made “inactive”.
TRACKING_INTERVAL	Frequency, in seconds, with which a tracked event occurrence is tested for completeness.

Event Definition Category Options - Type-Specific

These options are specific to event definitions with an EVENT_TYPE of **`FILE`**.

Option Name	Description
FILE_SPECIFICATION	Name or pattern of the file whose creation should be detected and tracked for completion.
MINIMUM_FILE_SIZE	Smallest size a file can be in order for it to be considered complete.
RENAME_FILE	Flag that indicates whether or not a completed file should be renamed.
RENAME_FILE_SPECIFICATION	Name or pattern to use when a file is renamed.

Event Handler Category Options

Option Name	Description
EVENT_STATE	Describes the event state that, when encountered, will result in the execution of the associated event handler.
HANDLER_ID	ID of a stored event handler record.
HANDLER_TYPE	Type of process to execute. Reserved for future integration with other Stonebranch Solutions applications.
MAXIMUM_RETURN_CODE	Highest return code that an event handler can exit with to be considered as having executed successfully.
OPTIONS	Values that are passed as command line arguments to a particular handler specified for a given EVENT_STATE.
USER_COMMAND	Complete path to an application file or remote script that should be executed on behalf of the event handler.
USER_SCRIPT	Complete path to a local script file that contains one or more system commands that should be executed on behalf of the event handler.
USER_SCRIPT_TYPE	Describes the type of script interpreter used to evaluate and execute the commands contained in USER_SCRIPT.

Installation Category Options

Option Name	Description
INSTALLATION_DIRECTORY	Directory in which UEM Manager is installed.

Message Category Options Summary

Option Name	Description
MESSAGE_LANGUAGE	Language of messages printed.
MESSAGE_LEVEL	Level of messages printed.
NLS_ DIRECTORY	Directory location of message catalog and code page tables
TRACE_FILE_LINES	Maximum number of lines written to a trace file before it wraps around.
TRACE_TABLE	Memory trace table specification.

Miscellaneous Category Options Summary

Option Name	Description
HELP	Write command option help.
VERSION	Write program version.

Monitoring Category Options Summary

Option Name	Description
MAX_OCCURRENCE_COUNT	Maximum number of event occurrences to monitor.
POLLING_INTERVAL	Frequency with which the UEM Server will detect new occurrences of the system event. The UEM Server will also check at this time to see if the monitored event should be made inactive.
WAIT	Forces the UEM Manager to wait for the completion of the UEM Server.

Network Category Options Summary

Option Name	Description
CODE_PAGE	Code page used for text translation.
CTL_SSL_CIPHER_LIST	SSL cipher list for the control session.
NETWORK_DELAY	Maximum number of seconds considered acceptable to wait for data communications.

Options Category Options Summary

Option Name	Description
COMMAND_FILE_ENCRYPTED	Encrypted file that contains command options.
COMMAND_FILE_PLAIN	Plain text file that contains command options.
COMMAND_ID	ID that identifies unit of work represented by the UEM Manager and its associated UEM Server
ENCRYPTION_KEY	Optional encryption key used to decrypt the encrypted user file specified by the COMMAND_FILE_ENCRYPTED option.

Remote Category Options Summary

Option Name	Description
CONNECT_TIMEOUT	Amount of time that a UEM Manager will wait for a connection to a remote Universal Broker to complete.
DNS_EXPAND	Number of IP addresses returned to UEM Manager following a DNS query issued to resolve a host name.
HOST_SELECTION	Host in the REMOTE_HOST list that the UEM Manager will choose to begin its attempts to connect to a remote Universal Broker.
HOSTNAME_RETRY_COUNT	Maximum number of attempts that will be made to establish a connection with the remote host.
OUTBOUND_IP	Host or IP address to use for all outgoing IP connections.
REMOTE_HOST	List of one or more hosts upon which a command may run.
REMOTE_PORT	TCP/IP port number on which the Universal Broker is accepting connections.

User Category Options Summary

Option Name	Description
LOGIN	Instruction for the UEM Server to establish an execution environment for a user account.
USER_ID	ID of a remote user account that the UEM Server uses to establish the security context in which event monitoring is performed.
USER_PASSWORD	Password associated with USER_ID.

4.2.3 Command Line Syntax

Figure 4.1, below, illustrates the command line syntax — using the command line, long form of the configuration options — of UEM Manager for Windows.

```
uem
-host hostlist
[-connect_timeout seconds]
[-dns_expand {yes|no}]
[-host_selection {sequential|random}]
[-port port]
[-userid user [-pwd password] ]
[-login {yes|no}]
[-codepage codepage]
[-file filename | -encryptedfile filename [-key key] ]  *
[-cmdid id]
[-ctl_ssl_cipher_list cipherlist]
{-event_id id | -event_type type -filespec filespecification
   [-min_file_size size[{b|k|m|g}] ] [-rename_file {yes|no}]
   [-rename_filespec renamespecification] }
[ {-triggered | -rejected | -expired
   {-handler_id id | -cmd command | -script filename [-script_type type] }
   [-options options] [-handler_type {cmd|script}] [-maxrc returncode] } ]
[-handler_opts options]
[-hostname_retry_count count]
[-inact_date_time time]
[-max_count count]
[-lang language]
[-level {trace|audit|info|warn|error}[,{time|notime}] ]
[-delay seconds]
[-outboundip host]
[-polling_int seconds]
[-tracefilelines lines]
[-trace_table size[{b|k|m|g}] [,{error|always|never}] ]
[-tracking_int seconds]
[-wait {yes|no}]

uem
{-help | -version}
```

* The command file (`-file` or `-encryptedfile`) can contain some or all required and/or optional configuration options, including `-host`. If a command file is specified on the command line, and it contains the required `-host` option, that option does not have to be specified additionally on the command line.

Figure 4.1 Universal Event Monitor Manager for Windows - Command Line Syntax

CHAPTER 5

Universal Event Monitor Manager for UNIX

5.1 Overview

This chapter provides information on Universal Event Monitor (UEM) Manager specific to the UNIX operating system.

A UEM Manager monitors a single event on any computer running a UEM Server component. The event to monitor is described using an existing record in the event definition database or values supplied via command parameters. Likewise, parameters describing event handlers also can be supplied from an existing event handler record or from command line options.

A UEM Manager causes a demand-driven UEM Server component to be started on the target system. The UEM Server is classified as demand-driven because it is started upon demand by a UEM Manager. A UEM Server is classified as event-driven when it is started automatically by the Universal Broker, without a request from a UEM Manager.

It is the demand-driven UEM Server that is responsible for monitoring the event and executing any processes on behalf of the event handlers. The UEM Manager may finish as soon as the Server begins monitoring the event, or it may wait until the UEM Server completes, in which case the Manager will receive status messages regarding monitoring activity.

5.2 Usage

UEM Manager for UNIX executes as a command line application.

This section describes the configuration, configuration options, and command line syntax of UEM Manager for UNIX.

5.2.1 Configuration

Configuration consists of:

- Setting default options and preferences for all executions of UEM Manager.
- Setting options and preferences for a single execution of UEM Manager.

UEM Manager for UNIX receives its configuration options from the following sources:

1. Command line
2. Command file
3. Environment variables
4. Configuration file

The order of precedence is the same as the list above; command line options being the highest and configuration file being the lowest. That is, options specified via a command line override options specified via a command file, and so on.

Detailed information on these methods of configuration can be found in the Configuration Management chapters of the Indesca and Infitran 4.2.0 User Guides.

Configuration File

The configuration file, **`uem.conf`**, provides the simplest method of specifying configuration options whose values will not change with each command invocation. These default values are used if the options are not read from one or more other sources.

Some options only can be specified in the configuration file; they have no corresponding command line equivalent. Other options cannot be specified in the configuration file; they must be specified via one or more other sources for a single execution of UEM Manager.

5.2.2 Configuration Options

This section describes the configuration options used to execute UEM Manager for UNIX.

Configuration Options Categories

Table 5.1, below, categories the configuration options according to function.

Category	Description
Event Definition	Describe the system event to monitor.
Event Handler	Describe the actions that should be taken when an event occurrence is triggered or rejected, or when an event is set to an expired state.
Installation	Options that specify installation requirements, such as directory locations
Local	Options required for local broker registration.
Message	UEM message options.
Miscellaneous	Display command help and program versions.
Monitoring	Control how an event is monitored by the remote UEM Server.
Network	Transferring data between the local and remote systems.
Options	Alternative methods available for specifying command options.
Remote	Network address of the remote system and connection options.
User	Identify the user account with which monitoring activity is conducted on the remote system.

Table 5.1 UEM Manager for UNIX - Configuration Options Categories

The UEM Manager configuration options for each category are summarized in the following tables. Each **Option Name** is a link to detailed information about that option.

Event Definition Category Options

Option Name	Description
EVENT_ID	ID of a stored event definition record.
EVENT_TYPE	Type of event to monitor.
HANDLER_OPTIONS	Options that are passed as command line arguments to all processes executed on behalf of an event handler. See OPTIONS for handler-specific command line options.
INACTIVE_DATE_TIME	Date and time at which the state of the monitored event should be made “inactive”.
TRACKING_INTERVAL	Frequency, in seconds, with which a tracked event occurrence is tested for completeness.

Event Definition Category Options - Type-Specific

These options are specific to event definitions with an EVENT_TYPE of **`FILE`**.

Option Name	Description
FILE_SPECIFICATION	Name or pattern of the file whose creation should be detected and tracked for completion.
MINIMUM_FILE_SIZE	Smallest size a file may be in order for it to be considered complete.
RENAME_FILE	Flag that indicates whether or not a completed file should be renamed.
RENAME_FILE_SPECIFICATION	Name or pattern to use when a file is renamed.

Event Handler Category Options

Option Name	Description
EVENT_STATE	Describes the event state that, when encountered, will result in the execution of the associated event handler.
HANDLER_ID	ID of a stored event handler record.
HANDLER_TYPE	Type of process to execute. Reserved for future integration with other Stonebranch Solutions applications.
MAXIMUM_RETURN_CODE	Highest return code that an event handler may exit with to be considered as having executed successfully.
OPTIONS	Values that are passed as command line arguments to a particular handler specified for a given EVENT_STATE.
USER_COMMAND	Complete path to an application file or script that should be executed on behalf of the event handler.
USER_SCRIPT	Complete path to a local script file that contains one or more system commands that should be executed on behalf of the event handler.
USER_SCRIPT_TYPE	Describes the type of script interpreter used to evaluate and execute the commands contained in USER_SCRIPT_TYPE.

Installation Category Options

Option Name	Description
INSTALLATION_DIRECTORY	Directory in which UEM Manager is installed.

Local Category Options

Option Name	Description
BIF_DIRECTORY	Broker Interface File (BIF) directory where the Universal Broker interface file is located.
PLF_DIRECTORY	Program Lock File (PLF) directory where the program lock files are located.

Message Category Options

Option Name	Description
MESSAGE_LANGUAGE	Language of messages printed.
MESSAGE_LEVEL	Level of messages printed.
NLS_ DIRECTORY	Directory location of message catalog and code page tables
TRACE_FILE_LINES	Maximum number of lines written to a trace file before it wraps around.
TRACE_TABLE	Memory trace table specification.

Miscellaneous Category Options

Option Name	Description
HELP	Write command option help.
VERSION	Write program version.

Monitoring Category Options

Option Name	Description
MAX_OCCURRENCE_COUNT	Maximum number of event occurrences to monitor.
POLLING_INTERVAL	Frequency with which the UEM Server will detect new occurrences of the system event. The UEM Server will also check at this time to see if the monitored event should be made inactive.
WAIT	Instructs the UEM Manager to wait for the completion of the UEM Server.

Network Category Options

Option Name	Description
CODE_PAGE	Code page used for text translation.
CTL_SSL_CIPHER_LIST	SSL cipher list for the control session.
NETWORK_DELAY	Maximum number of seconds considered acceptable to wait for data communications.

Options Category Options

Option Name	Description
COMMAND_FILE_ENCRYPTED	Encrypted file that contains command options.
COMMAND_FILE_PLAIN	Plain text file that contains command options.
COMMAND_ID	ID that identifies unit of work represented by the UEM Manager and its associated UEM Server
ENCRYPTION_KEY	Optional encryption key used to decrypt the encrypted user file specified by the COMMAND_FILE_ENCRYPTED option.

Remote Category Options

Option Name	Description
CONNECT_TIMEOUT	Amount of time that a UEM Manager will wait for a connection to a remote Universal Broker to complete.
DNS_EXPAND	Number of IP addresses returned to UEM Manager following a DNS query issued to resolve a host name.
HOST_SELECTION	Host in the REMOTE_HOST list that the UEM Manager will choose to begin its attempts to connect to a remote Universal Broker.
HOSTNAME_RETRY_COUNT	Maximum number of attempts that will be made to establish a connection with the remote host.
OUTBOUND_IP	Host or IP address to use for all outgoing IP connections.
REMOTE_HOST	List of one or more hosts upon which a command may run.
REMOTE_PORT	TCP/IP port number on which the Universal Broker is accepting connections.

User Category Options

Option Name	Description
LOGIN	Instruction for the UEM Server to establish an execution environment for a user account.
USER_ID	ID of a remote user account that the UEM Server uses to establish the security context in which event monitoring is performed.
USER_PASSWORD	Password associated with USER_ID.

5.2.3 Command Line Syntax

Figure 5.1, below, illustrates the command line syntax — using the command line, long form of the configuration options — of UEM Manager for UNIX.

```
uem
-host hostlist
[-connect_timeout seconds]
[-dns_expand {yes|no}]
[-host_selection {sequential|random}]
[-port port]
[-userid user [-pwd password] ]
[-login {yes|no}]
[-bif_directory directory]
[-plf_directory directory]
[-codepage codepage]
[-file filename | -encryptedfile filename [-key key] ]  *
[-cmdid id]
[-ctl_ssl_cipher_list cipherlist]
{-event_id id | -event_type type -filespec filespecification
   [-min_file_size size[{b|k|m|g}] ] [-rename_file {yes|no}]
   [-rename_filespec renamespecification] }
[ {-triggered | -rejected | -expired
   {-handler_id id | -cmd command | -script filename [-script_type type] }
   [-options options] [-handler_type {cmd|script}] [-maxrc returncode] } ]
[-handler_opts options]
[-hostname_retry_count count]
[-inact_date_time time]
[-max_count count]
[-lang language]
[-level {trace|audit|info|warn|error}[,{time|notime}] ]
[-delay seconds]
[-outboundip host]
[-polling_int seconds]
[-tracefilelines lines]
[-trace_table size[{b|k|m|g}] [,{error|always|never}] ]
[-tracking_int seconds]
[-wait {yes|no}]

uem
{-help | -version}
```

Figure 5.1 Universal Event Monitor Manager for UNIX - Command Line Syntax

CHAPTER 6

UEM Manager Configuration Options

6.1 Overview

This chapter provides detailed information on the configuration options available for use with the Universal Event Monitor Manager.

The options are listed alphabetically, without regard to any specific operating system.

Section 6.2 Configuration Options Information provides a guideline for understanding the information presented or each option.

6.2 Configuration Options Information

For each configuration option, this chapter provides the following information.

Description

Describes the configuration option and how it is used.

Usage

Provides a table of the following information:

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	<Format / Value>					
Command Line, Long Form	<Format / Value>					
Environment Variable	<Format / Value>					
Configuration File Keyword	<Format / Value>					

Method

Identifies the different methods used to specify Universal Event Monitor Manager configuration options:

- Command Line Option, Short Form
- Command Line Option, Long Form
- Environment Variable
- Configuration File Keyword

Note: Each option can be specified using one or more methods.

Syntax

Identifies the syntax of each method that can be used to specify the option:

• Format	Specific characters that identify the option.
• Value	Type of value(s) to be supplied for this method.

Note: If a Method is not valid for specifying the option, the Syntax field contains **n/a**.

(Operating System)

Identifies (with a ✓) the operating systems for which each method of specifying the option is valid:

- IBM i
- NonStop (HP NonStop)
- UNIX
- Windows
- z/OS

Values

Identifies all possible values for the specified value type.

Defaults are identified in **[bracketed bold type]**.

<Additional Information>

Identifies any additional information specific to the option.

6.3 Configuration Options List

Table 6.1, below, identifies all UEM Manager configuration options.

Option	Description	Page
BIF_DIRECTORY	Broker Interface Directory that specifies the location of the Universal Broker interface file	70
CODE_PAGE	Code page used for text translation.	71
COMMAND_FILE_ENCRYPTED	Encrypted file that contains some command options.	72
COMMAND_FILE_PLAIN	Plain text file that contains some command options.	73
COMMAND_ID	Identifier, saved by the Universal Broker, that is used to identify the unit of work represented by the UEM Manager and its associated UEM Server component.	74
CONNECT_TIMEOUT	Amount of time that a UEM Manager will wait for a connection to a remote Universal Broker to complete.	75
CTL_SSL_CIPHER_LIST	SSL cipher list for the control session established between the UEM Manager and Server.	77
DNS_EXPAND	Number of IP addresses returned to UEM Manager following a DNS query issued to resolve a host name.	78
ENCRYPTION_KEY	Optional encryption key used to decrypt the encrypted command file specified by the COMMAND_FILE_ENCRYPTED option.	80
EVENT_ID	ID of a stored event definition record.	81
EVENT_STATE	Event state that, when encountered, will result in the execution of the associated event handler.	82
EVENT_TYPE	Type of event to monitor.	83
FILE_SPECIFICATION	Name or pattern of the file whose creation should be detected and tracked for completion.	84
HANDLER_ID	ID of a stored event handler record.	86
HANDLER_OPTIONS	Options that are passed as command line arguments to any process executed on behalf of an event handler.	87
HANDLER_TYPE	Type of process to execute. Used primarily for integration with other Stonebranch Solutions applications.	89
HELP	Displays command option help	91
HOST_SELECTION	Host in the REMOTE_HOST list that the UEM Manager will choose to begin its attempts to connect to a remote Universal Broker.	92
HOSTNAME_RETRY_COUNT	Maximum number of attempts that will be made to establish a connection with the remote host.	93
INACTIVE_DATE_TIME	Date and time at which the state of the monitored event should be made **inactive**.	94
INSTALLATION_DIRECTORY	Base directory in which UEM Server is installed	96
LOGIN	Instructs the UEM Server to establish an execution environment for the user account	97
MAX_OCCURRENCE_COUNT	Maximum number of event occurrences to monitor.	98

Option	Description	Page
MAXIMUM_RETURN_CODE	Highest return code that an event handler may exit with to be considered as having executed successfully.	99
MESSAGE_LANGUAGE	Language of messages written.	100
MESSAGE_LEVEL	Level of messages Written.	101
MINIMUM_FILE_SIZE	Smallest size a file may be in order for it to be considered complete.	103
NETWORK_DELAY	Maximum number of seconds to wait for data communications.	104
NLS_DIRECTORY	Directory location of message catalog and code page tables	105
OPTIONS	Values that are passed as command line arguments to a particular handler specified for a given EVENT_STATE.	106
OUTBOUND_IP	Host or IP address to use for all outgoing IP connections.	108
PLF_DIRECTORY	Program Lock File directory that specifies the location of the UEM Manager program lock file	109
POLLING_INTERVAL	Frequency with which the UEM Server will: • Detect any new occurrences of the system event • See if the monitored event should be made **inactive**.	110
REMOTE_HOST	List of one or more hosts upon which a command may run.	111
REMOTE_PORT	TCP/IP port number on which the Universal Broker is accepting connections.	113
RENAME_FILE	Flag that indicates whether or not a completed file should be renamed.	114
RENAME_FILE_SPECIFICATION	Name or pattern to use when a file is renamed.	115
SYSTEM_ID	Local Universal Broker with which the Universal Event Monitor Manager must register	117
TRACE_FILE_LINES	Maximum number of lines written to a trace file before it wraps around.	118
TRACE_TABLE	Memory trace table specification.	119
TRACKING_INTERVAL	Frequency, in seconds, with which a tracked event occurrence is tested for completeness.	121
USER_COMMAND	Complete path to an application file or remote script that should be executed on behalf of the event handler.	122
USER_ID	ID of a remote user account that the UEM Server uses to establish the security context in which event monitoring is performed.	124
USER_PASSWORD	Password associated with USER_ID.	125
USER_SCRIPT	Complete path to a local script file or DD statement that contains one or more system commands that should be executed on behalf of the event handler.	126
USER_SCRIPT_TYPE	Type of script interpreter used to evaluate and execute the commands contained in USER_SCRIPT.	129
VERSION	Displays program version	130
WAIT	Forces the UEM Manager to wait for the completion of the UEM Server.	131

Table 6.1 UEM Manager Configuration Options

6.4 BIF_DIRECTORY

Description

The BIF_DIRECTORY option specifies the Broker Interface File (BIF) directory where the Universal Broker interface file, **`ubroker.bif`**, is located.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-bif_directory *directory*			√		
Environment Variable	UEMBIFDIRECTORY=*directory*			√		
Configuration File Keyword	n/a					

Values

directory is the name of the BIF directory.

[Default is `/var/opt/universal`.]

6.5 CODE_PAGE

Description

The CODE_PAGE option specifies the character code page that is used to translate text data received and transmitted over the network.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	-t *codepage*			√	√	√
Command Line, Long Form	-codepage *codepage*			√	√	√
Environment Variable	UEMCODEPAGE=*codepage*			√	√	
Configuration File Keyword	codepage *codepage*			√	√	√

Value

codepage is the character code page that is used to translate data.

codepage references a Universal Translate Table (UTT) file provided with the product (see Section 17.3 UTT Files for information on UTT files). UTT files are used to translate between Unicode and the local single-byte code page. (All UTT files end with an extension of `.utt`.)

[Default

The default is different for different operating systems:

• **ISO8859-1 (8-bit ASCII)**	**ASCII-based operating systems**
• **IBM1047 (EBCDIC)**	**Non-IBM i, EBCDIC-based operating system]**

See Section 17.2 Character Code Pages for a complete list of character code pages provided by Stonebranch Inc. for use with Stonebranch Solutions.

6.6 COMMAND_FILE_ENCRYPTED

Description

The COMMAND_FILE_ENCRYPTED option specifies a data set (for z/OS) or file (for Windows and UNIX) that contains encrypted values for command parameters. Storing these parameters and their associated values in an encrypted data set / file can be used in situations where it is not desirable to explicitly specify them on the command line.

The parameters contained in the data set / file must be in their respective command line formats.

The data set referenced by the DD statement identified by *ddname* must be encrypted with the Universal Encrypt utility. (For detailed information on the Universal Encrypt utility, see the Stonebranch Solutions Utilities 4.2.0 Reference Guide.)

It is strongly recommended that any command data set / file containing sensitive data (including those that are encrypted) be further protected from unauthorized access using a native operating system security method, such as RACF.

Note: If the data set / file name is specified in this option, it should not be specified additionally with the COMMAND_FILE_PLAIN option. If it is, command options contained in the data set / file name identified by the COMMAND_FILE_PLAIN will be used.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	-x *ddname* or *filename*			√	√	√
Command Line, Long Form	-encryptedfile *ddname* or *filename*			√	√	√
Environment Variable	n/a					
Configuration File Keyword	n/a					

Value

ddname (for z/OS) or *filename* (for UNIX and Windows) is the name of the data set or file, respectively, containing the parameters and their encrypted values.

6.7 COMMAND_FILE_PLAIN

Description

The COMMAND_FILE_PLAIN option specifies a data set (for z/OS) or local file (for UNIX and Windows) than contains command parameters. Storing these parameters and their associated values in a data set / file can be used in situations where it is not desirable to specify them explicitly on the command line.

The parameters contained in the data set / file must be in their respective command line formats.

The data set / file specified by this option is a plain text file. It is strongly recommended that this file be further protected from unauthorized access using a native operating system security method, such as RACF.

Note: If the data set / file name is specified in this option, it should not be specified additionally with the COMMAND_FILE_ENCRYPTED option. If it is, user information contained in the data set / file name file identified in this COMMAND_FILE_PLAIN option will be used.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	-f *data set* or *filename*			√	√	√
Command Line, Long Form	-file *data set* or *filename*			√	√	√
Environment Variable	n/a					
Configuration File Keyword	n/a					

Value

ddname (for z/OS) or *filename* (for UNIX and Windows) is the name of the data set or file, respectively, containing the parameters and their values.

6.8 COMMAND_ID

Description

The COMMAND_ID option specifies an identifier that is saved by the Universal Broker and which is used to identify the unit of work represented by the UEM Manager and its associated UEM Server component.

If this option is not used, UEM provides a default command ID. This default depends on how an event's description is supplied to UEM:

- If an event's description is provided via a stored event definition record, the ID of the record, specified by the EVENT_ID option, is used for the command ID.
- If an event's description is provided entirely from command line parameters, the default command ID depends on the type of event monitored.
- For events with an event type of **`FILE`** (see EVENT_TYPE), the name of the file specified by the FILE_SPECIFICATION option is used.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	-C *id*			√	√	√
Command Line, Long Form	-cmdid *id*			√	√	√
Environment Variable	UEMCMDID=*id*			√	√	
Configuration File Keyword	n/a					

Value

id is any value.

If *id* contains spaces, it must be enclosed in double (**"**) quotation marks.

UNIX and z/OS

If *id* contains spaces, it must be enclosed in either single (**'**) or double (**"**) quotation marks.

6.9 CONNECT_TIMEOUT

Description

The CONNECT_TIMEOUT option specifies how long a Universal Event Monitor Manager will wait for a connection to a remote Universal Broker to complete.

CONNECT_TIMEOUT is particularly helpful when more than one host is specified for the REMOTE_HOST option. By default, connection timeouts are controlled by the TCP/IP stack. Depending on this value, it may take several minutes to process a list of hosts before a connection actually succeeds. Setting a CONNECT_TIMEOUT value allows connection attempts to unreachable Universal Brokers to fail quickly, decreasing the time required to process a list of one or more hosts.

Note: CONNECT_TIMEOUT is most beneficial when set to a value that is less than the TCP/IP stack's default timeout, which is implementation dependent. A relatively small CONNECT_TIMEOUT value is recommended, to make sure it — and not the TCP/IP default — is applied.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-connect_timeout *seconds*			√	√	√
Environment Variable	UEMCONNECTTIMEOUT=*seconds*			√	√	
Configuration File Keyword	connect_timeout *seconds*			√	√	√

Values

seconds is the time, in seconds, that the UEM Manager will wait for a connection to a Universal Broker to complete. This value applies to each host contained in the resolved, expanded, and scrubbed REMOTE_HOST list.

Valid values for *seconds* are 0 (zero) to 300.

[Default is *0*.]
(This mean that each connection attempt will use the implementation-defined TCP/IP timeout value. This is the behavior of connection attempts prior to version 3.2.0 of Universal Event Monitor.)

If the time specified by *seconds* elapses before a successful connection to a Universal Broker on the specified system is established, the UEM Manager will select the next host in the list. If no more hosts are available, the application will end with an error to indicate that no connection was made.

Note: It is possible for the total time required to attempt connections to all hosts in the REMOTE_HOST list to exceed the number of seconds specified in this option.

6.10 CTL_SSL_CIPHER_LIST

Description

The CTL_SSL_CIPHER_LIST option specifies the acceptable and preferred SSL cipher suites to use for the control session between UEM components. The SSL protocol uses the cipher suites to specify which encryption and message authentication (or message digest) algorithms to use.

The UEM Manager can request one or more SSL ciphers, listed in order of preference. The list is forwarded to the UEM Server, which compares it to a list of SSL ciphers it is capable of accepting, and the first agreed-upon cipher is chosen.

Usage

Method Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-ctl_ssl_cipher_list *cipherlist*			√	√	√
Environment Variable	UEMCTLSSLCIPHERLIST=*cipherlist*			√	√	
Configuration File Keyword	ctl_ssl_cipher_list *cipherlist*			√	√	√

Values

cipherlist is a comma-separated list of SSL cipher suites. The list should be ordered with the most preferred suite first and the least preferred suite last.

Valid *list* values are:

• RC4-SHA	128-bit RC4 encryption and SHA-1 message digest
• RC4-MD5	128-bit RC4 encryption and MD5 message digest
• AES256-SHA	256-bit AES encryption and SHA-1 message digest
• AES128-SHA	128-bit AES encryption and SHA-1 message digest
• DES-CBC3-SHA	128-bit Triple-DES encryption and SHA-1 message digest
• DES-CBC-SHA	128-bit DES encryption and SHA-1 message digest

[Default is RC4-SHA,RC4-MD5,AES256-SHA,AES128-SHA,DES-CBC3-SHA, DES-CBC-SHA]

6.11 DNS_EXPAND

Description

The DNS_EXPAND option specifies how many IP addresses are returned to UEM Manager following a DNS query, which is issued to resolve a host name.

If the UEM Manager is configured to expand the results of the query, all IP addresses defined for a particular host name are returned and expanded (in-place) within the list of hosts specified for the REMOTE_HOST option. Otherwise, only the first host is returned, and no expansion is performed.

For example, if a host list contains six host names, and the name in the 3rd position resolves to five IP addresses, those addresses will occupy positions 3-7 in the expanded list. Subsequent hosts specified by the user will begin at the 8th position in the expanded list.

That is:

- If the REMOTE_HOST list contains:
  -host *host1,host2,host3,host4,host5,host6*
- And *host3* resolves to:
  ip3a, ip3b, ip3c, ip3d, and ip3e
- Then after the other hosts are resolved, the list will be expanded to:
  -host *ip1,ip2,ip3a,ip3b,ip3c,ip3d,ip3e,ip4,ip5,ip6*

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-dns_expand *option*			√	√	√
Environment Variable	UEMDNSEXPAND=*option*			√	√	
Configuration File Keyword	dns_expand *option*			√	√	√

Values

option specifies whether or not DNS query results are expanded.

Valid values for *option* are:

- **yes**
  All IP addresses returned by a DNS for a given query are returned.
- **no**
  Only the first IP address returned by a DNS for a given query is returned.

[Default is no.]

6.12 ENCRYPTION_KEY

Description

The ENCRYPTION_KEY option specifies the key that is used to decrypt the data set (for z/OS) or file (for Windows and UNIX) specified by the COMMAND_FILE_ENCRYPTED option. The key is required only if a key was specified when the command file was encrypted with Universal Encrypt.

A matching key must be provided to decrypt the file; otherwise, the decryption will fail. If no key is specified, a default key established by the UEM Manager is used.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	-K *key*			√	√	√
Command Line, Long Form	-key *key*			√	√	√
Environment Variable	UEMKEY=*key*			√	√	
Configuration File Keyword	n/a					

Value

key is the key used to decrypt the data set / file.

6.13 EVENT_ID

Description

The EVENT_ID option specifies the ID of an existing record stored in the event definition database. This record contains all of the parameters that are necessary to describe the event to monitor.

If this option is not used, the event, along with any additional parameters that it requires, must be described in the EVENT_TYPE option.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-event_id *id*			√	√	√
Environment Variable	n/a					
Configuration File Keyword	n/a					

Value

id is the ID of the existing record in the event definition database.

6.14 EVENT_STATE

Description

The EVENT_STATE option marks the beginning of a set of command line options that describes UEM Server's response to an event that enters the specified state.

When this option is used, a corresponding HANDLER_ID, USER_COMMAND, or USER_SCRIPT option also must be provided.

There are three different event states:

- Triggered
  An occurrence of a monitored event was completed.
- Rejected
  An occurrence of a monitored event was detected, but failed to complete before the date and time specified by the event's INACTIVE_DATE_TIME elapsed.
- Expired
  Date and time specified by an event's INACTIVE_DATE_TIME elapsed, with no occurrence of the monitored event detected.

The UEM Server may repeat the action it takes in response to Triggered and Rejected events if the Server detects more than one occurrence of a monitored event.

By definition, there are no event occurrences for an Expired event. UEM Server executes actions in response to such events just once.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-triggered | -rejected | -expired			√	√	√
Environment Variable	n/a					
Configuration File Keyword	n/a					

Value

(There are no values for these parameters.)

6.15 EVENT_TYPE

Description

The EVENT_TYPE option specifies the type of an event to monitor.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-event_type *type*			√	√	√
Environment Variable	n/a					
Configuration File Keyword	n/a					

Value

type is the event type to monitor.

Table 6.2 identifies valid UEM event types and UEM Manager configuration options associated with these event types.

Note: Currently, there is only one valid UEM event type: *FILE*.

Event Type	Description	Associated Options
FILE	Detects the creation of a file and monitors it, testing it for completeness.	• FILE_SPECIFICATION * • MINIMUM_FILE_SIZE • RENAME_FILE • RENAME_FILE_SPECIFICATION
* FILE_SPECIFICATION is required; it is the only option required for the *FILE* event type.		

Table 6.2 UEM Event Types

6.16 FILE_SPECIFICATION

Description

The FILE_SPECIFICATION option specifies the name of the file whose presence should be detected and monitored by UEM.

Additionally, FILE_SPECIFICATION also can specify an absolute path or a path relative to the working directory defined in the UEM Server's component definition.

Note: FILE_SPECIFICATION is valid only for events with an EVENT_TYPE of **`FILE`**.

Stoneman's Tip

If a Windows UEM Manager has issued a request to monitor an event on a UNIX system, and a complete path to the file is specified file specification, a leading space must be added to the path name and it must be enclosed in double quotes.

For example, if the UEM Server is asked to look for a file named **`file.ext`** in **`/home/user`**, the value specified for *filespecification* must be **`"/home/user/file.exe"`**. This requirement is a result of the way that command line arguments are read by a Windows UEM Manager, where a forward slash (**/**) also can be used as a command option prefix. The leading space allows the string beginning with the **/** to be interpreted as a option value.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-filespec *filespecification*			√	√	√
Environment Variable	n/a					
Configuration File Keyword	n/a					

Value

filespecification is the name of the path or file to be monitored by UEM.

If the path or file name contains spaces, it must be enclosed in double (**"**) quotation marks.

UNIX and z/OS

If the path or file name contains spaces, it must be enclosed in either single (**'**) or double (**"**) quotation marks.

Wildcards

filespecification also can include the following wildcards:

- * Match zero, one or more characters.
- ? Match zero or one character.

6.17 HANDLER_ID

Description

The HANDLER_ID option specifies the actions that should be taken whenever an event occurrence or an event itself enters the corresponding event state (see EVENT_STATE).

If the desired actions are not defined in an existing event handler record, the USER_COMMAND or USER_SCRIPT options can be used to define an event handler entirely from the command line.

USER_SCRIPT cannot be used with the USER_COMMAND or USER_SCRIPT options for a specified EVENT_STATE.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-handler_id *id*			√	√	√
Environment Variable	n/a					
Configuration File Keyword	n/a					

Value

id is the unique identifier of an existing record in the event handler database.

6.18 HANDLER_OPTIONS

Description

The HANDLER_OPTIONS option specifies a value that UEM Server passes as a command line argument to the process it executes on behalf of an event handler.

Any value specified for HANDLER_OPTIONS works in conjunction with — not as a replacement of — any value specified by the OPTIONS option. UEM Server uses the values of both options to construct a list of one or more command line arguments. Any value specified for HANDLER_OPTIONS always follows the value specified for OPTIONS.

HANDLER_OPTIONS allows an event definition to control handler process behavior, regardless of EVENT_STATE. That is, UEM Server appends the HANDLER_OPTIONS value to any command it executes on behalf of a monitored event, even if that event specifies three different handlers for its triggered, rejected, and expired states.

For example, you may find it useful to specify a HANDLER_OPTIONS value that causes all event handler processes executed on a particular event's behalf to redirect output to a single file.

HANDLER_OPTIONS also provides the ability to customize a stored event handler’s behavior.

For example, if an event handler record contains the following information:

```
HANDLER_ID optsexample
USER_SCRIPT “C:\UEMScripts\somescript.bat”
OPTIONS “-dirname C:\Program Files\Universal”
```

And the UEM Manager specifies the following HANDLER_OPTIONS value:

```
HANDLER_OPTIONS “-dirname C:\Program Files\Universal\ubroker”
```

If **`somescript.bat`** uses the last value specified to resolve duplicate command line arguments, then **`optsexample`** executes with **`dirname`** set to **`C:\Program Files\Universal\ubroker`**.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-handler_opts *options*			√	√	√
Environment Variable	n/a					
Configuration File Keyword	n/a					

Value

options is a character string that is appended to the command line constructed by Universal Event Monitor in order to execute the event handler process.

If *options* contains spaces, it must be enclosed in (**"**) quotation marks.

UNIX and z/OS

If *options* contains spaces, it must be enclosed in either single (**'**) or double (**"**) quotation marks.

If quotation marks are to be passed as part of the value (for example, an argument that contains spaces is to be passed to the handler process and treated as a single argument):

- Enclose *options* in apostrophes.
- Use quotation marks to enclose the quoted value.

For example, specifying `-handler_options 'parm1 “parm2a parm2b” parm3'`causes three arguments to be passed to the process executed on behalf of the event handler. The portion of the string enclosed in quotation marks (`“parm2a parm2b”`) is treated as a single value.

6.19 HANDLER_TYPE

Description

The HANDLER_TYPE option specifies the type of process that the UEM Server will execute for an event handler.

A UEM Server can execute either a:

- Command specified by the USER_COMMAND option.
- Script specified by the USER_SCRIPT option.

If this option is not used, UEM Server sets the handler type based on which execution option the UEM Manager specifies (that is, USER_COMMAND or USER_SCRIPT).

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-handler_type *type*			√	√	√
Environment Variable	n/a					
Configuration File Keyword	n/a					

Values

type is the type of process to be executed.

Valid values for *type* are:

- **cmd**
  Instructs the UEM Server to execute the command specified with the USER_COMMAND option.
- **script**
  Instructs the UEM Server to write the statements specified in the data set or DD name referenced by the USER_SCRIPT option to a temporary file on the UEM Server system, and then execute that file as a script.

[Default is cmd if a USER_COMMAND is provided.]

[Default is script if a USER_SCRIPT is provided.]

6.20 HELP

Description

The HELP option displays a description of the command options and their format.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	-h, -?			√	√	√
Command Line, Long Form	-help			√	√	√
Environment Variable	n/a					
Configuration File Keyword	n/a					

Value

(There are no values for the HELP option.)

6.21 HOST_SELECTION

Description

The HOST_SELECTION option specifies which host in the REMOTE_HOST list the UEM Manager will choose to begin its attempts to connect to a remote Universal Broker.

Regardless of how the first host is selected, UEM Manager processes the list sequentially until either a connection to a remote Universal Broker succeeds or all hosts in the list have been tried.

HOST_SELECTION is ignored if only one host is specified in the REMOTE_HOST list of hosts.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-host_selection *option*			√	√	√
Environment Variable	UEMHOSTSELECTION=*option*			√	√	
Configuration File Keyword	host_selection *option*			√	√	√

Values

option specifies how UEM Manager will select the first entry within a list of hosts specified by the REMOTE_HOST option.

Valid values for *option* are:

- **sequential**
  UCMD Manager will select the first host in the list, and then proceed through the hosts in the order in which they appear within the list.
- **random**
  UEM Manager will select any host in the list, in no particular order.
  Note:Attempts to connect to a remote Universal Broker begin with this randomly-selected host, and then proceed in a sequential manner, wrapping around the list (if necessary) until the first host selected is reached again.

[Default is sequential.]

6.22 HOSTNAME_RETRY_COUNT

Description

The HOSTNAME_RETRY_COUNT option specifies the number of times that the UEM Manager will attempt to establish a connection with the specified Universal Broker before it fails.

The UEM Manager will sleep for one second between connection attempts.

Connection errors occur for several reasons. A common reason is a failure to resolve the Universal Broker host name specified with the REMOTE_HOST option. This error can occur intermittently due to a temporary resource shortage or a temporary DNS problem. If your system is prone to host name resolution errors, it may help to have the UEM Manager retry the connection several times.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	N/A					
Command Line, Long Form	-hostname_retry_count *count*			√	√	√
Environment Variable	UEMHOSTNAMERETRYCOUNT= *count*			√	√	
Configuration File Keyword	hostname_retry_count *count*			√	√	√

Value

count is the number of times that UEM will attempt to establish a connection.

[Default is 1.]

6.23 INACTIVE_DATE_TIME

Description

The INACTIVE_DATE_TIME option specifies the date and time at which the UEM Server will stop testing for new occurrences of the specified system event. This is also the date and time by which any event occurrences detected by the UEM Server must complete in order for those occurrences to move into a **`triggered`** state.

If a detected event occurrence does not complete by this date and time, UEM will set that occurrence to a **`rejected`** state. If the specified date and time elapses, and no occurrence of the specified system event was detected by UEM, the event will be set to an **`expired`** state.

When the inactive date and time elapses, the UEM Server will wait for any executing handler processes to finish, and then end. When the UEM Server ends, the UEM Manager also will end at this time if it was instructed to wait for the Server by the WAIT option.

If INACTIVE_DATE_TIME is not used, default values are set by the UEM Manager:

- Default inactive date is the current date.
- Default inactive time is **`23:59`**.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-inact_date_time *yyyy.mm.dd,hh:mm* | *+mm*			√	√	√
Environment Variable	n/a					
Configuration File Keyword	n/a					

Value

An *absolute* or *relative* date and time can be specified in this option.

- *absolute* date and time is specified in the format **`yyyy.mm.dd,hh:mm`**, where:
  - **`yyyy.mm.dd`** is the date.
  - **`hh:mm`** is the time in 24-hour format.

    Either the date or the time can be omitted, in which case a default value is used.
  - If the date is omitted, the comma separator must be provided as a placeholder (that is, **`,hh:mm`**).
  - If the time is omitted, the comma is not required.

  Note:The maximum valid value for *absolute* date and time is **`2038.01.16,23:59`**.
- *relative* date and time is specified using the format **`+mm,`** where:
  - **`mm`** is the number of minutes that the event should remain active, starting from the time that the monitoring request was submitted.

6.24 INSTALLATION_DIRECTORY

Description

The INSTALLATION_DIRECTORY option specifies the location in which UEM Manager is installed.

Note: This option is required and cannot be overridden.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	n/a					
Environment Variable	n/a					
Configuration File Keyword	installation_directory *directory*			√	√	

Values

directory is the location in which UEM Manager is installed.

The full path name is required.

Defaults

UNIX

[Default is `/opt/universal/uemmgr`.]

Windows

[Default is `c:\Program Files\Universal\uemmgr`.]

6.25 LOGIN

Description

The LOGIN option instructs the UEM Server to establish an execution environment for the user account (specified by the USER_ID option) that resembles the environment that the user would have if the user were actually to log in to the system on which the UEM Server resides.

The differences between a login environment and a non-login environment depends on the UEM Server platform. For example, when a login environment is established on a UNIX system, the user's preferred shell is invoked as a login shell. This results in the execution of commands specified in the system profile and the user's profile.

For more information on the environment established for a given platform, see Chapter 7 Universal Event Monitor Server.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	-G *option*			√	√	√
Command Line, Long Form	-login *option*			√	√	√
Environment Variable	UEMLOGIN=*option*			√	√	
Configuration File Keyword	login *option*			√	√	

Values

option is the instruction for establishing an execution environment.

Valid *option* values are:

- **yes**
  Login environment is established.
- **no**
  Non-login environment is established.

If LOGIN is not used, a default value of **no** (set by the UEM Server configuration) is used.

6.26 MAX_OCCURRENCE_COUNT

Description

The MAX_OCCURRENCE_COUNT option specifies the maximum number of event occurrences that should be monitored for the current event. Any event occurrences detected by UEM that exceed this number are ignored.

This option is used to force an event that is monitored by a demand-driven UEM Server to become inactive when all occurrences detected for that event are set to a **`triggered`** state. This makes it possible for an event to become inactive without it having to wait for the date and time, as specified by the INACTIVE_DATE_TIME option, to elapse.

If the inactive date and time elapses before UEM detects the specified number of event occurrences, the inactive date and time takes precedence, and the event still will be made inactive. Any tracked occurrences that have not completed by this time will be set to a **`rejected`** state. If no event occurrences were detected, the event will be set to an **`expired`** state.

When the event monitored by the demand-driven UEM Server becomes inactive, the UEM Server process will finish when all processes executing on behalf of the specified event handler have finished. The UEM Manager then will wait for the UEM Server to finish (provided it was instructed to do so by the WAIT option).

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-max_count *count*			√	√	√
Environment Variable	UEMMAXCNT=*count*			√	√	
Configuration File Keyword	max_count *count*			√	√	√

Value

count is the number of event occurrences to be monitored.

If *count* = **`0`**, the demand-driven server will remain active until the INACTIVE_DATE_TIME option value has been met.

[Default is 1.]

6.27 MAXIMUM_RETURN_CODE

Description

The MAXIMUM_RETURN_CODE option specifies the highest return code value that a process executed on behalf of an event handler can return and still be considered as having executed successfully.

The situation in which any event handler process does not end successfully will be reflected in the return code of a waiting UEM Manager (see the WAIT option).

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-maxrc *returncode*			√	√	√
Environment Variable	n/a					
Configuration File Keyword	n/a					

Value

returncode is the highest return code value that can be returned for a process to be considered successful.

Default

If this option is not used, a default value of 0 (zero), as set by the UEM Server configuration, is used.

6.28 MESSAGE_LANGUAGE

Description

The MESSAGE_LANGUAGE option specifies the message catalog used to format messages issued by the UEM Manager.

z/OS

Message catalogs are stored as members in the library referenced by the UNVNLS DD statement in the UEMPRC catalogued procedure. The names of the message catalog members start with UEMMC.

UNIX and Windows

Message catalog files are located in the **`nls`** subdirectory of the main Stonebranch Solutions installation directory. All message catalog files end with an extension of **`.umc`**.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	-L *language*			√	√	√
Command Line, Long Form	-lang *language*			√	√	√
Environment Variable	UEMLANG=*language*			√	√	
Configuration File Keyword	language *language*			√	√	√

Values

language is the message catalog to be used.

The first three characters of **`language`** must match the last three characters of an installed message catalog (minus its extension).

[Default is `ENGLISH`, which instructs the UEM Manager to use the UEMMCENG message catalog.]

6.29 MESSAGE_LEVEL

Description

The MESSAGE_LEVEL option controls which messages are issued by a UEM Manager.

Optionally, it also allows a date and time stamp to be included with each message.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	-l *level [,time]*			√	√	√
Command Line, Long Form	-level *level [,time]*			√	√	√
Environment Variable	UEMLEVEL=*level [,time]*			√	√	
Configuration File Keyword	message_level *level [,time]*			√	√	√

Value

level is the type of messages issued.

Valid values for *level* are:

- **trace**
  Issues trace messages to the data set referenced by the UNVTRACE DD statement (for z/OS) or to a file that resides in the UEM Manager working directory named `uem.trc` (for Windows and UNIX).
  Note:**trace** is used for debugging purposes only. It should be used only when requested by Stonebranch Inc. Customer Support.
- **audit**
  Issues audit, informational, warning, and error messages.
- **info**
  Issues informational, warning and error messages.
- **warn**
  Issues warning and error messages.
- **error**
  Issues error messages only.

UNIX and Windows

[Default is warn.]

z/OS

[Default is info.]

time specifies whether or not a date and time stamp is to be included with each message.

Valid values for *time* are:

- **time**
  Include a date and time stamp with each message.
- **notime**
  Do not include a date and time stamp with each message.

UNIX and Windows

[Default is notime.]

z/OS

[Default is time.]

6.30 MINIMUM_FILE_SIZE

Description

The MINIMUM_FILE_SIZE option specifies the smallest allowable size for a file in order for it to be considered complete by UEM Server.

Note: This option is valid only for files being monitored via EVENT_TYPE (event type = **`FILE`**).

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-min_file_size *size[units]*			√	√	√
Environment Variable	n/a					
Configuration File Keyword	n/a					

Value

size is the file size. *units* is the unit of storage for that *size*.

The valid values for *units*, and the maximum value that can be specified for *size* for that unit of storage, is:

- **b** bytes (4,294,967,295)
- **k** kilobytes (4,194,304)
- **m** megabytes (4,096)
- **g** gigabytes (4)

If a value for *units* is not specified, the file size is assumed to be in bytes.

Default

If this option is not used, a default value of 0 (zero), as set by the UEM Server configuration, is used.

6.31 NETWORK_DELAY

Description

The NETWORK_DELAY option specifies the maximum acceptable delay in transmitting data between the UEM Manager and UEM Server over the network. If data transmission takes longer than the specified delay, the operation ends with a time-out error.

In this way, NETWORK_DELAY provides the ability to fine-tune Universal Event Monitor's network protocol.

When a data packet is sent over a TCP/IP network, the time it takes to reach the other end depends on many factors, such as network congestion and bandwidth. If the packet is lost before reaching the other end, the other end may wait indefinitely for the expected data. In order to prevent this situation, NETWORK_DELAY can be used to tell UEM how long to wait before setting a network time-out condition.

Note: An understanding of the TCP/IP protocol and the network configuration between the UEM Manager and UEM Server is required to determine the appropriate delay value.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	-d *seconds*			√	√	√
Command Line, Long Form	-delay *seconds*			√	√	√
Environment Variable	UEMNETWORKDELAY=*seconds*			√	√	
Configuration File Keyword	network_delay *seconds*			√	√	√

Value

seconds is the number of seconds to wait before setting a network time-out condition.

[Default is 120.]

6.32 NLS_ DIRECTORY

Description

The NLS_DIRECTORY option specifies the name of the directory where the UEM Manager message catalog and code page tables are located.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	n/a					
Environment Variable	n/a					
Configuration File Keyword	nls_directory *directory*			√	√	

Values

directory is the name of the directory where the message catalog and code page tables are located.

Full path names are recommended.

Relative path names are relative to the `universal` installation directory.

Defaults

UNIX

[Default is `/opt/universal/nls`.]

Windows

[Default is `..\nls`.]

6.33 OPTIONS

The OPTIONS option specifies a value that UEM Server uses for command line arguments to an event handler process it executes for a given EVENT_STATE.

Any value specified for OPTIONS works in conjunction with —- not as a replacement of — any value specified by the HANDLER_OPTIONS option. UEM Server uses the values of both options to construct a list of one or more command line arguments. Any value specified by OPTIONS always precedes the value specified by HANDLER_OPTIONS.

OPTIONS allows an event definition to control handler process behavior for a specific EVENT_STATE.

For example, given the following UEM Manager command parameters:

```
-event_type file -filespec c:\uemfiles\somefile.txt -inact_date_time +1
-handler_opts ">c:\uemlogs\somelog.txt 2>&1"
-triggered -script c:\uemscripts\somescript.bat
-expired -script c:\uemscripts\somescript.bat -options " -state expired"
```

If one or more occurrences of the event enter a triggered state, UEM Server (on Windows, in this example) executes **`c:\uemscripts\somescript.bat`**. If the event expires, UEM Server also executes **`c:\uemscripts\somescript.bat`**, but sets the script's **`-state`** command line parameter to "expired". UEM Server always redirects the script's stdout and stderr to **`c:\uemlogs\somelog.txt`**.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	-o *string*			√	√	√
Command Line, Long Form	-options *string*			√	√	√
Environment Variable	n/a					
Configuration File Keyword	n/a					

Values

string is a character string that UEM Server adds to the command line it constructs to execute an event handler process.

If *string* contains spaces, enclose it in double (**"**) quotation marks.

UNIX and z/OS

If *string* contains spaces, enclose it in single (**'**) or double (**"**) quotation marks.

To pass quotation marks as part of the value (for example, it is necessary to treat an argument with spaces as a single command argument):

- Enclose options in apostrophes.
- Use quotation marks to enclose the quoted value.

For example, specifying **`-options 'parm1 "parm2a parm2b" parm3'`** instructs UEM Server to pass three arguments to the process it executes on behalf of the event handler. UEM Server treats the portion of the string enclosed in quotation marks (**`"parm2a parm2b"`**) as a single value.

6.34 OUTBOUND_IP

Description

The OUTBOUND_IP option specifies the host or IP address over which the UEM Manager will establish all outbound network connections.

If this option is not used, the UEM Manager will establish its outbound connections on any interface that is available.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-outboundip *host*			√	√	√
Environment Variable	UEMOUTBOUNDIP=*host*			√	√	
Configuration File Keyword	outbound_ip *host*			√	√	√

Value

host is either:

- Host name (for example, **`myhost`**)
- IP address, in dotted decimal notation (for example, **`1.2.3.4`**)

[Default is `*`.]

6.35 PLF_DIRECTORY

Description

The PLF_DIRECTORY option specifies the Program Lock File (PLF) directory where the program lock files are located.

A program lock file is created and used by the UEM Manager process to store manager process termination information for the Universal Broker.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-plf_directory *directory*			√		
Environment Variable	UEMPLFDIRECTORY=*directory*			√		
Configuration File Keyword	n/a					

Values

directory is the name of the PLF directory.

[Default is `/var/opt/universal/tmp`.]

6.36 POLLING_INTERVAL

Description

The POLLING_INTERVAL option specifies the frequency with which the UEM Server will check to see if an event's inactive date and time has elapsed; if so, the monitored state of the event is set to `inactive`.

If the inactive date and time has not elapsed, and the event is to remain active, UEM also checks for new occurrences of the specified event whenever the polling interval expires.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-polling_int *seconds*			√	√	√
Environment Variable	UEMPOLLINGINT=*seconds*				√	
Configuration File Keyword	polling_int *seconds*			√	√	√

Value

seconds is the frequency (in seconds) for checking an event’s inactive date and time.

Default

If this option is not used, the UEM Server configuration sets a default value of 30 seconds.

6.37 REMOTE_HOST

Description

The REMOTE_HOST option specifies a comma-delimited list of one or more hosts upon which a command can run.

Each host in the list can be specified as a host name or an IP address in dotted decimal notation.

Note: For purposes of this discussion, even if one host is specified, it is considered a list with a single entry.

If multiple hosts are specified, the UEM Manager will try each one until it successfully connects to a Universal Broker. When a successful connection is established, no more hosts in the list are tried. If the UEM Manager is unable to establish a connection to a Universal Broker on any of the hosts in the list, the UEM Manager will fail.

Each host in the REMOTE_HOST list also can include a port number or service name that the UEM Manager will use to establish a connection with the Universal Broker on that host. This is useful in situations where the remote Broker is configured to accept incoming connections on a port that is different from the UEM Manager's configured value.

UEM Manager automatically removes any duplicates found in the list of hosts before it makes its first connection attempt. This includes any host names that resolve to the same IP address, or duplicate IP addresses that are added to the list following DNS expansion (see the DNS_EXPAND option).

Note: Duplicate IP addresses may appear in the resolved, expanded, scrubbed list of hosts if a different port number is specified for each occurrence of a particular IP address (for example, 192.168.1.1:7887 and 192.168.1.1:7888 are considered distinct entries).

To set configuration options that control selection of the hosts in the REMOTE_HOST list, see the HOST_SELECTION, CONNECT_TIMEOUT, and DNS_EXPAND options.

The following text briefly describes each of these options. Each is briefly described below.

HOST_SELECTION

UEM Manager uses the HOST_SELECTION option to control which host in the REMOTE_HOST list is selected first. UEM Manager can be configured to always select the first host or to select with a randomly chosen host as it begins its attempts to connect to a remote Universal Broker. In either case, when a host is selected, UEM Manager processes the list sequentially until either a connection succeeds or all hosts in the list are tried.

(See the HOST_SELECTION option for more information.)

CONNECT_TIMEOUT

UEM Manager uses the CONNECT_TIMEOUT option to specify how long it will wait for a connection attempt to succeed before it moves on to the next host in the list.

(See the CONNECT_TIMEOUT option for more information.)

DNS_EXPAND

The DNS_EXPAND option controls the number of IP addresses returned when UEM Manager issues a DNS query to resolve a host name. If the UEM Manager is configured to expand the results of the query, all IP addresses defined for a particular host name are returned and expanded (in-place) within the list of hosts. Otherwise, only the first host is returned, and no expansion is performed.

(See the DNS_EXPAND option for more information.)

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	-i *hostlist*			√	√	√
Command Line, Long Form	-host *hostlist*			√	√	√
Environment Variable	UCMDHOST hostlist			√	√	
Configuration File Keyword	host *hostlist*			√	√	√

Values

hostlist is a list of one or more hosts, in the following format:

host1[[:port1],host2[:port2],host3[:port3],…hostn[:portn]]

In this format:

- *host* is the IP address (in dotted decimal notation) or host name of the system upon which the command may run.
- *port* is an optional port number (or service name), which is necessary only if the remote Universal Broker is accepting incoming connections on a port that is different from the value specified by the REMOTE_PORT option.

Note: The *port* number, when specified, must be separated from the *host* by a colon (**:**).

6.38 REMOTE_PORT

Description

The REMOTE_PORT option specifies the TCP port upon which a Universal Broker, on the system specified by the REMOTE_HOST option, is accepting connections.

The UEM Manager connects to this Universal Broker in order to request the start of a demand-driven UEM Server, which is responsible for monitoring the defined event.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	-p *port*			√	√	√
Command Line, Long Form	-port *port*			√	√	√
Environment Variable	UEMPORT=*port*			√	√	
Configuration File Keyword	port *port*			√	√	√

Value

port is the port upon which the Universal Broker is accepting connections.

Valid values for *port* are:

- Number (for example, **`7887`**)
- Service name (for example, **`ubroker`**)

[Default is `7887`.]

6.39 RENAME_FILE

Description

The RENAME_FILE option specifies whether or not a file that is being monitored - as specified via an EVENT_TYPE of **`FILE`** - should be renamed by UEM when the event occurrence is set to a **`triggered`** state.

Renaming a file ensures that UEM will not treat a file that it just finished tracking as a new event occurrence.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-rename_file *option*			√	√	√
Environment Variable	n/a					
Configuration File Keyword	n/a					

Value

option is the specification for whether or not a file should be renamed.

Valid values for *option* are:

- **yes**
  File is renamed according to the RENAME_FILE_SPECIFICATION option.
- **no**
  File is not renamed.

If *option* is **no**, in order to prevent multiple detection of the file by the UEM Server, either:

- Set the value of the MAX_OCCURRENCE_COUNT option to **1**.
- Rename the file in a script invoked by an event handler when occurrence is triggered.

Default

If RENAME_FILE is not used, a default value of **yes**, as set by the UEM Server configuration, is used.

6.40 RENAME_FILE_SPECIFICATION

Description

The RENAME_FILE_SPECIFICATION option specifies the file name that the UEM Server should use when both:

- An event occurrence for an event with an EVENT_TYPE of **FILE** is set to a **triggered** state.
- RENAME_FILE option is set to **yes**.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-rename_filespec *renamespecification*			√	√	√
Environment Variable	n/a					
Configuration File Keyword	n/a					

Value

renamespecification is the file name to be used by the UEM Server.

Valid values for *renamespecification* can include an absolute path or a path that is relative to the location of the file monitored by UEM. If no path information is provided, the monitored file simply is renamed. The renamed file will reside in the same location as the original file.

If *renamespecification* contains spaces, it must be enclosed in double (**"**) quotation marks.

UNIX and z/OS

If *renamespecification* contains spaces, it must be enclosed in either single (**'**) or double (**"**) quotation marks.

Variables

Variables that UEM will substitute with actual runtime values can be included as part of *renamespecification*. These variables, and the values that UEM will substitute for them, are:

- **`$(compname)`**
  Component name
- **`$(compid)`**
  Component ID
- **`$(date)`**
  Current date, in the format *YYYYMMDD*
- **`$(time)`**
  Current time, in the format *HHMMSS*
- **`$(origname)`**
  Original base file name, minus its last extension
- **`$(origext)`**
  Original file extension
- **`$(seqnum)`**
  Sequence number that is initialized to 0 when the demand-driven UEM Server is started, and is then incremented by 1 for each file renamed.

Default

If RENAME_FILE_SPECIFICATION is not used, the following default value, set by the UEM Server configuration, is used:

```
$(compname).$(compid).$(date).$(seqnum)
```

6.41 SYSTEM_ID

Description

The SYSTEM_ID option identifies the local Universal Broker with which the UEM Manager must register before the Manager performs any request.

Each Universal Broker running on a system is configured with a system identifier that uniquely identifies the Broker.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-system_id *ID*					√
Environment Variable	UEMSYSTEMID=*ID*					√
Configuration File Keyword	n/a					

Values

ID is the system identifier of the local Universal Broker.

(Refer to the local Universal Broker administrator for the appropriate system ID to use.)

6.42 TRACE_FILE_LINES

Description

The TRACE_FILE_LINES option specifies the maximum number of lines to write to a trace file.

A trace file is generated when the MESSAGE_LEVEL option is set to **TRACE**. The trace file will wrap around when the number of lines specified by TRACE_FILE_LINES has been reached. New trace entries are written at the top of the trace file, just after the trace header lines.

z/OS

The trace file is written to the data set referenced by the UNVTRACE DD statement.

Note: This option has no effect if the UNVTRACE DD statement points to a JES SYSOUT file.

Windows and UNIX

Trace entries are written to the **uem.trc** file, which resides in the UEM Manager working directory.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-tracefilelines *lines*			√	√	√
Environment Variable	UEMTRACEFILELINES=*lines*			√	√	
Configuration File Keyword	trace_file_lines *lines*			√	√	√

Value

lines is the maximum number of lines to write to a trace file.

When setting this value, allow for an average trace file line size of 50 characters.

[Default is 50,000.]

6.43 TRACE_TABLE

Description

The TRACE_TABLE option specifies the size of a wrap-around trace table maintained in memory.

Tracing is activated when the MESSAGE_LEVEL option is set to **TRACE**.

z/OS

The trace table is written to a data set when the program ends under the conditions specified by TRACE_TABLE.

Windows and UNIX

The trace table is written to a file named **uem.trc** when the program ends under the conditions specified by TRACE_TABLE.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-trace_table *size[units],cond*			√	√	√
Environment Variable	UEMTRACETABLE=*size[units][,cond]*			√	√	
Configuration File Keyword	trace_table *size[units],cond*			√	√	√

Values

size is the size of the trace table.

If *size* is set to *0* (zero), the trace table is not used.

[Default is *0*.]

units is the unit of storage for that *size*.

The valid values for *units*, and the maximum value that can be specified for *size* for that unit of storage, is:

b	bytes (2,147,483,647)
k	kilobytes (2,097,152)
m	megabytes (2,048)
g	gigabytes (2)

If a value for *units* is not specified, the file size is assumed to be in bytes.

cond specifies the condition under which the trace table is written.

Valid values for *cond* are:

- **error**
  Writes the trace table if the program ends with a non-zero return code.
- **always**
  Writes the trace table when the program ends regardless of the return code.
- **never**
  Never write the trace table.

[Default is never.]

6.44 TRACKING_INTERVAL

Description

The TRACKING_INTERVAL option specifies the frequency with which the UEM Server will test for the completion of a tracked event occurrence.

The test for completeness depends upon the type of event that the tracked occurrence represents. For example, an occurrence tracked for an event with an EVENT_TYPE of **`FILE`** is considered complete when two consecutive checks on a file's size return the same value.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-tracking_int *seconds*			√	√	√
Environment Variable	n/a					
Configuration File Keyword	n/a					

Value

seconds is the frequency (in seconds) with which the UEM Server tests for completion of a tracked occurrence.

If TRACKING_INTERVAL is not used, a default value of 10 seconds, set by the UEM Server configuration, is used.

6.45 USER_COMMAND

Description

The USER_COMMAND option specifies an application or script that the demand-driven UEM Server should execute whenever an event occurrence, or the event itself, enters one of the states specified by the EVENT_STATE option.

USER_COMMAND cannot be used with the HANDLER_ID or USER_SCRIPT options for a specified EVENT_STATE.

Examples

The following example is invalid:

```
-triggered -cmd dir -handler_id id001
```

In this example, it is unclear whether the UEM Server should execute the **`dir`** command or the actions contained in the stored event handler **`id001`** when an event occurrence is set to a **`triggered`** state.

The following example is valid:

```
-triggered -handler_id id001 -expired -cmd dir
```

In this example, the **`-handler_id`** and **`-cmd`** options are specified for different event states.

- If an event occurrence is set to a **`triggered`** state, the UEM Server will execute the actions specified in the stored event handler **`id001`**.
- If the monitored event is set to an **`expired`** state, the UEM Server will execute the **`dir`** command.

Stoneman's Tip

If a Windows UEM Manager has issued a request to monitor an event on a UNIX system, and the value specified by command contains a complete path, a leading space must be added to the path name, and the path must be enclosed in quotes.

For example, if the UEM Server is to execute an application named **`someapp`** that resides in **`/opt`**, the value specified for command must be **`" /opt/someapp"`**.

This requirement is a result of the way that command line arguments are read by a Windows UEM Manager, where a forward slash (**`/`**) also can be used as a command option prefix. The leading space allows the string beginning with the **`/`** to be interpreted as a option value.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-cmd *command*			√	√	√
Environment Variable	n/a					
Configuration File Keyword	n/a					

Value

command is the application or script to be executed.

If the application or script is not in the UEM Server home directory, or if it resides outside of the system path, the complete path to the file must be provided.

If *command* contains spaces, it must be enclosed in double (**"**) quotation marks.

UNIX and z/OS

If *command* contains spaces, it must be enclosed in either single (**'**) or double (**"**) quotation marks.

6.46 USER_ID

Description

The USER_ID option specifies the user account in whose security context the demand-driven UEM Server will perform all event monitoring and execute all event handler processes.

A user ID is required only if security is enabled in the UEM Server's configuration.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	-u *user*			√	√	√
Command Line, Long Form	-userid *user*			√	√	√
Environment Variable	UEMUSERID=*user*			√	√	
Configuration File Keyword	userid *user*			√	√	√

Value

user is the user account.

Valid values for *user* are valid user accounts that are known to the target system (that is, the system identified by the REMOTE_HOST).

6.47 USER_PASSWORD

Description

The USER_PASSWORD option specifies a password for the user account specified by the USER_ID option.

A password is required only if the UEM Server has enabled its USER_SECURITY option.

Note: If the UEM Server is executing on a UNIX system, an entry can be added to the Universal Event Monitor Access ACL to not require authentication of certain user accounts. For those user accounts, a user password is not required. However, if the UEM Server is executing on a Windows system, this option is not available, as all user account authentication done by Windows requires a password.

For more information on Universal ACLs, see Chapter 12 Universal Event Monitor UACL Entries.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	-w password			√	√	√
Command Line, Long Form	-pwd *password*			√	√	√
Environment Variable	UEMPWD=*password*			√	√	
Configuration File Keyword	password *password*			√	√	√

Value

password is the password for the user account.

6.48 USER_SCRIPT

Description

The USER_SCRIPT option specifies the location of one or more system commands that are executed as a script file by the UEM Server whenever an event occurrence, or the event itself, enters one of the states specified by the EVENT_STATE option.

The system commands (that is, script statements) are read from:

- DD statement referenced by *ddname* (z/OS)
- Local file specified by *filename* (Windows and UNIX)

The statements are sent to the demand-driven UEM Server and stored in a temporary script file. It is this script file that is executed by UEM on behalf of the event handler.

To execute a script that resides on the remote system
where an event is being monitored by a demand-driven UEM Server,
use the USER_COMMAND option
and give the script file as the value for command.

Stoneman's Tip

The script's contents can include any statement that is valid for the system on which they will execute (that is, the system identified by the REMOTE_HOST option). The maximum size of a script file is approximately 64,000 characters.

USER_SCRIPT cannot be used with the HANDLER_ID or USER_COMMAND options for a specified EVENT_STATE.

Examples

z/OS

The following example is invalid:

```
-triggered -script myscript -handler_id id001
```

In this example, it is unclear whether the contents of the script contained in the data set referenced by the DD name **myscript**, or the actions contained in the stored event handler **id001**, should be executed whenever an event occurrence is set to a **triggered** state.

The following example is valid:

```
-triggered -handler_id id001 -expired -script myscript
```

In this example, the **-handler_id** and **-script** options are specified for different event states.

- If an event occurrence is set to a **triggered** state, the UEM Server will execute the actions specified in the stored event handler **id001**.
- If the monitored event is set to an **expired** state, the UEM Server will execute the contents of the script referenced by the DD statement **myscript**.

UNIX and Windows

The following example is invalid:

```
-triggered -script c:\myscript.txt -handler_id id001
```

In this example, it is unclear whether the contents of the script contained in the local file **c:\myscript.txt** or the actions contained in the stored event handler **id001** should be executed whenever an event occurrence is set to a triggered state.

The following example is valid:

```
-triggered -handler_id id001 -expired -script c:\myscript.txt
```

In this example, the **-handler_id** and **-script** options are specified for different event states.

- If an event occurrence is set to a **triggered** state, the demand-driven UEM Server will execute the actions specified in the stored event handler **id001**.
- If the monitored event is set to an **expired** state, the UEM Server will execute the contents of the script contained in the file **c:\myscript.txt**.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-script *ddname* or *filename*			√	√	√
Environment Variable	n/a					
Configuration File Keyword	n/a					

Value

ddname (for z/OS) and *filename* (for Windows and UNIX) is the location of the commands to be executed as a script file.

6.49 USER_SCRIPT_TYPE

Description

The USER_SCRIPT_TYPE option describes the type of statements contained within the script specified by the USER_SCRIPT option.

Note: Currently, USER_SCRIPT_TYPE applies only to scripts executed by UEM Servers running on a Windows system.

When the UEM Server writes the script statements to a temporary script file, the value specified by USER_SCRIPT_TYPE is used as that file's extension. On the target Windows system, an association between files with that file extension and an application responsible for opening files with that extension must exist prior to any attempt by a UEM Server to execute the script. Otherwise, execution of the script will fail.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-script_type *type*			√	√	√
Environment Variable	n/a					
Configuration File Keyword	n/a					

Value

type is the type of statement in the script specified by USER_SCRIPT.

If this option is not used, a default value of **bat**, set by the UEM Server configuration, is used.

6.50 VERSION

Description

The VERSION option instructs UEM Manager to display program version and copyright information.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	-v			√	√	√
Command Line, Long Form	-version			√	√	√
Environment Variable	n/a					
Configuration File Keyword	n/a					

Value

(There are no values for this option.)

6.51 WAIT

Description

The WAIT option instructs UEM Manager to execute until it receives notification that the demand-driven UEM Server has ended.

A demand-driven Server ends when both of the following occur:

- Event that it is monitoring is set to an inactive state.
- Any handler processes executed by the Server have completed.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-wait *option*			√	√	√
Environment Variable	UEMWAIT=*option*			√	√	
Configuration File Keyword	wait *option*			√	√	√

Value

option is the instruction for whether or not to wait for the UEM Server to finish executing.

Valid values for *option* are:

- **yes**
  UEM Manager will wait for the demand-driven UEM Server to finish executing.
- **no**
  UEM Manager will end as soon as it receives notification from the demand-driven UEM Server that the event parameters are valid and monitoring of the event has begun. All additional event monitoring activity will be reported only in the Universal Broker log.

[Default is yes.]

CHAPTER 7

Universal Event Monitor Server

7.1 Overview

Universal Event Monitor (UEM) Server is a Universal Broker-managed component whose primary functions include:

- Monitoring of system events described by event definitions
- Responding to different event outcomes
- Executing processes on behalf of event handlers
- Validating database maintenance requests received from the UEMLoad utility

An event definition describes a system event whose occurrence should be detected and monitored by a UEM Server. Event definition parameters are supplied to UEM Server by way of command options used to execute a UEM Manager, or from records stored in the event definition database.

UEM Server handles (that is, responds to) an event based on the result of monitoring system activity. For each defined system event monitored by UEM Server, zero to many occurrences of that event actually can be detected and tracked by UEM.

- If a UEM Server detects an occurrence of an event and later determines that the event occurrence has completed, UEM will set that occurrence to a **`triggered`** state.
- If a UEM Server detects an occurrence of an event but the occurrence does not complete within the time frame specified, UEM will set that occurrence to a **`rejected`** state.
- If a UEM Server does not detect any occurrence of a defined system event within the allotted time, UEM will set the entire event to an **`expired`** state.

UEM Server may follow up each of these states' responses with the execution of a system command or script specified by what is known as an event handler. An event handler's parameters can come from the command options specified for a UEM Manager or from a stored record in the event handler database.

For each event monitored, up to three different event handlers can be specified, one for each of the states (**`triggered`**, **`rejected`**, **`expired`**) described above.

Finally, a UEM Server is used to validate database maintenance requests issued by the UEMLoad utility.

UEMLoad is a client application that serves as the user interface for maintaining records in the event definition and event handler databases. Record parameters are passed to UEMLoad via its available command options. These parameters then are forwarded to a local UEM Server, which validates the parameters and supplies configured default values for any required parameters that were omitted in the request. The UEM Server then forwards the request to the Universal Broker so it actually can be applied to the appropriate database.

7.2 Demand-Driven vs. Event-Driven

References are made throughout this manual to *demand-driven* and *event-driven* UEM Servers. Both types of UEM Servers execute from the same application file, **`uemsrv.exe`**, (**`uemsrv`** on UNIX), and both types monitor events in essentially the same way.

However, the factors that distinguish a demand-driven Server from an event-driven Server include the following:

- Method in which the UEM Server component is defined
- Method used to initiate the startup of the UEM Server process
- Ability to recover previously monitored events during startup
- Method used to provide the UEM Server with event definitions and event handlers
- Destination of output for event monitoring activity
- Ability to refresh an active UEM Server's configuration
- Duration of the UEM Server process

7.2.1 Component Definitions

The values of certain parameters in the component definitions for a demand-driven UEM Server and event-driven UEM Server is one way to distinguish between them.

One parameter in particular, **`component_type`**, always can be relied upon to identify a demand-driven and an event-driven UEM Server:

- Demand-driven Server has a component type of **`uemd`**.
- Event-driven Server's component type is **`uems`**.

A demand-driven UEM Server also relies upon a command line argument, **`-demand yes`**, to know whether it should run in demand-driven or event-driven mode. This argument is supplied to the UEM Server process via the **`start_command`** parameter of the component definition. This value is set at installation time, and should not be changed.

7.2.2 Server Startup

As a Stonebranch Solutions component, both types of UEM Server only can be started by a Universal Broker.

A demand-driven UEM Server is started by Universal Broker when the Universal Broker receives a request to do so from one of the UEM client applications (that is, the UEM Manager or UEMLoad). Since a user request drives the startup of the UEM Server process, the process is said to be executed on demand; hence the term demand-driven Server.

An event-driven UEM Server typically is started automatically during Universal Broker startup. A UEM client application is not required to start an event-driven Server.

The Universal Control utility can be used to initiate the startup of an event-driven UEM Server. While it may appear that Universal Control is starting the UEM Server, it merely is, in fact, providing an interface from which the start request can be sent to the Universal Broker. The Universal Broker still is solely responsible for starting the UEM Server.

For complete information on Universal Control, see the Stonebranch Solutions Utilities 4.2.0 Reference Guide.

7.2.3 Event Recovery

All event monitoring activity conducted by a demand-driven or event-driven UEM Server is recorded in the event spool database. An event spool record is created when an occurrence of an event is detected, or when an event expires. This record also is used to track the status of any event handlers that are executed for the event.

When a demand-driven UEM Server ends, no association is maintained between the Server component and the event spool records it created. If an event occurrence is being tracked when a demand-driven Server ends, the processing state of the record is set to **`REJECTED`** (or **`CANCELLED`** if the UEM Manager was terminated forcefully). If a handler is being executed for an event or event occurrence, the handler status is set to **`SHUTDOWN`**. Unless the spool records are being retained by the Universal Broker (see the Stoneman's Tip, below), they are deleted when the UEM Server process ends.

When an event-driven UEM Server ends, any event spool records with a processing state of **`TRACKING`** or **`EXECUTING`** are retained automatically. Event spool records with all other processing states are only retained under the condition described in the Stoneman's Tip, below.

When the event-driven UEM Server is re-started, it will retrieve (based on component name) all event spool records that it retained from its previous run. If it finds any event occurrences whose processing state is **`TRACKING`**, it will resume tracking that occurrence. If the record indicates that an event handler process was **`EXECUTING`** when the UEM Server was brought down, that record's handler status will be set to **`UNRECOVERABLE`**. This is necessary because the only way to retrieve the handler process's status is via a process ID that could very likely no longer exist in the system. Further, if the process ID does exist, there is no way of knowing whether it belongs to the same process recorded in the event spool record.

Stoneman's Tip

An option can be set in the Universal Broker's configuration to prevent it from deleting any event spool records when the UEM Server component completes. Setting the **`comp_info_retention`** option to a value greater than 0 causes the event spool record to be preserved.

Because there is currently no database cleanup routine available, this option should be set only following a recommendation from, and with the assistance of, Stonebranch Inc. Customer Support.

Currently, event spool records only can be viewed with the Universal Spool List utility, **`uslist`**.

Figure 7.1, below, illustrates the syntax used for viewing event spool records.

```
uslist -list uems [-id recid]
```

Figure 7.1 USList Command Syntax for Viewing Event Spool Records

An optional event spool record ID, specified by **recid**, can be provided to obtain complete detail for a singe record.

Figure 7.2, below, illustrates a sample summary and detailed output listing of event spool records.

```
> uslist -list uems
SERIAL NOEVENT IDPRC STATEHANDLER IDEXIT CODEEXIT STATUSHANDLER STATUS
1        uemsam1 Triggereduemsam_t10       Normal    Successful
2        uemsam2 Triggereduemsam_t20       Normal    Successful
3        uemsam3 Expired  uemsam_x3 0      Normal    Successful
4        uemsam2 Tracking N/A      0       Default   Default
5        uemsam1 Triggereduemsam_t10       Default   Executing

> uslist -list uems -id 2
Serial No...............: 2
Event ID................: uemsam2
Component Name..........: uems
Component Description...: Universal Event Monitor Server
Component Version.......: 4.2.0 Level 0 Release Build 109
Component ID............: 1116252422
Event Type..............: FILE
System Object...........: C:\UEMFiles\uemsam2.dat
Processing State........: Triggered
Handler ID..............: uemsam_t2
User Cmd................: N/A
Process ID..............: 2572
User ID.................: sparkie
Start Time..............: 5/16/2010 10:08:52 AM
End Time................: 5/16/2010 10:08:55 AM
Exit Code...............: 0
Exit Status.............: Normal
Handler Status..........: Successful
Last Modified On........: 5/16/2010 10:08:55 AM
```

Figure 7.2 Sample Event Spool Listings

For additional information on all utilities provided with Stonebranch Solutions, see the Stonebranch Solutions Utilities 4.2.0 Reference Guide.

7.2.4 Source of Event Parameters

A demand-driven UEM Server relies upon its associated UEM Manager for event definition and event handler information. All the information needed to define the event and describe event handlers come from the command options used to invoke the UEM Manager. While these command options may reference the ID of an event definition or event handler stored in their respective database, the existence of such a record is not required.

Conversely, an event-driven UEM Server relies completely upon these stored event definition and event handler records for its input. One of the first things an event-driven UEM Server component does upon startup is retrieve a list of event definitions that are assigned to it. It then determines which events are active, and begins monitoring them.

7.2.5 Reporting of Monitoring Activity

Demand-driven and event-driven UEM Servers record all significant monitoring activity in the Universal Broker log.

This activity includes:

- Detection of event occurrences
- State changes for events and event occurrences
- Results of event handler execution
- Recovery of outstanding event occurrences during process startup

This activity only is viewable on the system where the UEM Server executes. To allow monitoring activity to be followed from the system where the monitoring request was issued, a demand-driven UEM Server also provides feedback to a waiting UEM Manager.

Monitoring activity for demand-driven and event-driven UEM Servers is also captured in the event spool database.

See the Section 7.2.3 Event Recovery (and the Databases chapter of the Indesca or Infitran 4.2.0 User Guide) for more information on the event spool database.

7.2.6 Configuration Refresh

Because an event-driven UEM Server typically is a long-running process, the ability to refresh an active UEM Server's configuration and list of assigned event definitions is provided. Automatic refresh of configuration and event information for a demand-driven UEM Server is not supported; the values it obtains at startup are the ones it uses throughout its lifetime.

When a change is made to the stored UEM Server configuration settings, active event-driven UEM Servers must be notified that a change has taken place. This is done with Universal Control, using the REFRESH_CMD option, along with a component type value that identifies the component to refresh.

Figure 7.3, below, illustrates the command syntax that demonstrates how to use Universal Control to refresh an event-driven UEM Server's configuration.

For more information on Universal Control, see the Stonebranch Solutions Utilities 4.2.0 Reference Guide.

```
uctl -host myhost -refresh uems
```

Figure 7.3 Configuration Refresh using Universal Control

Windows

A request to update the configuration of local event-driven UEM Servers is issued automatically whenever a change is made to a UEM Server's configuration through the Universal Configuration Manager.

When Universal Control or the Universal Configuration Manager (Windows only) instructs an active event-driven UEM Server to refresh its cached configuration, the event-driven Server processes the request immediately.

The UEMLoad utility automatically notifies an event-driven UEM Server of an event definition change via a flag that resides in the local Universal Broker. UEM Server checks this flag every two minutes and updates its cached list of event definitions whenever UEMLoad updates them. This eliminates the need to refresh UEM Server with Universal Control following a database change.

7.2.7 Process Lifetime

A demand-driven Server is responsible only for monitoring a single event. When that event's inactive date/time has elapsed, or the maximum number of event occurrences has been detected, the event will be made inactive and the UEM Server process will end.

Conversely, an event-driven UEM Server executes until it receives a shutdown request from the Universal Broker that started it. This shutdown request typically is sent during Universal Broker shutdown, when it forces the termination of all active components.

Universal Control can be used to initiate the shutdown of an event-driven UEM Server. While it may appear that Universal Control is stopping the UEM Server component, it is, in fact, merely providing an interface from which the stop request may be sent to the Universal Broker. The UEM Server will stop only after it receives the shutdown request from the Universal Broker.

For complete information on Universal Control, see the Stonebranch Solutions Utilities 4.2.0 Reference Guide.

CHAPTER 8

Universal Event Monitor Server for Windows

8.1 Overview

This chapter describes the information for running the Universal Event Monitor Server under the Windows operating system.

This information is separated into the following sections:

- Server Environment
- Component Definition
- Configuration
- Universal Access Control List

8.2 Server Environment

UEM Server is a Universal Broker-managed component, which means it only can be started and (gracefully) shut down by the Universal Broker. It runs as a background process and, except for displaying version and copyright information, it provides no native user interface or console interaction.

Execution Context

The security context that a UEM Server executes with depends on UEM Server configuration options and whether the Server is demand-driven or event-driven (see Section 7.2 Demand-Driven vs. Event-Driven).

If the USER_SECURITY option is enabled in the UEM Server configuration, a demand-driven UEM Server executes in the security context of the user account specified via the UEM Manager. If security is not enabled in the UEM Server configuration, it runs in the security context of the Universal Broker.

An event-driven UEM Server always executes in the security context of the Universal Broker, regardless of the value of the security configuration option.

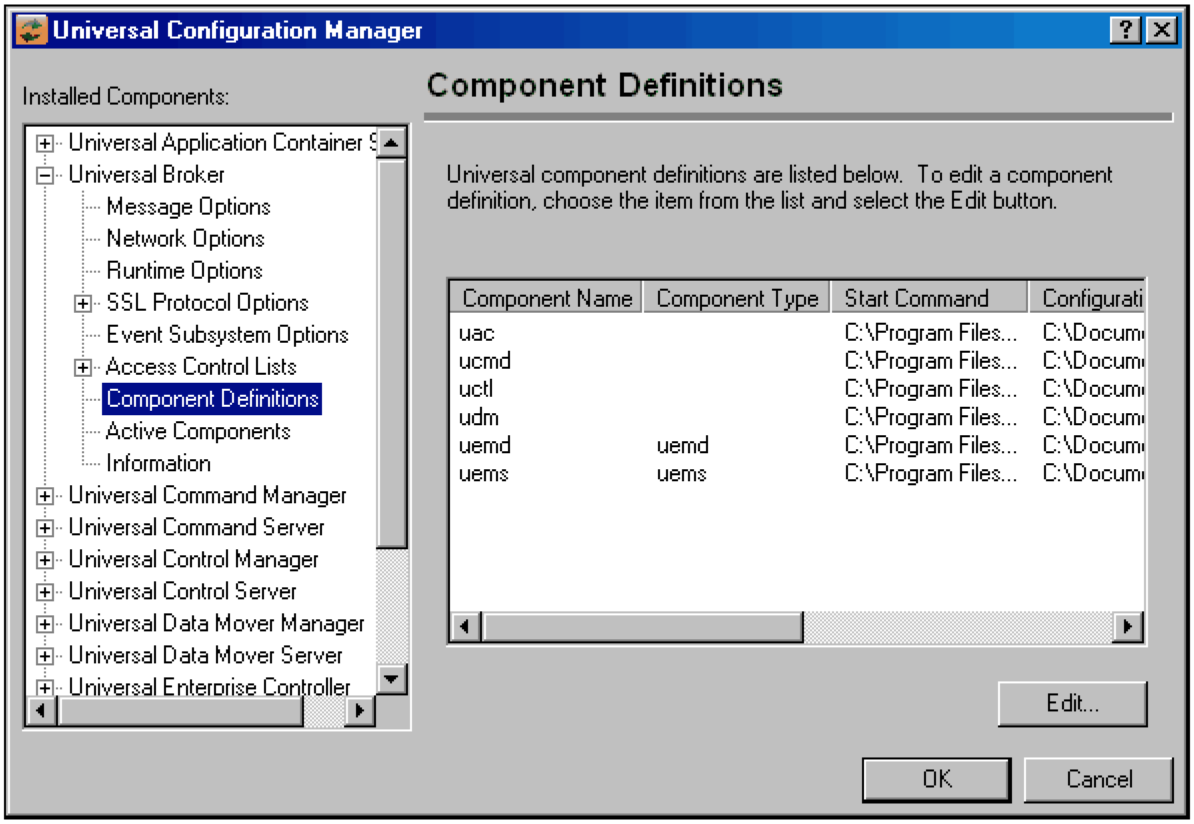
8.3 Component Definition

All Stonebranch Solutions components managed by Universal Broker have a component definition. The component definition is a text file of options containing component-specific information required by Universal Broker. (For details on how Universal Broker manages components, see the Universal Broker 4.2.0 Reference Guide.)

Note: The syntax of a component definition file is the same as a configuration file.

Although component definition files can be edited with any text editor (for example, Notepad), the Universal Configuration Manager application is the recommended way to edit component definitions for Windows.

Note: The component definitions for all Stonebranch Solutions components are identified in the Component Definitions property page of the Universal Broker (see Figure 8.1, below).



Universal component definitions are listed below. To edit a component definition, choose the item from the list and select the Edit button.

Component Name	Component Type	Start Command	Configurati
uac		C:\Program Files...	C:\Docum
ucmd		C:\Program Files...	C:\Docum
uctl		C:\Program Files...	C:\Docum
udm		C:\Program Files...	C:\Docum
uemd	uemd	C:\Program Files...	C:\Docum
uems	uems	C:\Program Files...	C:\Docum

Figure 8.1 Universal Configuration Manager - Component Definitions

Table 8.1 identifies all of the options that comprise the UEM Server for Windows component definition.

Each **Option Name** is a link to detailed information about that component definition option.

Option Name	Description
AUTOMATICALLY_START	Specification for whether or not UEM Server starts automatically when Universal Broker is started
COMPONENT_NAME	Name by which the clients know the UEM Server
COMPONENT_TYPE	Type of UEM Server (demand-driven or event-driven)
CONFIGURATION_FILE	Full path name of the UEM Server configuration file
RUNNING_MAXIMUM	Maximum number of UEM Servers that can run simultaneously
START_COMMAND	Full path name of the UEM Server program
WORKING_DIRECTORY	Full path name of the UEM Server working directory

Table 8.1 UEM Server for Windows - Component Definition Options

8.4 Configuration

All UEM Servers defined on a particular system share a single configuration. A demand-driven UEM Server reads configuration options when it starts, and uses those values while it executes. Because an event-driven UEM Server can run for an extended period of time, the configuration information it caches at start-up can be refreshed whenever the stored configuration options are refreshed (see Section 7.2.6 Configuration Refresh).

A UEM Server's configuration consists of default values that are used to establish the runtime environment and provide missing values for required event definition and event handler parameters.

The UEMLoad utility can override a configured default via its command options when it adds records to the event definition and event handler databases. Also, a UEM Manager can override options for a demand-driven UEM Server when it submits a monitoring request.

The syntax for those overrides, when available, are included with the description of those options.

8.4.1 Configuration Options

Table 8.2, below, categorize available UEM Server configuration options according to function.

Category	Description
Event Definition	Allow default values to be set for any required event definition parameters that were omitted from the UEM Manager or UEMLoad utility request.
Event Handler	Allow default values to be set for any required event handler parameters that were omitted from the UEM Manager or UEMLoad utility request.
Installation	Options that specify installation requirements, such as directory locations
Message	UEM message options.
Monitoring	Control UEM activity.
Network	Manage data transmission between local and remote systems.

Table 8.2 UEM Server for Windows - Configuration Options Categories

The UEM Server configuration options for each of the categories listed in Table 8.2 are summarized in the following tables. Each Option Name is a link to detailed information about that option.

Event Definition Category Options

Option Name	Description
ASSIGNED_COMPONENT_NAME	UEM Server component to which a stored event definition is assigned.
TRACKING_INTERVAL	Frequency, in seconds, with which a tracked event occurrence is tested for completeness.

Type-Specific Event Definition Options

These options are specific to event definitions with an event type of **FILE**.

Option Name	Description
MINIMUM_FILE_SIZE	Smallest size a file may be in order for it to be considered complete.
RENAME_FILE	Flag that indicates whether or not a completed file should be renamed.
RENAME_FILE_SPECIFICATION	Name or pattern to use when a file is renamed.

Event Handler Category Options

Option Name	Description
EXPIRED_HANDLER_ID	ID of a stored event handler record that should be used whenever an event is set to an expired state.
INTERACT_WITH_DESKTOP	Specifies whether or not the desktop of the current interactive logon session is accessible to the handler process.
LOGIN	Specifies whether or not to load a user's profile and environment when a handler process is executed.
LOGON_METHOD	Specifies how a user account is logged on when the environment for execution of a handler process is established.
MAXIMUM_RETURN_CODE	Highest return code than an event handler may exit with to still be considered as having executed successfully.
REJECTED_HANDLER_ID	ID of a stored event handler record that should be used whenever an event occurrence is set to a rejected state.
TRIGGERED_HANDLER_ID	ID of a stored event handler record that should be used whenever an event occurrence is set to a triggered state.
USER_SCRIPT_TYPE	Type of script interpreter used to evaluate and execute a script on behalf of an event handler.
USER_SECURITY	Specifies whether or not user account authorization is required to monitor an event and/or execute a handler process.

Installation Category Options

Option Name	Description
INSTALLATION_DIRECTORY	Base directory in which the product is installed

Message Category Options

Option Name	Description
MESSAGE_LEVEL	Level of messages printed.
NLS_ DIRECTORY	Directory location of message catalog and code page tables
TRACE_DIRECTORY	Specifies the directory in which trace files are written.
TRACE_FILE_LINES	Maximum number of lines written to a trace file before it wraps around.
TRACE_TABLE	Memory trace table specification.

Monitoring Category Options

Option Name	Description
POLLING_INTERVAL	Frequency with which a UEM Server will detect new occurrences of a system event. The UEM Server will also check at this time see if a monitored event should be made inactive.

Network Category Options

Option Name	Description
CODE_PAGE	Code page used for text translation.
KEEPALIVE_INTERVAL	Frequency with which a Keep-Alive message is sent to the UEM Manager.

8.5 Universal Access Control List

The UEM Server uses Universal Access Control Lists (UACLs) as an extra layer of security. UEM Server uses ACLs to grant or deny access to its own execution, execution of event handler processes, and database maintenance.

8.5.1 UACL Entries

The syntax of a UACL entry file is the same as the Universal Event Monitor configuration file.

Table 8.3, below, identifies all Universal Event Monitor for Windows UACL entries. Each **UACL Entry Name** is a link to detailed information about that UACL entry.

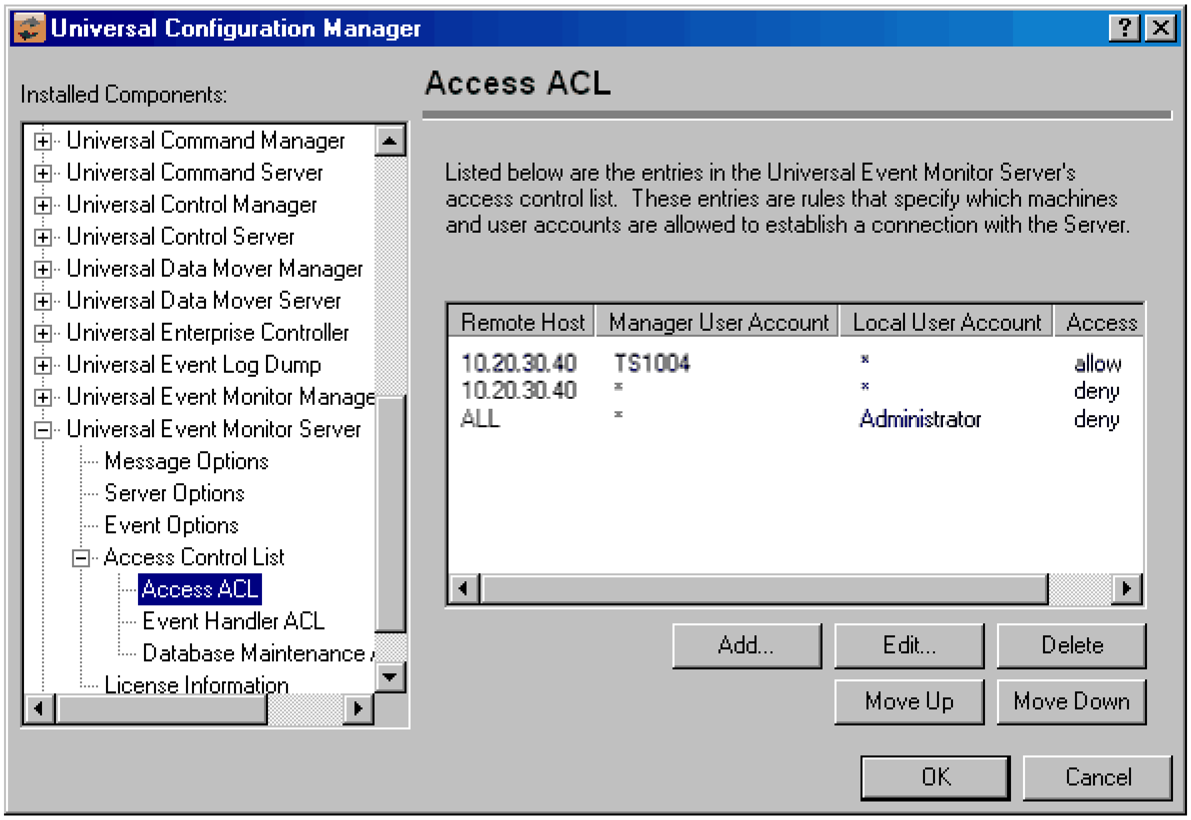
UACL Entry Name	Description
ACCESS_ACL	Allows or denies access to a demand-driven Universal Event Monitor Server.
DATABASE_MAINTENANCE_ACL	Allows or denies a user account access to the event definition and event handler databases.
EVENT_HANDLER_ACL	Allows or denies a user account the authority to execute an event handler process.

Table 8.3 Universal Event Monitor Server for Windows – UACL Entries

8.5.2 Updating the Universal Event Monitor Server ACL Entries

Although UACL files can be edited with any text editor (for example, Notepad), the Universal Configuration Manager application, accessible via the Control Panel, is the recommended way to update UACL entries (see Chapter 12 Universal Event Monitor UACL Entries). From there, ACL entries can be added, changed, deleted, or sorted (rules are applied in the order in which they are listed).

Figure 8.2, below, illustrates an example. The set of ACL entries only allows connections from host 10.20.30.40 if the user on that host is TS1004. All other remote users will be blocked. TS1004 may run processes on the local system using any user account, provided the correct password is supplied. No processes may be run with Universal Event Monitor using the Administrator account on the local system, regardless of where the request originated.



Listed below are the entries in the Universal Event Monitor Server's access control list. These entries are rules that specify which machines and user accounts are allowed to establish a connection with the Server.

Remote Host	Manager User Account	Local User Account	Access
10.20.30.40	TS1004	*	allow
10.20.30.40	*	*	deny
ALL	*	Administrator	deny

Figure 8.2 Universal Configuration Manager - Universal Event Monitor Server - Access ACL

CHAPTER 9

Universal Event Monitor Server for UNIX

9.1 Overview

This chapter describes the information for running the Universal Event Monitor Server under the UNIX operating system.

This information is separated into the following sections:

- Server Environment
- Component Definition
- Configuration
- Universal Access Control List

9.2 Server Environment

UEM Server is a Universal Broker-managed component, which means it only can be started and (gracefully) shutdown by the Universal Broker. It runs as a background process and, except for displaying version and copyright information, it provides no native user interface or console interaction.

9.2.1 Execution Context

The security context that a UEM Server executes with depends on UEM Server configuration options and whether the Server is demand-driven or event-driven (see Section 7.2 Demand-Driven vs. Event-Driven).

If the USER_SECURITY option is enabled in the UEM Server configuration, a demand-driven Server executes in the security context of the user account specified via the UEM Manager's command options.

If security is not enabled in the UEM Server configuration, it runs in the security context of the Universal Broker.

An event-driven UEM Server always executes in the security context of the Universal Broker, regardless of the value of the security configuration option.

HP-UX 11.00 and later

By default, supplemental group memberships are recorded in the **/etc/group** file. However, if an **/etc/logingroup** file exists, it governs all supplemental group memberships and effectively overrides the entries in **/etc/group**.

Note: **/etc/logingroup** is not required to record supplemental group membership. If **/etc/logingroup** does not exist, **/etc/group** is sufficient to record the groups in which a user belongs.

If any Stonebranch Solutions component fails to access system resources that are secured based on supplemental group membership, make sure that the authenticated user has an entry in **/etc/logingroup**, if that file exists. Otherwise, the default entry in **/etc/group** should be sufficient.

For more information about **/etc/logingroup**, please see the HP-UX system documentation.

9.3 Component Definition

All Stonebranch Solutions components managed by Universal Broker have a component definition. The component definition is a text file of options containing component-specific information required by Universal Broker. (For details on how Universal Broker manages components, see the Universal Broker 4.2.0 Reference Guide.)

The syntax of a component definition file is the same as a configuration file.

The UEM Server for UNIX component definition is located in the **`/etc/universal/comp`** directory.

Table 9.1, below, identifies all of the options that comprise the UEM Server for UNIX component definition.

Each **Option Name** is a link to detailed information about that component definition option.

Option Name	Description
AUTOMATICALLY_START	Specification for whether or not UEM Server is started automatically when Universal Broker is started
COMPONENT_NAME	Name by which clients know the UEM Server
COMPONENT_TYPE	Type of UEM Server (demand-driven or event-driven)
CONFIGURATION_FILE	Full path name of the UEM Server configuration file
RUNNING_MAXIMUM	Maximum number of UEM Servers that can run simultaneously
START_COMMAND	Full path name of the UEM Server program
WORKING_DIRECTORY	Full path name of the UEM Server working directory

Table 9.1 UEM Server for UNIX - Component Definition Options

9.4 Configuration

All UEM Servers defined on a particular system share a single configuration. A demand-driven UEM Server reads configuration options when it starts, and uses those values while it executes. Because an event-driven Server can run for an extended period of time, the configuration information it caches at start-up can be refreshed whenever the stored configuration options are refreshed (see Section 7.2.6 Configuration Refresh).

A UEM Server's configuration consists of default values that are used to establish the runtime environment and provide missing values for required event definition and event handler parameters.

Stored configuration options reside in the UEM Server configuration file. The name of this file can be set in the UEM component definition file. The default name for this file is **`uems.conf`**. This is a plain text file that can be edited with any text editor.

Configuration options stored in the configuration file are done so using a keyword/value combination. The keywords for each configuration option are included along with the description of the option.

The UEMLoad utility can override a configured default via its command options when it adds records to the event definition and event handler databases. Also, a UEM Manager can override options for a demand-driven UEM Server when it submits a monitoring request. The syntax for those overrides, when available, are included with the option's description.

9.4.1 Configuration Options

Table 9.2, below, categorizes available UEM Server configuration options according to function.

Category	Description
Event Definition	Allow default values to be set for any required event definition parameters that were omitted from the UEM Manager or UEMLoad utility request.
Event Handler	Allow default values to be set for any required event handler parameters that were omitted from the UEM Manager or UEMLoad utility request.
Installation	Options that specify installation requirements, such as directory locations
Message	UEM message options.
Monitoring	Control UEM activity.
Network	Manage data transmission between local and remote systems.

Table 9.2 UEM Server for UNIX - Configuration Options Categories

The UEM Server configuration options for each of the categories listed in Table 9.2 are summarized in the following tables. Each Option Name is a link to detailed information about that option.

Event Definition Category Options

Option Name	Description
ASSIGNED_COMPONENT_NAME	UEM Server component to which a stored event definition is assigned.
TRACKING_INTERVAL	Frequency, in seconds, with which a tracked event occurrence is tested for completeness.

Event Definition Options - Type-Specific

These options are specific to event definitions with an event type of **FILE**.

Option Name	Description
MINIMUM_FILE_SIZE	Smallest size a file may be in order for it to be considered complete.
RENAME_FILE	Flag that indicates whether or not a completed file should be renamed.
RENAME_FILE_SPECIFICATION	Name or pattern to use when a file is renamed.

Event Handler Category Options

Option Name	Description
EXPIRED_HANDLER_ID	ID of a stored event handler record that should be used whenever an event is set to an expired state.
LOGIN	Specifies whether or not to load a user's profile and environment when a handler process is executed.
MAXIMUM_RETURN_CODE	Highest return code than an event handler may exit with to still be considered as having executed successfully.
REJECTED_HANDLER_ID	ID of a stored event handler record that should be used whenever an event occurrence is set to a rejected state.
SHELL	Default shell interpreter for event handler processes executed as scripts.
TRIGGERED_HANDLER_ID	ID of a stored event handler record that should be used whenever an event occurrence is set to a triggered state.
USER_SECURITY	Specifies whether or not user account authorization is required to monitor an event and/or execute a handler process.

Installation Category Options

Option Name	Description
INSTALLATION_DIRECTORY	Base directory in which the product is installed

Message Category Options

Option Name	Description
MESSAGE_LEVEL	Level of messages printed.
NLS_ DIRECTORY	Directory location of message catalog and code page tables
TRACE_DIRECTORY	Directory where trace files are written.
TRACE_FILE_LINES	Maximum number of lines written to a trace file before it wraps around.
TRACE_TABLE	Memory trace table specification.

Monitoring Category Options

Option Name	Description
POLLING_INTERVAL	Frequency with which a UEM Server will detect new occurrences of a system event. The UEM Server also will check at this time see if a monitored event should be made inactive.

Network Category Options

Option Name	Description
CODE_PAGE	Code page used for text translation.
KEEPALIVE_INTERVAL	Frequency with which a Keep-Alive message is sent to the UEM Manager.

9.5 Universal Access Control List

The UEM Server uses Universal Access Control Lists (UACLs) as an extra layer of security. UEM Server uses ACLs to grant or deny access to its own execution, execution of event handler processes, and database maintenance.

9.5.1 UACL Entries

The syntax of a UACL entry file is the same as the Universal Event Monitor configuration file.

Table 9.3, below, identifies all Universal Event Monitor for UNIX UACL entries. Each **UACL Entry Name** is a link to detailed information about that UACL entry.

UACL Entry Name	Description
ACCESS_ACL	Allows or denies access to a demand-driven Universal Event Monitor Server.
DATABASE_MAINTENANCE_ACL	Allows or denies a user account access to the event definition and event handler databases.
EVENT_HANDLER_ACL	Allows or denies a user account the authority to execute an event handler process.

Table 9.3 Universal Event Monitor Server for UNIX – UACL Entries

CHAPTER 10

UEM Server Configuration Options

10.1 Overview

This chapter provides detailed information on the configuration options available for use with the Universal Event Monitor Server.

The options are listed alphabetically, without regard to any specific operating system.

Section 10.2 Configuration Options Information provides a guideline for understanding the information presented or each option.

10.2 Configuration Options Information

For each configuration option, this chapter provides the following information.

Description

Describes the option and how it is used.

Usage

Provides a table of the following information:

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Configuration File Keyword	<Format / Value>					
UEM Load Override	<Format / Value>					
UEM Manager Override	<Format / Value>					

Method

Identifies the different methods used to specify Universal Event Monitor Server configuration options:

- Configuration File Keyword
- UEMLoad Override
- UEM Manager Override

Note: Each option can be specified using one or more methods.

Syntax

Identifies the syntax of each method that can be used to specify the option:

- Format Specific characters that identify the option.
- Value Type of value(s) to be supplied for this method.

Note: If a Method is not valid for specifying the option, the Syntax field contains **n/a**.

(Operating System)

Identifies (with a ✓) the operating systems for which each method of specifying the option is valid:

- IBM i
- NonStop (HP NonStop)
- UNIX
- Windows
- z/OS

Values

Identifies all possible values for the specified value type.

Defaults are identified in **[bracketed bold type]**.

<Additional Information>

Identifies any additional information specific to the option.

10.3 Configuration Options List

Table 10.1 UEM Server Configuration Options identifies all UEM Server configuration options.

Option	Description	Page
ASSIGNED_COMPONENT_NAME	UEM Server component to which a stored event definition is assigned.	164
CODE_PAGE	Code page used for text translation	165
EXPIRED_HANDLER_ID	ID of a stored event handler record that should be used whenever an event is set to an expired state.	166
INSTALLATION_DIRECTORY	Base directory in which UEM Server is installed	167
INTERACT_WITH_DESKTOP	Specifies whether or not the desktop of the current interactive logon session is accessible to the handler process.	168
KEEPALIVE_INTERVAL	Frequency with which a Keep-Alive message is sent to the UEM Manager.	170
LOGIN	Specifies whether or not to load a user's profile and environment when a handler process is executed.	171
LOGON_METHOD	Specifies how a user account is logged on when the environment for execution of a handler process is established.	173
MAXIMUM_RETURN_CODE	Highest return code than an event handler may exit with to still be considered as having executed successfully.	174
MESSAGE_LEVEL	Level of messages printed.	175
MINIMUM_FILE_SIZE	Smallest size a file may be in order for it to be considered complete.	176
NLS_DIRECTORY	Directory location of message catalog and code page tables	177
POLLING_INTERVAL	Frequency with which a UEM Server will detect new occurrences of a system event. The UEM Server will also check at this time see if a monitored event should be made inactive.	178
REJECTED_HANDLER_ID	ID of a stored event handler record that should be used whenever an event occurrence is set to a rejected state.	179
RENAME_FILE	Flag that indicates whether or not a completed file should be renamed.	180
RENAME_FILE_SPECIFICATION	Name or pattern to use when a file is renamed.	181
SHELL	Default shell interpreter for event handler processes executed as scripts.	183
TRACE_DIRECTORY	Specifies the directory in which trace files are written.	184
TRACE_FILE_LINES	Maximum number of lines written to a trace file before it wraps around.	185
TRACE_TABLE	Memory trace table specification.	186
TRACKING_INTERVAL	Frequency, in seconds, with which a tracked event occurrence is tested for completeness.	188

Option	Description	Page
TRIGGERED_HANDLER_ID	ID of a stored event handler record that should be used whenever an event occurrence is set to a triggered state.	189
USER_SCRIPT_TYPE	Type of script interpreter used to evaluate and execute a script on behalf of an event handler.	190
USER_SECURITY	Specifies whether or not user account authorization is required to monitor an event and/or execute a handler process.	191

Table 10.1 UEM Server Configuration Options

10.4 ASSIGNED_COMPONENT_NAME

Description

The ASSIGNED_COMPONENT_NAME option specifies the event-driven UEM Server component to which new event definitions will be assigned if no such value is provided when the record is added with the UEMLoad utility.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Configuration File Keyword	comp_name *compname*				√	
UEM Load Override	-comp_name *compname*				√	
UEM Manager Override	n/a					

Value

compname is the name of the event-driven UEM Server.

[Default is *uems*.]

10.5 CODE_PAGE

Description

The CODE_PAGE option specifies the local character code page that is used to translate text data transmitted and received over the network.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Configuration File Keyword	codepage *codepage*			√	√	
UEM Load Override	n/a					
UEM Manager Override	n/a					

Value

codepage is the character code page that is used to translate data.

codepage references a Universal Translate Table (UTT) file provided with the product (see Section 17.3 UTT Files for information on UTT files). UTT files are used to translate between Unicode and the local single-byte code page. (All UTT files end with an extension of `.utt`.)

[Default

The default code page is different for different operating systems:

• **ISO8859-1 (8-bit ASCII)**	**ASCII-based operating systems**
• **IBM1047 (EBCDIC)**	**Non-IBM i, EBCDIC-based operating system]**

See Section 17.2 Character Code Pages for a complete list of character code pages provided by Stonebranch Inc. for use with Stonebranch Solutions.

10.6 EXPIRED_HANDLER_ID

Description

The EXPIRED_HANDLER_ID option specifies the ID of a default stored event handler record that should be used when an event expires.

The UEM Server stores this ID in records that are added to the event definition database if it is not overridden by the specified UEMLoad utility command option.

This ID also is used when both of the following occur:

1. Event monitored by a demand-driven Server expires.
2. Specified UEM Manager command option was omitted.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Configuration File Keyword	expired_id *id*			√	√	
UEM Load Override	-expired_id *id*			√	√	
UEM Manager Override	-expired -handler_id *id*			√	√	

Value

id is the ID to be used.

[There are no defaults.]

10.7 INSTALLATION_DIRECTORY

Description

The INSTALLATION_DIRECTORY option specifies the full path name of the directory in which the Universal Event Monitor Server is installed.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Configuration File Keyword	installation_directory *directory*			√	√	
UEM Load Override	n/a					
UEM Manager Override	n/a					

Value

directory is the full path name of the directory.

Defaults

UNIX

[Default is `/opt/universal/uemsrv`.]

Windows

[Default is `c:\Program Files\Universal\uemsrv`.]

10.8 INTERACT_WITH_DESKTOP

Description

The INTERACT_WITH_DESKTOP option specifies whether or not event handler processes are allowed to interact with the current console logon session.

If event handler processes are allowed to interact, they are run in a context that permits the current interactive console session to interact with them. This interaction can go both ways, as the process may invoke system functions that access desktop elements (for example, Windows, menus, and buttons) associated with the session. This is considered a security risk, in that it creates an opportunity for malicious code to hijack the desktop. If the security context of the interactive session is higher than that of the process, the process could invoke code using an elevated security context.

If event handler processes are not allowed to interact, they execute in a context that is isolated from the current interactive logon session. Unless event handler processes requiring user interaction are executed, this is the recommended value.

Stoneman's Tip

This option applies only when the LOGON_METHOD option is set to INTERACTIVE.

If user accounts are authenticated using the BATCH logon method, the established security context already disallows all interaction with the desktop.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Configuration File Keyword	interact_with_desktop *option*				√	
UEM Load Override	n/a					
UEM Manager Override	n/a					

Value

option is the specification for whether or not the event handler processes are allowed to interact:

Valid values for *option* are:

- **yes**
  Allow event handler processes to interact with the current console logon session.
- **no**
  Do not allow event handler processes to interact with the current console logon session.

[Default is no.]

10.9 KEEPALIVE_INTERVAL

Description

The KEEPALIVE_INTERVAL option specifies the frequency with which a Keep-Alive message is sent to the UEM Manager.

The Keep-Alive message is used to verify the presence of a connection between the UEM Server and UEM Manager during periods of network inactivity.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Configuration File Keyword	keep_alive_interval *frequency*			√	√	
Manager Override	n/a					

Values

frequency is the frequency (in seconds) with which a Keep-Alive message is sent.

[Default is 120.]

10.10 LOGIN

Description

The LOGIN option instructs the UEM Server whether or not to establish an environment, when executing a handler process, that resembles the environment that the user account (in whose security context the process executes) would have were the user actually to log in to the system on which the UEM Server resides.

Windows

If the environment is established, the user's environment block is loaded and the user's registry hive is mapped to HKEY_CURRENT_USER. This makes the user's profile available to the handler process.

UNIX

If the environment is established, the SHELL used to execute an event handler process is invoked as a login shell, which reads and executes commands from the system and user profiles. The profiles that are read and executed depends on the type of shell invoked.

AIX

The AIX platform provides a **`/etc/environment`** that allows global environmental variables to be exported for all users of a given machine. UEM Server adds the environment variables defined in this file to the user's login environment.

The order in which it is processed by UEM Server is slightly different than the order used by the AIX login process. The AIX login process reads and processes the following files in the order listed:

1. **`/etc/profile`**
2. **`/etc/environment`**
3. **`$HOME/.profile`**
4. **`$HOME/.env`**

The LOGIN option directs a UEM Server to use the login shell to execute the event handler process. The AIX login shell does not process the **`/etc/environment`** file, so UEM Server processes the files in the following order:

1. **`/etc/environment`**
2. **`/etc/profile`**
3. **`$HOME/.profile`**
4. **`$HOME/.env`**

Note: If the UEM Server USER_SECURITY option is set to *none*, the LOGIN option is ignored.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Configuration File Keyword	login *option*			√	√	
UEM Load Override	n/a					
UEM Manager Override	-login *option*			√	√	

Values

option is the indication for whether or not to establish the environment.

Windows

Valid value for *option* are:

- **yes**
  Login environment is established.
- **no**
  Non-login environment is established.

[Default is no.]

UNIX

Valid values for *option* are:

- **yes**
  Invoke the shell as a login shell.
- **no**
  Do not invoke the shell as a login shell.

[Default is no.]

10.11 LOGON_METHOD

Description

The LOGON_METHOD option specifies the logon method used when a user is logged on by UEM Server prior to execution of an event handler process.

Note: If the UEM Server's USER_SECURITY option is set to *none*, the LOGON_METHOD option is ignored.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Configuration File Keyword	logon *method*				√	
UEM Load Override	n/a					
UEM Manager Override	n/a					

Value

method is the logon method used.

Valid values for *method* are:

- **batch**
  Windows log on method is batch. A batch log on prevents the handler process from interacting with the desktop. The user account logging on as a batch user requires the Windows User Right: Log on as a batch job. If the user does not have this right, the log on action will fail.
- **interactive**
  Windows log on method is interactive. An interactive log on permits the handler process to interact with the desktop. No additional rights are required for a user to log on as an interactive user.

[Default is interactive.]

10.12 MAXIMUM_RETURN_CODE

Description

The MAXIMUM_RETURN_CODE option specifies the highest return code value that a process, executed on behalf of an event handler, can return still to be considered as having executed successfully.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Configuration File Keyword	maxrc *returncode*			√	√	
UEM Load Override	-maxrc *returncode*			√	√	
UEM Manager Override	-maxrc *returncode*			√	√	

Value

returncode is the return code value.

[Default is *0* (zero).]

10.13 MESSAGE_LEVEL

Description

The MESSAGE_LEVEL option controls which messages are issued by a UEM Server.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Configuration File Keyword	message_level *level*			√	√	
UEM Load Override	n/a					
UEM Manager Override	n/a					

Value

level is the level of messages to be issued by the UEM Server.

Valid values for *level* are:

- **trace**
  Issues trace messages to a file named **`uemsrv-nnnnnnnnnn-1.trc`**, where **nnnnnnnnnn** is the 10-digit component ID.

  Windows

  The trace file is located in the **`.\Universal\UEMSrv`** directory.

  UNIX

  The trace file is located in the **`./Universal/trace`** directory.

  Note: Use **trace** only for debugging, and only when requested by Stonebranch Inc. Customer Support.
- **audit**
  Issues audit, informational, warning, and error messages.
- **info**
  Issues informational, warning and error messages.
- **warn**
  Issues warning and error messages.
- **error**
  Issues error messages only.

[Default is info.]

10.14 MINIMUM_FILE_SIZE

Description

The MINIMUM_FILE_SIZE option specifies the smallest size that a file - tracked for an event with an event type of **`FILE`** - can be in order to be considered complete.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Configuration File Keyword	min_file_size *size[units]*					
UEM Load Override	-min_file_size *size[units]*			√	√	
UEM Manager Override	-min_file_size *size[units]*			√	√	

Value

size is the file size. *units* is the unit of storage for that *size*.

The valid values for *units*, and the maximum value that can be specified for *size* for that unit of storage, is:

- **b** bytes (4,294,967,295)
- **k** kilobytes (4,194,304)
- **m** megabytes (4,096)
- **g** gigabytes (4)

If a value for *units* is not specified, the file size is assumed to be in bytes.

[Default is *0* (zero).]

10.15 NLS_ DIRECTORY

Description

The NLS_DIRECTORY option specifies the name of the directory where the UEM Server message catalog and code page tables are located.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Configuration File Keyword	nls_directory *directory*			√	√	
UEM Load Override	n/a					
UEM Manager Override	n/a					

Values

directory is the name of the directory where the message catalog and code page tables are located.

Full path names are recommended.

Relative path names are relative to the `universal` installation directory.

Defaults

UNIX

[Default is `/opt/universal/nls`.]

Windows

[Default is `c:\Program Files\Universal\uemmgr`.]

10.16 POLLING_INTERVAL

Description

The POLLING_INTERVAL option specifies the frequency with which the Universal Event Monitor Server will check to see if an event's inactive date and time has elapsed.

If the inactive date and time has not elapsed, and the monitored event is to remain active, the UEM Server then will check for new occurrences of the event.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Configuration File Keyword	polling_int *secs*			√	√	
UEM Load Override	n/a					
UEM Manager Override	-polling_int *secs*			√	√	

Value

secs is the frequency (number of seconds) with which the UEM Server checks.

[Default is *30*.]

10.17 REJECTED_HANDLER_ID

Description

The REJECTED_HANDLER_ID option specifies the ID of a default stored event handler record that should be used when an event occurrence is rejected.

The UEM Server stores this value in records added to the event definition database if it is not overridden by the specified UEMLoad utility command option. This ID also is used for event occurrences that are rejected by a demand-driven Server if the specified UEM Manager command option was omitted.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Configuration File Keyword	rejected_id *id*			√	√	
UEM Load Override	-rejected_id *id*			√	√	
UEM Manager Override	-rejected_id *id*			√	√	

Value

id is the ID to be used.

[There is no default.]

10.18 RENAME_FILE

Description

The RENAME_FILE option specifies whether or not a file tracked for an event with an event type of **`FILE`** should be renamed by the UEM Server when the event occurrence is set to a **`triggered`** state.

Renaming a file ensures that UEM won't treat a file that it just finished tracking as a new event occurrence.

Note: This option is used only for events with an event type of **`FILE`**.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Configuration File Keyword	rename_file *option*			√	√	
UEM Load Override	-rename_file *option*			√	√	
UEM Manager Override	-rename_file *option*			√	√	

Value

option is the specification for whether or not the file should be renamed.

Valid values for *option* are:

- **yes**
  File is renamed according to the RENAME_FILE_SPECIFICATION option.
- **no**
  File is not renamed.

If *option* is **no**, you also must either:

- Set the value of the UEM Manager MAX_OCCURRENCE_COUNT option to *1*.
- Rename the file in a script invoked by an event handler when the occurrence is triggered. This will prevent multiple detection of the file by the UEM Server.

[Default is yes.]

10.19 RENAME_FILE_SPECIFICATION

Description

The RENAME_FILE_SPECIFICATION option specifies the file name that a Universal Event Monitor Server should use when both of the following occur:

- Event occurrence for an event with an event type of **`FILE`** is set to a **`triggered`** state.
- RENAME_FILE option is set to **yes**.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Configuration File Keyword	rename_filespec *renamespecification*			√	√	
UEM Load Override	-rename_filespec *renamespecification*			√	√	
UEM Manager Override	-rename_filespec *renamespecification*			√	√	

Values

renamespecification is the file name to be used. It can include an absolute path or a path that is relative to the location of the file monitored by UEM. If no path information is provided, the monitored file is simply renamed, and the renamed file will reside in the same location as the original file.

Variables that UEM will substitute with actual runtime values also can be included as part of *renamespecification*. These variables, and the values that UEM will substitute for them, are listed below.

- **`$(compname)`**
  Component name
- **`$(compid)`**
  Component ID
- **`$(date)`**
  Current date, in the format *YYYYMMDD*
- **`$(time)`**
  Current time, in the format *HHMMSS*
- **`$(origname)`**
  Original base file name, minus its last extension
- **`$(origext)`**
  Original file extension
- **`$(seqnum)`**
  A sequence number that is initialized to 0 when a UEM Server is started, and is then incremented by 1 for each file renamed.

If *renamespecification* contains spaces, it must be enclosed in double (**"**) quotation marks.

UNIX

If *renamespecification* contains spaces, it must be enclosed in either single (**'**) or double (**"**) quotation marks.

[Default is *$(compname).$(compid).$(date).$(seqnum)*.]

10.20 SHELL

Description

The SHELL option specifies the UNIX command shell that should be used to execute an event handler process.

This command shell must be specified with an argument that directs it to execute the given command and then exit.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Configuration File Keyword	shell *shell*			√		
UEM Load Override	n/a					
UEM Manager Override	n/a					

Value

shell is the name of the UNIX command shell.

[Default is `/bin/sh –c`.]

10.21 TRACE_DIRECTORY

Description

The TRACE_DIRECTORY option specifies the location where trace files generated by UEM Server are stored.

Trace files are generated when the MESSAGE_LEVEL option is set to *TRACE*.

A complete path, or a path relative to the UEM Server installation directory (for example, **`C:\Program Files\Universal\UEMSrv`**) can be specified.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Configuration File Keyword	trace_directory *directory*			√	√	
UEM Load Override	n/a					
UEM Manager Override	n/a					

Value

directory is the location where trace files are stored.

Windows

[Default is the UEM Server installation directory.]

UNIX

[Default is `/var/opt/universal/trace`.]

10.22 TRACE_FILE_LINES

Description

The TRACE_FILE_LINES option specifies the maximum number of lines to write to a trace file.

A trace file is generated when the MESSAGE_LEVEL option is set to *TRACE*. In this situation, the trace file will wrap around when the number of lines specified by this TRACE_FILE_LINES option has been reached.

New trace entries are written at the top of the trace file, just after the trace header lines. The trace entries are written to a file named **`uemsrv-nnnnnnnnnn-1.trc`** file, where **nnnnnnnnnn** is the 10-digit component ID assigned to the UEM Server process.

Windows

The trace file is located in the **`.\Universal\UEMSrv`** directory.

UNIX

The trace file is located in the **`./Universal/trace`** directory.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Configuration File Keyword	trace_file_lines *lines*			√	√	
UEM Load Override	n/a					
UEM Manager Override	n/a					

Value

lines is the maximum number of lines to write to a trace file.

When setting this value, allow for an average trace file line size of 50 characters.

[Default is *50,000*.]

10.23 TRACE_TABLE

Description

The TRACE_TABLE option specifies the size of a wrap-around trace table maintained in memory.

Tracing is activated when the MESSAGE_LEVEL option is set to TRACE. The trace table then is printed when the program ends under the conditions specified by this TRACE_TABLE option.

The trace file is named **`uemsrv-nnnnnnnnnn-1.trc`**, where **nnnnnnnnnn** represents the 10-digit component ID assigned to the UEM Server process when it is started by the Universal Broker.

Windows

The trace file is located in the **`.\Universal\UEMSrv`** directory.

UNIX

The trace file is located in the **`./Universal/trace`** directory.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Configuration File Keyword	trace_table *size[units][,cond]*			√	√	
UEM Load Override	n/a					
UEM Manager Override	n/a					

Value

size is the size of the trace table.

If *size* is set to *0* (zero), the trace table is not used.

units is the unit of storage for that *size*.

The valid values for *units*, and the maximum value that can be specified for *size* for that unit of storage, is:

- **b** bytes (2,147,483,647)
- **k** kilobytes (2,097,152)
- **m** megabytes (2,048)
- **g** gigabytes (2)

If a value for *units* is not specified, the file size is assumed to be in bytes.

[Default is 0.]

cond specifies the condition under which the trace table is written.

Valid values for *cond* are:

- **error**
  Writes the trace table if the program ends with a non-zero return code.
- **always**
  Writes the trace table when the program ends regardless of the return code.
- **never**
  Never write the trace table.

[Default is never.]

10.24 TRACKING_INTERVAL

Description

The TRACKING_INTERVAL option specifies the frequency with which a Universal Event Monitor Server tests for the completion of a tracked event occurrence.

The test for completeness depends upon the type of event that the tracked occurrence represents. For example, an occurrence tracked for an event with an event type of **FILE** is considered complete when two consecutive checks on a file's size return the same value.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Configuration File Keyword	tracking_int *secs*			√	√	
UEM Load Override	-tracking_int *secs*			√	√	
UEM Manager Override	-tracking_int *secs*			√	√	

Value

secs is the frequency (number of seconds) with which the UEM Server tests for completion of an event occurrence.

[Default is *10*.]

10.25 TRIGGERED_HANDLER_ID

Description

The TRIGGERED_HANDLER_ID option specifies the ID of a default stored event handler record that should be used when an event occurrence is triggered.

The UEM Server stores this ID in records that are added to the event definition database if it is not overridden by the specified UEMLoad utility command option.

The ID also is used for event occurrences triggered by a demand-driven UEM Server if the specified UEM Manager command option was omitted.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Configuration File Keyword	triggered_id *id*			√	√	
UEM Load Override	-triggered_id *id*			√	√	
UEM Manager Override	-triggered_id *id*			√	√	

Value

id is the ID to be used.

[There is no default.]

10.26 USER_SCRIPT_TYPE

Description

The USER_SCRIPT_TYPE option specifies the command processor that UEM Server uses to execute an event handler process on Windows.

The value specified for USER_SCRIPT_TYPE is a file extension. To execute the event handler process, UEM Server invokes the default program or command interpreter assigned to open files with that extension.

For example:

- To have UEM Server execute a script containing Windows command statements, set USER_SCRIPT_TYPE to `bat`.
- To have UEM Server execute a script with Perl statements, set USER_SCRIPT_TYPE to an extension (for example, `pl`) that, by default, invokes a Perl interpreter when opened.

If no association exists between a USER_SCRIPT_TYPE's corresponding file extension and a default program, the script fails.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Configuration File Keyword	script_type *type*				√	
UEM Load Override	-script_type *type*				√	
UEM Manager Override	-script_type *type*			√	√	√

Value

type is a file extension that describes the script's type.

[Default is *bat*.]

10.27 USER_SECURITY

Description

The USER_SECURITY option specifies user security usage for the UEM Server.

The way in which this option is used depends on whether a UEM Server is running in demand-driven or event-driven mode.

For a demand-driven UEM Server, enabling this option means an authenticated local user account is required to start the Server and monitor an event. All event monitoring activity, including execution of event handler processes, is done in the security context of this user.

For an event-driven UEM Server, enabling this option means that before an event handler process can be executed, a local user account and password must be stored in the event handler record (remember that an event-driven Server relies completely upon stored event definition and event handler records for its input). If the stored account information is valid, any processes executed on behalf of the event handler will be done so in the security context of that account.

Windows

This option should be set only via the Universal Configuration Manager; check the box labelled “Require user ID and password to start process” on the Server Options page of the Universal Event Monitor Server configuration control.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Configuration File Keyword	security *method*			√	√	
UEM Load Override	n/a					
UEM Manager Override	n/a					

Values

method is the user authentication method to be used.

Windows

Valid values for *method* are:

- **default**
  Windows authenticates provided account information, requiring a user ID and password to do so.
- **none**
  Disable user security.

UNIX

Valid values for *method* are:

- **default**
  Default UNIX user authentication method, which relies upon the **`/etc/passwd`** file.
- **pam**
  Pluggable Authentication Module interface
- **trusted**
  HP Trusted Security authentication
- **none**
- Disable user security.

CHAPTER 11

Universal Event Monitor Component Definition Options

11.1 Overview

This chapter provides detailed information about the options that comprise Universal Event Monitor (UEM) component definitions.

The options are listed alphabetically, without regard to any specific operating system.

Information on how component definitions are used is documented in the Indesca and Infitran 4.2.0 User Guides.

Section 11.2 Component Definition Options Information provides a guideline for understanding the information presented for each component definition option.

11.2 Component Definition Options Information

For each component definition option, this chapter provides the following information.

Description

Describes the option and how it is used.

Usage

Provides a table of the following information:

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Component Definition Keyword	<Format / Value>					

Method

Identifies the method used for specifying a Universal Event Monitor component definition option:

- Component Definition Keyword

Syntax

Identifies the syntax of the method used to specify the option:

• Format	Specific characters that identify the option.
• Value	Type of value(s) to be supplied for this method.

(Operating System)

Identifies (with a ✓) the operating systems for which the method of specifying the option is valid:

- IBM i
- NonStop (HP NonStop)
- UNIX
- Windows
- z/OS

Values

Identifies all possible values for the specified value type.

Defaults are identified in **[bracketed bold type]**.

11.3 Component Definition Options

Table 11.1 identifies all Universal Event Monitor component definitions.

Component	Description	Page
AUTOMATICALLY_START	Specification for whether or nor the UEM Server starts automatically when the Universal Broker is started	197
COMPONENT_NAME	Name by which clients know the UEM Server	198
COMPONENT_TYPE	Type of UEM Server (demand-driven or event-driven)	199
CONFIGURATION_FILE *	Full path name of the UEM Server configuration file	200
RUNNING_MAXIMUM	Maximum number of UEM Servers that can run simultaneously	201
START_COMMAND *	Full path name of the UEM Server executable file	202
WORKING_DIRECTORY *	Full path name of the UEM Server working directory	204
* These options are required in all component definitions.		

Table 11.1 Universal Event Monitor - Component Definition Options

11.4 AUTOMATICALLY_START

Description

The AUTOMATICALLY_START option indicates whether or nor UEM Server is started automatically when the Universal Broker is started.

Note: AUTOMATICALLY_START is optional in a component definition.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Component Definition Keyword	auto_start *option*			√	√	

Values

option is the specification for how the UEM Server is started.

Valid values for *option* are:

- **yes**
  UEM Server is started automatically when the Universal Broker is started.

  Note: Select **yes** for an event-driven UEM Server. The attempt to start a UEM Server automatically will succeed only if the values for the COMPONENT_TYPE and START_COMMAND component definition options are compatible with an event-driven UEM Server.
- **no**
  UEM Server is not started automatically when the Universal Broker is started.

  Note: Select **no** for a demand-driven UEM Server and for any event-driven UEM Server that should not be started automatically.

[Default is yes.]

11.5 COMPONENT_NAME

Description

The COMPONENT_NAME option specifies the name of the UEM Server.

Component start requests refer to the UEM Server by this name.

Note: COMPONENT_NAME is optional in a component definition. If it is not specified, the file name is used as the component name.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Component Definition Keyword	component_name *name*			√	√	

Values

name is the name by which the clients know the UEM Server:

- Name of the demand-driven UEM Server is **`uemd`**.
- Name of the default event-driven UEM Server is **`uems`**.

11.6 COMPONENT_TYPE

Description

The COMPONENT_TYPE option identifies whether the component definition is for a demand-driven or event-driven UEM Server.

Note: COMPONENT_TYPE is optional in a component definition. If it is not specified, the component name is used.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Component Definition Keyword	component_type *type*			√	√	

Values

type is the type of component.

- For demand-driven UEM Servers, *type* is **`uemd`**.
- For event-driven UEM servers, *type* is **`uems`**.

Note: This value is used for internal validation by the Universal Broker and should not be changed.

11.7 CONFIGURATION_FILE

Description

The CONFIGURATION_FILE option specifies the full path name of the UEM Server configuration file.

Note: CONFIGURATION_FILE is required in a component definition.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Component Definition Keyword	configuration_file *filename*			√	√	

Values

filename is the name of the configuration file.

filename can be any valid file name.

Defaults

UNIX

Installation default is **`/etc/universal/uems.conf`**.

Windows

Installation default is **`C:\Documents and Settings\All Users\Universal\conf\uems.conf`**.

11.8 RUNNING_MAXIMUM

Description

The RUNNING_MAXIMUM option specifies the maximum number of UEM Servers that can run simultaneously.

If this maximum number is reached, any command received to start a UEM Server is rejected.

Note: RUNNING_MAXIMUM is optional in a component definition.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Component Definition Keyword	running_max *max*			√	√	

Values

max is the maximum number of UEM Servers that can run simultaneously.

[Default is *100*.]

11.9 START_COMMAND

Description

The START_COMMAND option specifies the full path name to the UEM Server program.

START_COMMAND also specifies whether the UEM Server is to run in demand-driven mode or event-driven mode.

- For a demand-driven UEM Server (that is, a UEM Server whose COMPONENT_TYPE is set to **`uemd`**), the following parameter and value are required as part of this option: -demand **yes**.
- For an event-driven UEM Server, (that is, a UEM Server whose COMPONENT_TYPE is set to **`uems`**), the -demand parameter must be omitted from this option.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Component Definition Keyword	start_command *name* -demand *option*			√	√	

Values

name is the full path name of the UEM Server program.

option is the specification for whether the UEM Server is to be run in demand-driven mode or event-driven mode.

Defaults

UNIX

- For a demand-driven UEM Server: **`INSTALLDIR/universal/uemsrv/bin/uemsrv -demand yes`**, where **`INSTALLDIR`** is the primary installation directory (for example, **`/opt`**).
- For an event-driven UEM Server: **`INSTALLDIR/universal/uemsrv/bin/uemsrv`**, where **`INSTALLDIR`** is the primary installation directory (for example, **`/opt`**).

Windows

- For the demand-driven UEM Server: **`INSTALLDIR\Universal\UEMSrv\bin\uemsrv.exe -demand yes`**, where **`INSTALLDIR`** is the primary installation directory (for example, **`C:\Program Files`**).
- For the event-driven UEM Server: **`INSTALLDIR\Universal\UEMSrv\bin\uemsrv.exe`**, where **`INSTALLDIR`** is the primary installation directory (for example, **`C:\Program Files`**).

Note: These values are set during product installation and should not be changed.

11.10 WORKING_DIRECTORY

Description

The WORKING_DIRECTORY option specifies the full path name used as the working directory of UEM Server.

Note: WORKING_DIRECTORY is required in a component definition.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
Component Definition Keyword	working_directory *directory*			√	√	

Values

directory is the full path name of the working directory.

UNIX

By default, the value of this option is **`INSTALLDIR/uemsrv`**, where **`INSTALLDIR`** is the primary installation directory (for example, **`/opt/universal`**).

When a demand-driven UEM Server is run with the USER_SECURITY configuration option set to *yes*, the working directory is the home directory of the user account specified by the UEM Manager's command options.

The working directory for all event-driven UEM Servers and demand-driven UEM Servers run when the security configuration option is set to *no* is the one specified by this parameter.

Windows

By default, the value of this parameter is **`INSTALLDIR\Universal\UEMHome`**, where **`INSTALLDIR`** is the primary installation directory (for example, **`C:\Program Files`**). If this directory does not exist, it is created during product installation.

When a demand-driven UEM Server is run with the USER_SECURITY option set to **yes**, the actual working directory is a subdirectory of the directory specified by this WORKING_DIRECTORY option. This subdirectory's name matches the value of the user ID specified by the UEM Manager.

The working directory for all event-driven UEM Servers, and demand-driven UEM Servers run when the USER_SECURITY option is set to **no**, is the one specified by this option.

CHAPTER 12

Universal Event Monitor UACL Entries

12.1 Overview

This chapter provides detailed information on the Universal Access Control List (UACL) entries available for use with Universal Event Monitor.

The UACL entries are listed alphabetically, without regard to any specific operating system.

Information on how these UACL entries are used is documented in the Indesca and Infitran 4.2.0 User Guides.

Section 12.2 UACL Entries Information provides a guideline for understanding the information presented for each UACL entry.

12.2 UACL Entries Information

For each UACL entry, this chapter provides the following information.

Description

Describes the UACL entry and how it is used.

Usage

Provides a table of the following information:

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
UACL File Keyword	<Type / Rule>					

Method

Identifies the method used for specifying a UACL entry:

- UACL FIle Keyword

Syntax

Identifies the syntax of the method used for a UACL entry:

• Type	Stonebranch Solutions component to which the rule applies.
• Rule	Client’s identity, request to which the entry pertains, and security attributes that the entry enforces.

(Operating System)

Identifies (with a ✓) the operating systems for which the method of specifying the UACL entry is valid:

- IBM i
- NonStop (HP NonStop)
- UNIX
- Windows
- z/OS

Values

Identifies all possible values for the fields in a UACL entry rule.

Defaults are identified in **[bracketed bold type]**.

12.3 UACL Entries List

Table 12.1 identifies all Universal Event Monitor UACL entries.

UACL Entry	Description	Page
ACCESS_ACL	Allows or denies access to a demand-driven Universal Event Monitor Server.	209
DATABASE_MAINTENANCE_ACL	Allows or denies a user account access to the event definition and event handler databases.	211
EVENT_HANDLER_ACL	Allows or denies a user account the authority to execute an event handler process.	212

Table 12.1 Universal Event Monitor UACL Entries

12.4 ACCESS_ACL

Description

The ACCESS_ACL option is used to allow or deny access to a demand-driven Universal Event Monitor Server. These rules are applied anytime a request to start a demand-driven Server is made, regardless of which Universal Event Monitor client application (that is, the UEMLoad utility or UEM Manager) issued the request.

Access is allowed or denied based on the following attributes:

- IP address or host name of the system on which the UEM client application is executing (*host*)
- ID of the user account with which a UEM Manager is executing (*remote_user*)
- ID of a user account, specified via the USER_ID parameter of the UEM Manager, that is defined to the system on which the UEM Server is executing (*local_user*)

UNIX

An Access ACL entry also uses the value specified by **`auth`** to determine whether the local user account must be authenticated in order for the UEM Server to continue.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
UACL File Keyword	uem_access *host,remote_user, local_user,access, auth*			√	√	

Value

access is the specification for whether or not access is permitted.

Valid values for *access* are:

- **allow**
  Access is permitted.
- **deny**
  Access is not permitted. A message is returned to the client application (either the UEMLoad utility or UEM Manager), and the connection between the client and the Server is closed.

UNIX

auth is the specification for whether or not the local user account must be authenticated.

Valid values for *auth* are:

- **auth**
  Local user account must be authenticated. The UEM Manager must provide a proper password for the account.
- **noauth**
  Local user account does not require user authentication. The UEM Manager still must supply a password to satisfy command syntax rules, but it will not be verified; any password value will suffice.

  CAUTION: **noauth** should be used with care. Turning off user authentication may violate your local security policies on the UEM Server system.

12.5 DATABASE_MAINTENANCE_ACL

Description

The DATABASE_MAINTENANCE_ACL option allows or denies a user account access to the event definition and event handler databases.

Access is granted or denied based on the following attributes:

- ID of the user account executing the Universal Event Monitor Load utility (**`local_user`**)
- Type of database access requested (**`add`**,**`update`**,**`delete`**,**`list`**)

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
UACL File Keyword	uem_maintenance *local_user,add,update,delete,list*			√	√	

Value

Valid values for *add*, *update*, *delete*, and *list* are:

- **allow**
  Access is permitted for the specified operation for the user account identified by **`local_user`**.
- **deny**
  Access is not permitted for the specified operation. If this operation is attempted by the user account identified by **`local_user`**, a message is returned to the UEMLoad utility and the connection between the client and the UEM Server is closed.

12.6 EVENT_HANDLER_ACL

Description

The EVENT_HANDLER_ACL option allows or denies a user account the authority to execute an event handler process.

Usage

Method	Syntax	IBM i	NonStop	UNIX	Windows	z/OS
UACL File Keyword	uem_handler *user,access,auth*			√	√	

Values

user is the name of the user account.

Windows

For a demand-driven Server, *user* is the one specified from the Universal Event Monitor Manager's command options. For an event-driven Server, the *user* checked is the one stored in the event handler record.

For either type of Server, *user* is checked only if the USER_SECURITY configuration option is not set to **none**.

access specifies the security context of the event handler processes.

Valid values for *access* are:

- **allow**
  Event handler processes can be run in the security context of the user account specified by *user*.
- **deny**
  Event handler processes cannot be run in the security context of the user account specified by *user*.

UNIX

An Event Handler ACL entry also uses the value specified by *auth* to determine whether the user account must be authenticated with a password before the handler process can be executed.

Valid values for *auth* are:

- **auth**
  User account must be authenticated.
  - For a demand-driven UEM Server, a proper password must be provided by the UEM Manager for the account.
  - For an event-driven UEM Server, a valid user ID and password must be available in the stored event handler record.
- **noauth**
  The user account does not require user authentication. A password still may be necessary to satisfy command syntax rules, but it will not be verified. Any password value will suffice.

  Note:**noauth** should be used with care. Turning off user authentication may violate your local security policies on the Server system.

CHAPTER 13

UEMLoad Utility

13.1 Overview

A Universal Event Monitor (UEM) Server has three database files that it can use during event processing:

1. **`ueme.db`** stores event definitions.
2. **`uemh.db`** stores event handlers.
3. **`uems.db`** is a spool file that records all activity related to event monitoring.

The UEMLoad utility (**`uemload`**) manages the event definition and event handler database files. (For information on the spool database file, see Chapter 7 Universal Event Monitor Server.)

13.2 Database Files

This section describes the contents of the following database files:

- Event definition
- Event handler

For operating system-specific information on the UEMLoad utility, see Chapter 14 UEMLoad Utility for Windows and 15 UEMLoad Utility for UNIX.

13.2.1 Database Files Location

The event definition and event handler database files are local to each system. The files are created automatically when the initial attempt to open them is made. They reside in a directory that, if it does not already exist, is created during product installation.

UNIX

The default database directory is **/var/opt/universal/spool**.

Windows

The default database directory is **C:\Program Files\Universal\spool**.

The event definition and event handler database files reside in a subdirectory named **ubroker**. This subdirectory is used only on Windows installations to improve database performance. The name chosen for the directory signifies that all access to the event definition and event handler databases is routed through Universal Broker.

Additional installation recommendations and requirements for all database files used by Stonebranch Solutions components can be found in the Stonebranch Solutions 4.2.0 Installation Guide.

13.3 Event Definition Database File

Event definitions are stored in the database file **`ueme.db`**.

This section describes the parameters that comprise a record in the event definition file.

13.3.1 Event Definition Parameters

For UEM, an event definition represents a system event.

The parameters in an event definition record:

- Describe a system event.
- Establish the criteria that UEM uses to test for the completion of an event occurrence.

An event definition record also can contain the ID of a record in the event handler database, which dictates how an event or an event occurrence is handled (that is, responded to) when certain conditions are met.

Event Definition Parameters - Categories

Event definition parameters fall into two categories:

1. Parameters that are present in all event definition records. The value of one of these general parameters, Event Type, dictates the parameters that fall under the second category.
2. Parameters that are specific to an Event Type. Currently, UEM supports a single event type, **`FILE`**, which detects the creation of a file.

13.3.2 Event Definition Parameters - General

Table 13.1, below, describes the general event definition parameters.

The parameters that make up the record's key are underlined.

Parameter	Description	Remarks
Event ID	Unique identifier for the event definition.	Length of the handler ID cannot be greater than 32 characters. This value is case-sensitive.
Assigned UEM Component	Name of an event-driven Universal Event Monitor Server component responsible for monitoring the event.	
Enabled Flag	A *true* / *false* value that determines whether an event-driven UEM Server processes the event definition.	If *true*, the event is checked periodically by its assigned event-driven UEM Server to see if it should be made active or inactive. If *false*, the event, while still included in a UEM Server's list of assigned components, is not checked to see if it should be made active. [Default is *true*.]
Active Flag	A *yes* / *no* value that indicates whether UEM Server is monitoring the event.	If *yes*, the current date/time is greater than the defined **Activation Date/Time**, but less than the defined **Inactivation Date/Time**. If *no*, the current date/time is greater than the defined **Inactivation Date/Time**. This parameter is set only through UEM Server. It cannot be set using the UEMLoad utility. [Default is *no*.]
Event Type	System event represented by the event definition.	Value corresponds to one of the supported event types. (Currently, **FILE** is the only support event type.)
Activation Date	Date on which UEM will begin checking for the occurrence of the system event represented by the event definition.	Specified using the format *YYYY.MM.DD*. [Default is current date.]
Activation Time	Time on the **Activation Date** at which UEM will begin checking for the occurrence of the system event represented by the event definition.	Specified using the format *HH:MM*. [Default is current time.]
Inactivation Date	Date on which UEM will stop checking for the occurrence of the system event represented by the event definition.	Specified using the format *YYYY.MM.DD*. [Default is *2038.01.16* for event definitions stored with the UEMLoad utility.
Inactivation Time	Time on the **Inactivation Date** at which UEM will stop checking for the occurrence of the system event represented by the event definition.	Specified using the format *HH:MM*. [Default is *23:59*.]

Parameter	Description	Remarks
Tracking Interval	Frequency with which UEM tests for the completion of an occurrence of the system event represented by the event definition.	Value is expressed in seconds. Value depends on the event type. Some system events will be considered complete as soon as their occurrence is detected by UEM. For these events, this parameter should be *0* (zero). [Default is *10*.]
Triggered Event Handler ID	ID of an event handler database record whose specified actions should be taken when an event occurrence satisfies its defined completion criteria.	Value is case sensitive. Value can be blank, in which case the default value specified in the UEM Server configuration (empty string for Windows, *NONE* for UNIX) will be used.
Rejected Event Handler ID	ID of an event handler database record whose specified actions should be taken if one or more tracked event occurrences fail to complete before the event is made inactive by UEM.	Value is case sensitive. Value can be blank, in which case the default value specified in the UEM Server configuration (empty string for Windows, *NONE* for UNIX) will be used.
Expired Event Handler ID	ID of an event handler database record whose specified actions should be taken if an event is made inactive with no occurrence of the system event represented by the event definition being detected by UEM.	Value is case sensitive. Value can be blank, in which case the default value specified in the UEM Server configuration (empty string for Windows, *NONE* for UNIX) will be used.
Handler Options	String that contains one or more parameters that UEM adds to the command line it constructs to execute an event handler's specified command or script.	Added to the command line for every process executed for the event handlers referenced by this event definition. This includes those event handlers referenced by the Triggered Event Handler ID, Rejected Event Handler ID, and Expired Event Handler ID fields. When command line options are specified in the event definition (via Handler Options) and in the event handler (via Options), both are used. However, the event handler's Options value is added first.
Last Update Date/Time	Date and time the event definition was created or last modified.	
Last Update User	ID of the user account that created or most recently updated the event definition.	

Table 13.1 Event Definition Parameters - General

13.3.3 Event Definition Parameters - Event Type FILE

Table 13.2, below, describes the parameters that are available in event definitions for an Event Type of **`FILE`**.

Parameter	Description	Remarks
File Specification	Complete path of the file to watch.	File specification can contain wildcards: • ? is available to match up to 1 character. • * can be used to match 0 or more characters.
Minimum File Size	Smallest size that a file can be in order to be considered complete.	Value of *0* indicates that a file of any size is acceptable. To specify a storage unit for this value, add either of the following after the file size: • (b)ytes • (k)ilobytes • (m)egabytes • (g)igabtyes If a storage unit is not specified, UEM assumes the file size is given in bytes.
Rename File	Indicates whether or not UEM Server will rename the file prior to executing the triggered event handler. This will prevent multiple detections of the same file.	Acceptable values are *yes* or *no*. [Default is *yes*.]

Parameter	Description	Remarks
Rename File Specification	Format used by UEM to rename a file.	Complete file name or a file mask that accepts specific well-known variables that will be substituted by UEM Server at run time. When specified, variables must be in the format **`$(var)`**. The following variables are available: • **`$(compname)`** – component name • **`$(compid)`** - component ID • **`$(date)`** - current date in the format YYYYMMDD • **`$(time)`** - current time, n the format HHMMSS • **`$(origname)`** - original base file name, minus its last extension • **`$(origext)`** - original file extension • **`$(seqnum)`** - sequence number that starts at 0 when a UEM Server component is started, and is then incremented by 1 for each file renamed. If no path is specified, the file is simply renamed. Otherwise, the file can be renamed, moved from its original location, and placed in the path specified by this parameter. If no value is specified, a default value of **`$(compname).$(compid).$(date).$(seqnum)`**, specified in the UEM Server configuration, will be used.

Table 13.2 Event Definition Parameters for Events of Type **`FILE`**

13.4 Event Handler Database File

Event handlers are stored in the database file **`uemh.db`**.

This section describes the parameters that comprise a record in the event handler file.

13.4.1 Event Handler Parameters

For UEM, an event handler represents the action that should be taken in response to a monitored event.

An event handler record consists of parameters that:

- Describe the action to take.
- Establish an execution environment for the action.
- Define success or failure criteria for the action's outcome.

UEM Server executes the specified action on behalf of the event handler. Accordingly, the action also can be referred to as the event handler process.

Table 13.3, below, describes the event handler parameters.

The parameters that make up the record's key are <u>underlined</u>.

Parameter	Description	Remarks
<u>Handler ID</u>	Unique identifier for the event handler.	Length of the handler ID cannot be greater than 32 characters. Value is case-sensitive.
Handler Type	Type of action that will be executed.	If the Handler Actions field contains a system command or other command line application, this field has a value of *cmd*. If the Handler Actions field contains a series of system commands that should be executed as a script file, this field has a value of *script*.
Script Type	Script interpreter to invoke to execute script statements contained in the Handler Actions parameter.	On Windows-based systems, this value will be used as a file extension for the temporary script constructed by UEM Server in order to execute the script statements specified in the Handler Actions parameter. To insure proper execution of the script, a file association should exist on the target system that specifies the application with which the script should be executed. This parameter is ignored on UNIX-based systems. This parameter is ignored if the Handler Type has a value of *cmd*.

Parameter	Description	Remarks
Maximum Acceptable Return Code	Highest value that the handler process can return to still be considered as having executed successfully.	If the value returned by the action is less than or equal to the value specified by this parameter, UEM will report a successful completion of the handler process. If the value returned by the action is greater than the value specified by this parameter, UEM will report a failure of the handler process.
User ID	ID of the user account in whose security context the handler actions will be run.	If the UEM Server processing this record is configured to run without security, the value in this parameter is ignored.
Password	Password of the user account specified in the User ID parameter.	If the UEM Server processing this record is configured to run without security, this parameter is ignored. On UNIX-based systems, a UACL entry can be defined that permits handler execution for some user accounts without requiring authentication. In such a situation, this parameter is ignored. This parameter is encrypted prior to being stored in the database.
Encrypted File	Complete path to a file encrypted with Universal Encrypt. The userid and password options can be stored in an external file instead of specifying the handler userid and password parameters. UEM reads the contents of this file to obtain the User ID and Password of a user account, which then is used to establish a security context in which the event handler process is executed. If an encryption key was used other than uencrypt's default, that key can be specified with the Encrypted Key parameter.	A Password may not be necessary on UNIX, depending on the existing EVENT_HANDLER_ACL entries. Storing the path to an encrypted file allows the file to be shared between Stonebranch Solutions applications. This makes it unnecessary to update individual event handler records whenever account information is updated or security requirements change.
Encrypted Key	Optional encryption key used to decrypt the encrypted file.	This must be the same key that was used to encrypt the file.
Handler Actions	Command or script executed by UEM Server for this handler.	For records with a Handler Type of *cmd*, this parameter contains the syntax required to execute a system command or other command line application. For records with a Handler Type of *script*, this parameter contains one or more system commands that are executed collectively as a batch script.
Last Update Date/Time	Date and time that the event handler was created or last modified.	
Last Update User	ID of the user account that created or most recently updated the event handler.	

Parameter	Description	Remarks
Options	String with one or more parameters that UEM adds to the command line that it constructs in order to execute a specified command or script.	This parameter serves the same purpose as the event definition's Handler Options parameter, but differs in scope. An event definition's Handler Options are applied to every handler process executed on its behalf. An event definition's Handler Options may specify global handler execution options (for example, output redirection) or may customize event handler behavior for that specific event. On the other hand, an event handler's Options value is applied every time the handler is executed, regardless of which event definition causes it to execute. When command line options are specified in the event definition (via Handler Options) and in the event handler (via Options), both are used. However, the event handler's Options value is added first.

Table 13.3 Event Handler Parameters

CHAPTER 14

UEMLoad Utility for Windows

14.1 Overview

This chapter provides information on Universal Event Monitor (UEM) Load utility, specific to the Windows operating system.

Note: Because the UEM Server component is the only UEM component that uses the databases managed by UEMLoad, and because the UEM Server is available only on UNIX and Windows, UEMLoad also is available only on those operating systems. The command syntax and general usage of UEMLoad is the same on both operating systems, except where noted.

UEMLoad manages records in the event definition and event handler databases. The records in these databases then are used as input to a local UEM Server.

UEMLoad is capable of processing multiple event definition and/or event handler records when the parameters for those records are supplied using a text load file. If no load file is specified, the parameters for a single event definition and/or event handler record may be specified from the command line.

Stoneman's Tip

Although UEMLoad can access only local event definition and event handler databases, it is possible to store definition load files in a single location (for example, a PDS on z/OS) and distribute them to remote systems via Universal Command.

Simply redirect the definition load file from stdin and have Universal Command execute UEMLoad on the remote system. UEMLoad will read the redirected input and process it just as it would a local definition load file. This simplifies central administration of remote databases, because definition load files do not have to be stored (and managed) across several systems.

14.2 Usage

The UEMLoad utility, executes as a command line application.

The syntax for invoking UEMLoad from the command line requires the name of the program, **`uemload`**, followed by a list of configuration options.

This section describes the configuration and configuration options, command input, and command line syntax of UEMLoad for Windows.

14.2.1 Configuration

Configuration consists of:

- Setting default options and preferences for all executions of UEMLoad.
- Setting options and preferences for a single execution of UEMLoad.

UEMLoad for Windows receives its configuration options from the following sources:

1. Command line
2. Environment variables

The order of precedence is the same as the list above; command line options being the highest and environment variables being the lowest. That is, options specified via a command line override options specified via environment variables.

Definition and Event Handler Parameters

The configuration options source list, above, applies only to those general configuration options used to control execution of UEMLoad.

For event definition and/or event handler parameters, input can come from one of three mutually exclusive sources:

1. Command line
2. Definition load file
3. Definition load file redirected from **`stdin`**

Definition Load File

When record parameters are provided from the command line, only one event definition and/or event handler can be specified. To add, update, or delete multiple event definition and/or event handler records at the same time, use a definition load file.

If a definition load file is specified from the command line (with the **`-deffile`** option) along with other event definition and/or event handler record parameters (for example, **`-event_id`** and **`-handler_id`**), the UEMLoad request will fail.

In this situation, either:

- Add the event definition and/or event handler parameters to the load file.
- Remove the **`-deffile`** option from the request.

If no record parameters are specified from a definition load file or from the command line, UEMLoad assumes a definition load file is provided via standard input (that is, **`stdin`**) redirection.

If UEMLoad detects no input at all – either from the command line or a definition load file stored locally or redirected from **`stdin`** – UEMLoad assumes that input is being supplied from **`stdin`**, and remains active until it receives an end-of-file indicator.

The UEMLoad utility displays the following message to indicate that it is waiting for input:

```
UNV3678I Reading input from stdin. Enter Ctrl+Z <Enter> to cancel
wait…
```

Cancel the wait by supplying the end-of-file indicator: press <Ctrl+Z> <Enter>.

If a definition load file is provided to UEMLoad via **`stdin`** redirection, then no prompt is displayed. UEMLoad will read and process the redirected input just as it would a local definition load file.

14.2.2 Configuration Options

This section describes the configuration options used to execute UEMLoad for Windows.

Configuration Options Categories

Table 14.1, below, categorizes the configuration options according to function.

Category	Description
Event Definition	Parameters that correspond to fields in an event definition database record.
Event Handler	Parameters that correspond to fields in an event handler database record.
General	Affect the overall behavior of the UEMLoad utility.

Table 14.1 UEMLoad for Windows - Configuration Options Categories

The UEMLoad configuration options for each category are summarized in the following tables.

Each **Option Name** is a link to detailed information about that option.

Event Definition Options

Option Name	Description
ACTIVE_DATE_TIME	Date and time at which UEM will begin monitoring an event definition.
ASSIGNED_COMPONENT_NAME	Event-driven UEM Server responsible for monitoring the event.
EVENT_ID	Identifier that uniquely identifies an event definition record.
EVENT_STATE	Event definitions that should be processed or ignored by UEM.
EVENT_TYPE	Type of system event represented by the event definition record.
EXPIRED_HANDLER_ID	ID of an event handler record that UEM will execute when an event expires.
HANDLER_OPTIONS	String that will be passed as command line arguments to the event handler executed by UEM.
INACTIVE_DATE_TIME	Date and time at which UEM will stop monitoring an event definition.
REJECTED_HANDLER	ID of an event handler record that UEM will execute when an event occurrence is rejected.
TRACKING_INTERVAL	Frequency with which UEM will test for the completion of an event occurrence.
TRIGGERED_HANDLER_ID	ID of an event handler record that UEM will execute when an event occurrence is triggered.

Event Definition Options - Type-Specific

These options are specific to event definitions with an EVENT_TYPE of `FILE`.

Option Name	Description
FILE_SPECIFICATION	Name of a file to monitor.
MINIMUM_FILE_SIZE	Size a file must be in order to be considered complete by UEM.
RENAME_FILE	Specifies whether or not UEM should rename a monitored file when an event occurrence is triggered.
RENAME_FILE_SPECIFICATION	Specifies how a file should be renamed when an event occurrence is triggered.

Event Handler Options

Option Name	Description
ENCRYPTION_KEY	Key that was used with Universal Encrypt to encrypt the encrypted user file.
HANDLER_ID	Identifier that uniquely identifies an event handler record.
HANDLER_TYPE	Type of process executed for the event handler, based on the contents of the USER_COMMAND and USER_SCRIPT parameters
MAXIMUM_RETURN_CODE	Highest value with which a handler can exit to still be considered as having executed successfully.
USER_COMMAND	Command to execute on behalf of the event handler.
USER_FILE_ENCRYPTED	Complete path to a file encrypted with Universal Encrypt. If this file contains a user ID and password, the values for each are stored in the USER_ID and USER_PASSWORD fields, respectively. A UEM Server will re-read this file as it prepares an event handler process for execution, in order to obtain any changes to the user ID and/or password values contained in the file.
USER_FILE_PLAIN	Name of a text file that contains user ID and/or password information.
USER_ID	ID of a user account in whose security context the handler process will be executed.
USER_PASSWORD	Password for the user account specified by the USER_ID parameter.
USER_SCRIPT	Text file that contains statements that are executed collectively as a script by UEM.
USER_SCRIPT_TYPE	Type of script statements contained in the action field of the event handler record.
OPTIONS	String value that is added to the command line UEM Server builds in order to execute an event handler process. The event definition's HANDLER_OPTIONS field is used for the same purpose, but is added to the command string after this field, in order to customize process behavior for that event.

General Command Options

Option Name	Description
ACTION	Requested database operation.
BROKER_PORT	Port on which a local Universal Broker is accepting incoming connections.
COMMAND_ID	Unique command ID associated with the database request.
DEFINITION_FILE	Name of a file that contains event definition and/or event handler parameters.
HELP	Displays command line help.
MESSAGE_LEVEL	Sets the level of messages reported by UEMLoad.
VERSION	Displays version information.

14.2.3 Command Line Syntax

Figure 14.1, below, illustrates the command line syntax — using the command line, long form of the configuration options — of UEMLoad for Windows.

```
uemload
-add
{ [-deffile filename] | [eventdefopts [handlerdefopts]] | [handlerdefopts
[eventdefopts]] }
[-cmdid id]
[-level msglevel]
[-port port]

where eventdefopts:
-event_id id
-event_type type
type-specificopts
[-comp_name compname]
[-state state]
[-act_date_time yyyy.mm.dd,hh:mm]
[-inact_date_time yyyy.mm.dd,hh:mm]
[-tracking_int seconds]
[-triggered_id handlerid]
[-rejected_id handlerid]
[-expired_id handlerid]
[-handler_opts options]

where type-specificopts:
{ -event_type FILE -filespec filename [-min_file_size size] [-rename_file
option] [-rename_filespec renamefile] }

where handleropts:
-handler_id id
{ -cmd command | -script filename [-script_type scripttype] }
[ {-file filename | -encryptedfile filename [-key key] | -userid uid
   [-pwd password] } ]
[-handler_type type]
[-maxrc returncode]
[-options cmdlineopts]
```

```
uemload
-delete
[-cmdid id]
[-level msglevel]
[-port port]
{ -deffile filename | -event_id id [-handler_id id ] | -handler_id id
   [-event_id id] }

uemload
-update
{ [-deffile filename] | [eventdefopts [handlerdefopts]] | [handlerdefopts
[eventdefopts]] }
[-cmdid id]
[-level msglevel]
[-port port]

where eventdefopts:
-event_id id
[-event_type type]
[type-specificopts]
[-comp_name compname]
[-state state]
[-act_date_time yyyy.mm.dd,hh:mm]
[-inact_date_time yyyy.mm.dd,hh:mm]
[-tracking_int seconds]
[-triggered_id handlerid]
[-rejected_id handlerid]
[-expired_id handlerid]
[-handler_opts options]

where type-specificopts:
{ -event_type FILE [-filespec filename] [-min_file_size size] [-rename_file
option] [-rename_filespec renamefile] }

where handleropts:
-handler_id id
{ -cmd command | -script filename [-script_type scripttype] }
[ {-file filename | -encryptedfile filename [-key key] | -userid uid
   [-pwd password] } ]
[-handler_type type]
[-maxrc returncode]
[-options cmdlineopts]
```

```
uemload
-list
[-event_id id]
[-comp_name compname]
[-handler_id id]
[-cmdid id]
[-level msglevel]
[-port port]

uemload
-export
[-deffile filename]
[-event_id id]
[-comp_name compname]
[-handler_id id]

uemload
{ -help | -version }
```

Figure 14.1 UEMLoad for Windows - Command Line Syntax

CHAPTER 15

UEMLoad Utility for UNIX

15.1 Overview

This chapter provides information on Universal Event Monitor (UEM) Load utility, specific to the UNIX operating system.

Note: Because the UEM Server component is the only UEM component that uses the databases managed by UEMLoad, and because the UEM Server is available only on UNIX and Windows, UEMLoad also is available only on those platforms. The command syntax and general usage of UEMLoad is the same on both platforms, except where noted.

UEMLoad manages records in the event definition and event handler databases. The records in these databases then are used as input to a local UEM Server.

UEMLoad is capable of processing multiple event definition and/or event handler records when the parameters for those records are supplied using a text load file. If no load file is specified, the parameters for a single event definition and/or event handler record can be specified from the command line.

Stoneman's Tip

Although UEMLoad can access only local event definition and event handler databases, it is possible to store definition load files in a single location (for example, a PDS on z/OS) and distribute them to remote systems via Universal Command.

Simply redirect the definition load file from stdin and have Universal Command execute UEMLoad on the remote system. UEMLoad will read the redirected input and process it just as it would a local definition load file. This simplifies central administration of remote databases, because definition load files do not have to be stored (and managed) across several systems.

15.2 Usage

The UEMLoad utility executes as a command line application.

The syntax for invoking UEMLoad from the command line requires the name of the program, **`uemload`**, followed by a list of configuration options.

This section describes the configuration, configuration options, and command line syntax of UEMLoad for Windows.

15.2.1 Configuration

Configuration consists of:

- Setting default options and preferences for all executions of UEMLoad.
- Setting options and preferences for a single execution of UEMLoad.

UEMLoad for UNIX receives its configuration options from the following sources:

1. Command line
2. Environment variables

The order of precedence is the same as the list above; command line options being the highest and environment variables being the lowest. That is, options specified via a command line override options specified via environment variables.

Definition and Event Handler Parameters

The configuration options source list, above, applies only to those general command options used to control execution of UEMLoad.

For event definition and/or event handler parameters, input can come from one of three mutually exclusive sources:

1. Command line
2. Definition load file
3. Definition load file redirected from **`stdin`**

Definition Load File

When record parameters are provided from the command line, only one event definition and/or event handler can be specified. To add, update, or delete multiple event definition and/or event handler records at the same time, use a definition load file.

If a definition load file is specified from the command line (with the **`-deffile`** option) along with other event definition and/or event handler record parameters (for example, **`-event_id`** and **`-handler_id`**), the UEMLoad request will fail.

In this situation, either:

- Add the event definition and/or event handler parameters to the load file.
- Remove the **`-deffile`** option from the request.

If no record parameters are specified from a definition load file or from the command line, UEMLoad assumes a definition load file is provided via standard input (that is, **`stdin`**) redirection.

If UEMLoad detects no input at all — either from the command line or a definition load file stored locally or redirected from **`stdin`** — UEMLoad assumes that input is being supplied from **`stdin`**, and remains active until it receives an end-of-file indicator.

The UEMLoad utility displays the following message to indicate that it is waiting for input:

```
UNV3678I Reading input from stdin. Enter Ctrl+D to cancel wait…
```

Cancel the wait by supplying the end-of-file indicator: press <Ctrl+D>.

If a definition load file is provided to UEMLoad via **`stdin`** redirection, then no prompt is displayed. UEMLoad will read and process the redirected input just as it would a local definition load file.

15.2.2 Configuration Options

This section describes the configuration options used to execute UEMLoad for UNIX.

Configuration Options Categories

Table 15.1, below, categorizes the configuration options according to function.

Category	Description
Event Definition	Parameters that correspond to fields in an event definition database record.
Event Handler	Parameters that correspond to fields in an event handler database record.
General	Affect the overall behavior of the UEMLoad utility.

Table 15.1 UEMLoad for UNIX - Configuration Options Categories

The UEMLoad configuration options for each category are summarized in the following tables.

Each **Option Name** is a link to detailed information about that option.

Event Definition Options

Option Name	Description
ACTIVE_DATE_TIME	Date and time at which UEM will begin monitoring an event definition.
ASSIGNED_COMPONENT_NAME	Event-driven UEM Server responsible for monitoring the event.
EVENT_ID	Identifier that uniquely identifies an event definition record.
EVENT_STATE	Event definitions that should be processed or ignored by UEM.
EVENT_TYPE	Type of system event represented by the event definition record.
EXPIRED_HANDLER_ID	ID of an event handler record that UEM will execute when an event expires.
HANDLER_OPTIONS	String that will be passed as command line arguments to the event handler executed by UEM.
INACTIVE_DATE_TIME	Date and time at which UEM will stop monitoring an event definition.
REJECTED_HANDLER	ID of an event handler record that UEM will execute when an event occurrence is rejected.
TRACKING_INTERVAL	Frequency with which UEM will test for the completion of an event occurrence.
TRIGGERED_HANDLER_ID	ID of an event handler record that UEM will execute when an event occurrence is triggered.

Event Definition Options - Type-Specific

These options are specific to event definitions with an EVENT_TYPE of **`FILE`**.

Option Name	Description
FILE_SPECIFICATION	Name of a file to monitor.
MINIMUM_FILE_SIZE	Size a file must be in order to be considered complete by UEM.
RENAME_FILE	Specifies whether or not UEM should rename a monitored file when an event occurrence is triggered.
RENAME_FILE_SPECIFICATION	Specifies how a file should be renamed when an event occurrence is triggered.

Event Handler Options

Option Name	Description
ENCRYPTION_KEY	Key that was used with Universal Encrypt to encrypt the encrypted user file.
HANDLER_ID	Identifier that uniquely identifies an event handler record.
HANDLER_TYPE	Type of process executed for the event handler, based on the contents of the USER_COMMAND and USER_SCRIPT parameters
MAXIMUM_RETURN_CODE	Highest value with which a handler may exit to still be considered as having executed successfully.
USER_COMMAND	Command to execute on behalf of the event handler.
USER_FILE_ENCRYPTED	Complete path to a file encrypted with Universal Encrypt. If this file contains a user ID and password, the values for each are stored in the USER_ID and USER_PASSWORD fields, respectively. A UEM Server will re-read this file as it prepares an event handler process for execution, in order to obtain any changes to the user ID and/or password values contained in the file.
USER_FILE_PLAIN	Name of a text file that contains user ID and/or password information.
USER_ID	ID of a user account in whose security context the handler process will be executed.
USER_PASSWORD	Password for the user account specified by the USER_ID parameter.
USER_SCRIPT	Text file that contains statements that are executed collectively as a script by UEM.
USER_SCRIPT_TYPE	Type of script statements contained in the action field of the event handler record.
OPTIONS	String value added to the command line that the UEM Server builds in order to execute an event handler process. The event definition's HANDLER_OPTIONS field is used for the same purpose, but is added to the command string after this field, in order to customize process behavior for that event.

General Options

Option Name	Description
ACTION	Requested database operation.
BROKER_PORT	Port on which a local Universal Broker is accepting incoming connections.
COMMAND_ID	Unique command ID associated with the database request.
DEFINITION_FILE	Name of a file that contains event definition and/or event handler parameters.
HELP	Displays command line help.
MESSAGE_LEVEL	Sets the level of messages reported by UEMLoad.
VERSION	Displays version information.

15.2.3 Command Line Syntax

Figure 15.1, below, illustrates the syntax of the UEMLoad utility for UNIX.

```
uemload
-add
{ [-deffile filename] | [eventdefopts [handlerdefopts]] | [handlerdefopts
[eventdefopts]] }
[-cmdid id]
[-level msglevel]
[-port port]

where eventdefopts:
-event_id id
-event_type type
type-specificopts
[-comp_name compname]
[-state state]
[-act_date_time yyyy.mm.dd,hh:mm]
[-inact_date_time yyyy.mm.dd,hh:mm]
[-tracking_int seconds]
[-triggered_id handlerid]
[-rejected_id handlerid]
[-expired_id handlerid]
[-handler_opts options]

where type-specificopts:
{ -event_type FILE -filespec filename [-min_file_size size]
  [-rename_file option] [-rename_filespec renamefile] }

where handleropts:
-handler_id id
{ -cmd command | -script filename [-script_type scripttype] }
[ {-file filename | -encryptedfile filename [-key key] | -userid uid
   [-pwd password] } ]
[-handler_type type]
[-maxrc returncode]
[-options cmdlineopts]
```

```
uemload
-delete
[-cmdid id]
[-level msglevel]
[-port port]
{ -deffile filename | -event_id id [-handler_id id ] | -handler_id id
   [-event_id id] }

uemload
-update
{ [-deffile filename] | [eventdefopts [handlerdefopts]] | [handlerdefopts
[eventdefopts]] }
[-cmdid id]
[-level msglevel]
[-port port]

where eventdefopts:
-event_id id
[-event_type type]
[type-specificopts]
[-comp_name compname]
[-state state]
[-act_date_time yyyy.mm.dd,hh:mm]
[-inact_date_time yyyy.mm.dd,hh:mm]
[-tracking_int seconds]
[-triggered_id handlerid]
[-rejected_id handlerid]
[-expired_id handlerid]
[-handler_opts options]

where type-specificopts:
{ -event_type FILE [-filespec filename] [-min_file_size size]
  [-rename_file option] [-rename_filespec renamefile] }

where handleropts:
-handler_id id
{ -cmd command | -script filename [-script_type scripttype] }
[ {-file filename | -encryptedfile filename [-key key] | -userid uid
   [-pwd password] } ]
[-handler_type type]
[-maxrc returncode]
[-options cmdlineopts]
```

```
uemload
-list
[-event_id id]
[-comp_name compname]
[-handler_id id]
[-cmdid id]
[-level msglevel]
[-port port]

uemload
-export
[-deffile filename]
[-event_id id]
[-comp_name compname]
[-handler_id id]

uemload
{ -help | -version }
```

Figure 15.1 UEMLoad Utility for UNIX - Command Line Syntax

CHAPTER 16

UEMLoad Utility Configuration Options

16.1 Overview

This chapter provides detailed information on the configuration options available for use with the Universal Event Monitor Load Utility.

The options are listed alphabetically, without regard to any specific operating system.

Section 16.2 Configuration Options Information provides a guideline for understanding the information presented or each option.

16.2 Configuration Options Information

For each configuration option, this chapter provides the following information.

Description

Describes the option and how it is used.

Usage

Provides a table of the following information:

Specification Method	Parameter / Value	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	<Format / Value>					
Command Line, Long Form	<Format / Value>					
Definition File Keyword	<Format / Value>					

Specification Method

List of all possible methods for specifying an option:

- Command Line, Short Form
- Command Line, Long Form
- Definition File Keyword

Parameter / Value

Syntax for specifying the option:

• Parameter	Parameter syntax for the corresponding Specification Method.
• Value	Specify alphanumeric value for that parameter

(Operating System)

Identifies (with a ✓) the operating systems for which each method of specifying the option is valid:

- IBM i
- NonStop (HP NonStop)
- UNIX
- Windows
- z/OS

Values

Identifies all possible values for the option.

Default

Identifies default information regarding the option or its values.

Defaults for option values are identified in **[bracketed bold type]**.

<Additional Information>

Identifies any additional information specific to that option.

16.3 Configuration Options List

Table 16.1 UEMLoad Utility - Configuration Options identifies all UEMLoad Utility configuration options.

Option	Description	Page
ACTION	Requested database operation.	247
ACTIVE_DATE_TIME	Date and time at which UEM will begin monitoring an event definition.	249
ASSIGNED_COMPONENT_NAME	Event-driven UEM Server responsible for monitoring the event.	250
BROKER_PORT	Port on which a local Universal Broker is accepting incoming connections.	252
COMMAND_ID	Unique command ID associated with the database request.	253
DEFINITION_ FILE	Name of a file that contains event definition and/or event handler parameters.	254
ENCRYPTION_KEY	Key that was used with Universal Encrypt to encrypt the encrypted user file.	255
EVENT_ID	Identifier that uniquely identifies an event definition record.	256
EVENT_STATE	Event definitions that should be processed or ignored by UEM.	257
EVENT_TYPE	Type of system event represented by the event definition record.	258
EXPIRED_HANDLER_ID	ID of an event handler record that UEM will execute when an event expires.	259
FILE_SPECIFICATION	Name of a file to monitor.	260
HANDLER_ID	Identifier that uniquely identifies an event handler record.	261
HANDLER_OPTIONS	String that will be passed as command line arguments to the event handler executed by UEM.	263
HANDLER_TYPE	Type of process executed for the event handler, based on the contents of the USER_COMMAND and USER_SCRIPT parameters	265
HELP	Displays command line help.	266
INACTIVE_DATE_TIME	Date and time at which UEM will stop monitoring an event definition.	267
MAXIMUM_RETURN_CODE	Highest value with which a handler may exit to still be considered as having executed successfully.	269
MESSAGE_LEVEL	Sets the level of messages reported by UEMLoad.	270
MINIMUM_FILE_SIZE	Size a file must be in order to be considered complete by UEM.	271
OPTIONS	String value that is added to the command line UEM Server builds in order to execute an event handler process. The event definition's HANDLER_OPTIONS field is used for the same purpose, but is added to the command string after this field, in order to customize process behavior for that event.	272

Option	Description	Page
REJECTED_HANDLER_ID	ID of an event handler record that UEM will execute when an event occurrence is rejected.	274
RENAME_FILE	Specifies whether or not UEM should rename a monitored file when an event occurrence is triggered.	275
RENAME_FILE_SPECIFICATION	Specifies how a file should be renamed when an event occurrence is triggered.	276
TRACKING_INTERVAL	Frequency with which UEM will test for the completion of an event occurrence.	278
TRIGGERED_HANDLER_ID	ID of an event handler record that UEM will execute when an event occurrence is triggered.	279
USER_COMMAND	Command to execute on behalf of the event handler.	280
USER_FILE_ENCRYPTED	Complete path to a file encrypted with Universal Encrypt. If this file contains a user ID and password, the values for each are stored in the USER_ID and USER_PASSWORD fields, respectively. A UEM Server will re-read this file as it prepares an event handler process for execution, in order to obtain any changes to the user ID and/or password values contained in the file.	281
USER_FILE_PLAIN	Name of a text file that contains user ID and/or password information.	283
USER_ID	ID of a user account in whose security context the handler process will be executed.	285
USER_PASSWORD	Password for the user account specified by the USER_ID parameter.	286
USER_SCRIPT	Text file that contains statements that are executed collectively as a script by UEM.	287
USER_SCRIPT_TYPE	Type of script statements contained in the action field of the event handler record.	288
VERSION	Displays version information.	289

Table 16.1 UEMLoad Utility - Configuration Options

16.4 ACTION

Description

The ACTION option specifies the requested database operation. It is a required option.

Only one database operation can be specified for each invocation of the UEMLoad utility. The database operations supported by the UEMLoad utility are listed below, in Values.

Usage

Specification Method	Parameter / Value	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-*action* {add | update | delete | list | export}			√	√	
Definition File Keyword	n/a					

Values

There are no values for this option. Instead, the desired action is specified by the parameter:

- **add**
  Writes one or more new event definition and/or event handler records to the appropriate database.
- **update**
  Changes one or more existing event definition and/or event handler records.
- **delete**
  Removes the specified event definition and/or event handler records from the appropriate database.
- **list**
  Displays the complete contents of the specified event definition and/or event handler records.
- **export**
  Dumps the contents of the specified event definition and/or event handler records to a text file that can be used as input to a subsequent run of the UEMLoad utility.

add

If **add** is specified, any optional event definition or event handler parameters not specified will be provided with default values as specified in the UEM Server configuration. If an attempt is made to add a record that already exists, the operation will fail.

update

If **update** is specified, only those parameters being changed are required. Values for other parameters are obtained from the existing event definition or event handler record. If an attempt is made to update a record that does not exist, the operation will fail.

list

If **list** is specified, the only other recognized configuration options are:

- EVENT_ID
- ASSIGNED_COMPONENT_NAME
- HANDLER_ID

export

If **export** is specified, the only other recognized configuration options are:

- DEFINITION_FILE
- EVENT_ID
- ASSIGNED_COMPONENT_NAME
- HANDLER_ID

Stoneman's Tip

To list or export all event definition and event handler records, no configuration options are required; simply specify the *list* or *export* action, respectively.

To export all records to a text file, simply specify a definition load file using the DEFINITION_FILE option.

In both cases, if no event ID, assigned component name, or handler ID is specified, all records in the event definition and event handler databases will be retrieved.

16.5 ACTIVE_DATE_TIME

Description

The ACTIVE_DATE_TIME option specifies the date and time when UEM Server will begin checking for the occurrence of the system event represented by the event definition.

If ACTIVE_DATE_TIME option is not used, default values are set by the UEM Server. When an event definition is added, the date portion of the date and time value is set to the current date. If no time value is provided, the current time is used.

When an event definition is updated and a value for either the date or the time is omitted, the existing value is preserved.

ACTIVE_DATE_TIME is ignored for all other requests.

Usage

Specification Method	Parameter / Value	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-act_date_time *yyyy.mm.dd,hh:mm*			√	√	
Definition File Keyword	act_date_time *yyyy.mm.dd,hh:mm*			√	√	

Values

yyyy.mm.dd,hh:mm is the date and time when the UEM Server will begun checking.

Either the date or the time can be omitted.

- If the date is omitted, the comma separator must be provided to serve as a placeholder (for example: *,hh:mm*).
- If the time is omitted, the comma is not required.

Note: The maximum valid value for date and time is **`2038.01.16,23:59`**.

16.6 ASSIGNED_COMPONENT_NAME

Description

The ASSIGNED_COMPONENT_OPTION specifies the name of an event-driven UEM Server component that is responsible for monitoring the event represented by the event definition record.

To ensure proper assignment, this name should match the name of an installed UEM Server component.

Windows

A component's name matches the name of its component definition file. A component definition file is a text file that contains parameters used by the Universal Broker when it starts a component.

Component definition files are installed in the **`%ALLUSERSPROFILE%\Application Data\Universal\comp`** directory, where **`%ALLUSERSPROFILE%`** is an environment variable that expands to the All Users folder; typically **`C:\Documents`** and **`Settings\All Users`**.

A list of installed components can be viewed from the Universal Configuration Manager by selecting the Component Definitions page of the Universal Broker configuration control.

UNIX

A component's name matches the name of its component definition file. A component definition file is a text file that contains parameters used by the Universal Broker when it starts a component.

Component definition files are provided by the Stonebranch Solutions installation, and are installed in the **`etc/universal/comp`** directory.

When an event-driven UEM Server starts, it obtains all event definition records that have been assigned to it, based on the value specified by this option. An event-driven UEM Server is any UEM Server component whose component definition has the value **`uems`** specified for its **`component_type`** parameter.

If a demand-driven UEM Server is asked to monitor an event using a stored event definition record (that is, a UEM Manager was started using the EVENT_ID option), the value stored in this field is ignored.

By default, new event definition records are assigned to the UEM Server component defined by the **`uems`** component definition.

Stoneman's Tip

The assignment of an event definition to a component is provided primarily for future support of concurrent event-driven UEM Servers.

As of the current release, support for only a single instance of an event-driven Server is provided. Therefore, for any event definitions that you want to be monitored by an event-driven Server, it is strongly recommended that you use the default assignment of **uems**.

To store event definitions that do not need to be monitored by an event-driven Server, you can assign them to any component that does not match **uems**.

For example, to create an event definition that is intended to only be referenced by a UEM Manager, use a value of **uemd** for the component name parameter.

Usage

Specification Method	Parameter / Value	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-comp_name *compname*			√	√	
Definition File Keyword	comp_name *compname*			√	√	

Value

compname is the name of the event-driven UEM Server.

[Default is *uems*.]

16.7 BROKER_PORT

Description

The BROKER_PORT option specifies the TCP port upon which a local Universal Broker is accepting incoming connections.

The UEMLoad utility establishes a connection to the Broker over this port to request the startup of a UEM Server component that will handle processing of the UEMLoad request.

Usage

Specification Method	Parameter / Value	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	-p *port*			√	√	
Command Line, Long Form	-port *port*			√	√	
Definition File Keyword	n/a					

Values

port is the TCP port upon which the local Universal Broker is accepting connections.

The format of *port* is either:

- Number (for example, *7887*)
- Service name (for example, *ubroker*)

[Default is *7887*.]

16.8 COMMAND_ID

Description

The COMMAND_ID option specifies an identifier that is saved by the Universal Broker and which is used to identify the unit of work represented by an instance of the UEMLoad utility and its associated UEM Server component.

Usage

Specification Method	Parameter / Value	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	-C *id*			√	√	
Command Line, Long Form	-cmdid *id*			√	√	
Definition File Keyword	n/a					

Value

id is any value.

If *id* contains spaces, it must be enclosed in double (**"**) quotation marks.

UNIX

If *id* contains spaces, it must be enclosed in either single (**'**) or double (**"**) quotation marks.

Defaults

If the ACTION option is set to *add*, *update*, or *delete*, the default *id* is *UEMLoad – Database Update*.

If the ACTION option is set to *list*, the default *id* is *UEMLoad – Database List*.

For database *export*s, the default *id* is *UEMLoad – Database Export*.

16.9 DEFINITION_FILE

Description

The DEFINITION_FILE option specifies a file that is either read from or written to, depending on the value of the ACTION option:

- If ACTION specifies *–add*, *-update*, or *–delete*, DEFINITION_FILE specifies the path to a text file that contains the parameters for one or more event definition and/or event handler records.
- If ACTION specifies *-export*, DEFINITION_FILE specifies the path to a text file to which the contents of the event definition and event handler records (requested using the EVENT_ID and HANDLER_ID options, respectively) are written.

Note: A database export is the only situation in which DEFINITION_FILE and the EVENT_ID and/or HANDLER_ID options can be specified together. DEFINITION_FILE is ignored if ACTION specifies *-list*.

When parameters for more than one event definition or event handler record are specified in a definition file, certain keywords are required to mark the beginning and ending of each record. These keywords, along with other rules specific to definition files, are described in the Definition File Format sections (Chapters 14 UEMLoad Utility for Windows and 15 UEMLoad Utility for UNIX).

Usage

Specification Method	Parameter / Value	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-deffile *filename*			√	√	
Definition File Keyword	n/a					

Value

filename is the file to be written to or read from.

Valid values contain either:

- Complete path
- Path that is relative to the directory from which the UEMLoad utility is invoked

16.10 ENCRYPTION_KEY

Description

The ENCRYPTION_KEY option specifies the key that is used to decrypt the file specified by the USER_FILE_ENCRYPTED option.

This key is required only if a key was used to encrypt the user file with Universal Encrypt. (For complete information on the Universal Encrypt utility, see the Stonebranch Solutions Utilities 4.2.0 Reference Guide.)

If this option is not used, a default key established by the UEMLoad utility is used.

Usage

Specification Method	Parameter / Value	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	-K *key*			√	√	
Command Line, Long Form	-key *key*			√	√	
Definition File Keyword	key *key*			√	√	

Value

key is the key used to decrypt the data set / file.

16.11 EVENT_ID

Description

The EVENT_ID option specifies a value that uniquely identifies an event definition record.

If an attempt is made to *add* an event definition record with an event ID that matches this value, the request will fail. All other operations will fail if the value does NOT match the ID of an existing event definition record.

Usage

Specification Method	Parameter / Value	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-event_id *id*			√	√	
Definition File Keyword	event_id *id*			√	√	

Value

id is the ID of an event definition record.

The length of *id* must not exceed 32 characters.

id is case-insensitive. When an event definition is added, *id* is stored within the record exactly as specified. However, when that record later is referenced by event ID, case is ignored. For example, if a record is added with an *id* of Event001, the mixed case of the ID is preserved within the record. To access the record after it's been added, an event ID of any case (for example: event001, eVENt001, or EvenT001) can be specified.

Wildcards

If a *-list*, *-export*, or *-delete* operation is being used for UEMLoad, wildcards can be used in *id* to select multiple event definition records:

- Asterisk (*****) will match 0 or more characters.
- Question mark (**?**) can be used to match any single character.

16.12 EVENT_STATE

Description

The EVENT_STATE option specifies whether or not an event is processed by the event-driven UEM Server to which it is assigned (via the event definition record's ASSIGNED_COMPONENT_NAME field).

Usage

Specification Method	Parameter / Value	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-state *eventstate*			√	√	
Definition File Keyword	state *eventstate*			√	√	

Value

eventstate is the specification for whether or not the event is processed.

Valid values for *eventstate* are:

- **enable**
  Event definition is processed by UEM.
- **disable**
  Event definition is ignored by UEM.

An event definition whose *eventstate* is set to **disable** still is added to the list of assigned event definitions obtained by an event-driven UEM Server during startup processing. However, the UEM Server will not process that event until:

1. *eventstate* is set to **enable**.
2. UEM Server refreshes its list of assigned event definitions.

[Default is enable.]

16.13 EVENT_TYPE

Description

The EVENT_TYPE option specifies the type of system event represented by the event definition record.

EVENT_TYPE is required for new event definition records.

For updates to existing event definition records, EVENT_TYPE is required only if a change is being made to one of the event definition options that are specific to event types (see Table 16.2, below).

EVENT_TYPE is ignored for all other requests.

Usage

Specification Method	Parameter / Value	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-event_type *type*			√	√	
Definition File Keyword	event_type *type*			√	√	

Value

type is the event type represented by the event definition record.

It must match one of the known event types supported by UEM (Table 16.2).

Note: Currently, there is only one valid UEM event type: FILE.

Event Type	Description	Associated Options
FILE	Detects the creation of a file and monitors it, testing it for completeness.	• FILE_SPECIFICATION * • MINIMUM_FILE_SIZE • RENAME_FILE • RENAME_FILE_SPECIFICATION

Table 16.2 UEM Event Types

16.14 EXPIRED_HANDLER_ID

Description

The EXPIRED_HANDLER_ID option specifies the event handler that UEM Server will execute for the event whenever it is set to an `expired` state.

To remove the assignment of an event handler for a particular event definition:

1. Use the *-update* operation (see ACTION).
2. Specify a value of *none* for this option.

If EXPIRED_HANDLER_ID is not used when an event definition record is added, the following default value is set by the UEM Server:

• Windows	Empty string
• UNIX	*NONE*

Usage

Specification Method	Parameter / Value	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-expired_id *handlerid*			√	√	
Definition File Keyword	expired_id *handlerid*			√	√	

Value

handlerid is the ID of the event handler that UEM will execute.

It must match the ID of an existing record in the event handler database. Execution of the event handler will fail if the UEM Server cannot find the specified handler record.

If *handlerid* contains spaces, it must be enclosed in double (**"**) quotation marks.

UNIX

If *handlerid* contains spaces, it must be enclosed in either single (**'**) or double (**"**) quotation marks.

16.15 FILE_SPECIFICATION

Description

The FILE_SPECIFICATION option specifies the name of a file whose presence should be detected and monitored by UEM Server.

An absolute path or a path relative to the UEM Server working directory, as defined in the component definition, also can be specified by this option.

Note: This option is valid only for events with an EVENT_TYPE of **FILE**.

Usage

Specification Method	Parameter / Value	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-filespec *filename*			√	√	
Definition File Keyword	filespec *filename*			√	√	

Values

filename is the name of the file to be monitored by UEM.

If *filename* contains spaces, it must be enclosed in double (**"**) quotation marks.

UNIX

If *filename* contains spaces, it must be enclosed in either single (**'**) or double (**"**) quotation marks.

Wildcards

The following wildcards can be specified as part of *filename*.

- ***** Match zero, one or more characters.
- **?** Match zero or one character.

16.16 HANDLER_ID

Description

The HANDLER_ID option specifies a value that uniquely identifies an event handler record.

If an attempt is made to *add* an event handler record with a HANDLER_ID value that matches the ID of an existing event handler record, the request will fail. All other operations will fail if the HANDLER_ID value does NOT match the ID of an existing event handler record.

Usage

Specification Method	Parameter / Value	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-handler_id *id*			√	√	
Definition File Keyword	handler_id *id*			√	√	

Value

id is the ID of an event handler record.

The length of *id* must not exceed 32 characters.

id is case-insensitive. When an event handler is added, *id* is stored within the record exactly as specified. However, when that record later is referenced by handler ID, case is ignored. For example, if a record is added with a handler ID of Handler001, the mixed case of the ID is preserved within the record. To access the record after it's been added, a handler ID of any case (for example: handler001, hANDLEr001, or HANDler001) can be specified.

If *id* contains spaces, it must be enclosed in double (**"**) quotation marks.

UNIX

If *id* contains spaces, it must be enclosed in either single (**'**) or double (**"**) quotation marks.

Wildcards

If a *-list*, *-export*, or *-delete* operation is being used for UEMLoad, wildcards can be used in *id* to select multiple event handler records:

- Asterisk (*****) will match 0 or more characters.
- Question mark (**?**) can be used to match any single character.

16.17 HANDLER_OPTIONS

Description

The HANDLER_OPTIONS option specifies a value that is passed as a command line argument to the process that is executed on behalf of an event handler.

Usage

Specification Method	Parameter / Value	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-handler_opts *string*			√	√	
Definition File Keyword	handler_opts *string*			√	√	

Value

string is a character string that is appended to the command line constructed by UEM in order to execute the event handler process.

A value of *none* can be used to remove the handler options string from an existing event definition record.

If *string* contains spaces, it must be enclosed in double (**"**) quotation marks.

UNIX

If *string* contains spaces, it must be enclosed in either single (**'**) or double (**"**) quotation marks.

Windows

If quotes are to be passed as part of the parameter's value (for example, an argument that contains spaces is to be passed to the handler process and treated as a single argument), the required syntax depends on whether the parameter is specified from the command line or from a definition file.

From the command line, use a backslash (\) to escape the quotes that need to be preserved (for example: *-handler_options "parm1 \\"parm2a parm2b\\" parm3"*). From a definition file, use an extra double quote to escape the quotes (for example, *handler_opts "parm1 ""parm2a parm2b"" parm3"*).

In both cases, three arguments will be passed to the event handler process. The portion of the string enclosed in double quotes (that is, *"parm2a parm2b"*) is treated as a single argument.

UNIX

If quotes are to be passed as part of the parameter's value (for example, an argument that contains spaces is to be passed to the handler process and treated as a single argument), enclose *options* in single quotes. Use a set of double quotes to enclose the quoted value.

For example, specifying *-handler_options 'parm1 "parm2a parm2b" parm3'* will cause three arguments to be passed to the process executed on behalf of the event handler. The portion of the string enclosed in double quotes (that is, *"parm2a parm2b"*) is treated as a single value.

The HANDLER_OPTIONS value is very similar to the OPTIONS value stored in an event handler record. UEM adds both as command line parameters to the command string that it builds to execute an event handler process. The difference is that the value stored in HANDLER_OPTIONS is used by every event handler referenced by an event definition.

For example, if an event definition contains a value for TRIGGERED_HANDLER_ID and EXPIRED_HANDLER_ID, the command line arguments stored in HANDLER_OPTIONS are passed to both handler processes.

On the other hand, the value stored in OPTIONS is used every time the event handler's process is executed. Keep in mind that a single event handler record can be referenced by many event definition records. Because of this one-to-many relationship, UEM adds the value stored in HANDLER_OPTIONS to the command string after the value stored in the event handler's OPTIONS. This effectively allows an event definition to change the behavior of an event handler by overriding command line parameters (assuming that the last duplicate parameter specified is the one used by the process).

Default

There is no default value for *options* for new event definition records.

16.18 HANDLER_TYPE

Description

The HANDLER_TYPE option provides UEM with information that it needs when building a command string in order to execute a process for the event handler.

Usage

Specification Method	Parameter / Value	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-handler_type *type*			√	√	
Definition File Keyword	handler_type *type*			√	√	

Values

type is the type of information provided to UEM.

Valid values for *type* are:

- **cmd**
  Instructs UEM to execute the command stored in the USER_COMMAND field of the event handler record.
- **script**
  Instructs UEM to write a set of stored script statements to a temporary file, and then execute that file as a script. These script statements can be provided instream from a definition load file using a `begin_/end_script` block. The statements also can be imported from a local file, using the USER_SCRIPT option.

Default

If a *type* is not specified, the default is:

- **cmd**, if a USER_COMMAND is specified.
- **script**, if a USER_SCRIPT is specified.

16.19 HELP

Description

The HELP option displays a description of the UEMLoad command options and their required format.

Usage

Specification Method	Parameter / Value	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	-h -?			√	√	
Command Line, Long Form	-help			√	√	
Definition File Keyword	n/a					

Value

(There are no values for the HELP option.)

16.20 INACTIVE_DATE_TIME

Description

The INACTIVE_DATE_TIME option specifies the date and time at which UEM Server will stop checking for the occurrence of the system event represented by the event definition.

If this specified date and time elapses, and no occurrence of the event has been detected, the event is set to an `expired` state.

This option also specifies the date and time by which a tracked event occurrence must complete. Any event occurrences being tracked when this date and time elapse are set to a `rejected` state.

Usage

Specification Method	Parameter / Value	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-inact_date_time *yyyy.mm.dd,hh:mm*			√	√	
Definition File Keyword	inact_date_time *yyyy.mm.dd,hh:mm*			√	√	

Values

yyyy.mm.dd,hh:mm is the date and time - separated by a comma (**,**) - when the UEM Server will stop checking for the occurrence of the system event.

Either the date or time can be omitted:

- If the date is omitted, the comma must be specified to serve as a placeholder (that is: *,hh:mm*).
- If the time is omitted, a comma is not required.

Default

If INACTIVE_DATE_TIME is not used, default values are set by the UEM Server.

- When an event definition is added, the date defaults to *2038.01.16*.
- If time only is omitted, a default of *23:59* is used.

When an event definition is updated, and a value for date and/or time is omitted, the existing value is preserved.

INACTIVE_DATE_TIME is ignored for all other requests.

16.21 MAXIMUM_RETURN_CODE

Description

The MAXIMUM_RETURN_CODE option specifies the highest value that a process executed on behalf of an event handler can return to be considered as having execute successfully.

UEM Server reports an execution error if the value returned by the handler process is greater than the value specified by MAXIMUM_RETURN_CODE.

Usage

Specification Method	Parameter / Value	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-maxrc *returncode*			√	√	
Definition File Keyword	maxrc *returncode*			√	√	

Value

returncode is the highest value that a process can return to be considered as having been successful.

If MAXIMUM_RETURN_CODE is not used, a default value of *0* (zero) is set by the UEM Server.

16.22 MESSAGE_LEVEL

Description

The MESSAGE_LEVEL option specifies the level of messages that will be issued.

Usage

Specification Method	Parameter / Value	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	-l *level*			√	√	
Command Line, Long Form	-level *level*			√	√	
Definition File Keyword	n/a					

Value

level is the level of messages to be issued.

Valid values for *level* are:

- **trace**
  Generates a text file that contains detailed program execution information.
  Note: **trace** is used for debugging purposes only. It should be used only when requested by Stonebranch Inc. Customer Support.
- **audit**
  Issues audit, informational, warning and error messages.
- **info**
  Issues informational, warning and error messages.
- **warn**
  Issues warning and error messages.
- **error**
  Issues error messages only.

[Default is info.]

16.23 MINIMUM_FILE_SIZE

Description

The MINIMUM_FILE_SIZE option specifies the smallest allowable size for a file in order for it to be considered complete by UEM Server.

Note: This option is valid only for files being monitored via EVENT_TYPE (event type = **`FILE`**).

Usage

Specification Method	Parameter / Value	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-min_file_size *size[units]*			√	√	
Definition File Keyword	min_file_size *size[units]*			√	√	

Value

size is the file size. *units* is the unit of storage for that *size*.

The valid values for *units*, and the maximum value that can be specified for *size* for that unit of storage, is:

- **b** bytes (4,294,967,295)
- **k** kilobytes (4,194,304)
- **m** megabytes (4,096)
- **g** gigabytes (4)

If a value for *units* is not specified, the file size is assumed to be in bytes.

If this option is not used, a default value of 0 (zero), as set by the UEM Server configuration, is used.

A value of 0 (zero) also can be used to indicate that the file can be any size.

16.24 OPTIONS

Description

The OPTIONS option specifies a value that is passed as a command line argument to the process executed for an event handler.

UEM adds this value to the command string that it builds as it prepares the event handler process for execution.

OPTIONS is very similar to the HANDLER_OPTIONS option stored in the event definition. UEM adds the parameters for both as command line parameters to the command string built in order to execute an event handler process. The difference is that the value stored in HANDLER_OPTIONS is used for every event handler referenced by an event definition. For example, if an event definition contains a value for the TRIGGERED_HANDLER_ID and EXPIRED_HANDLER_ID, then the command line arguments stored in HANDLER_OPTIONS are passed to both handler processes.

On the other hand, the value stored in OPTIONS is used every time that the event handler's process is executed. Keep in mind that a single event handler record can be referenced by many event definition records. Because of this one-to-many relationship, UEM adds the value stored HANDLER_OPTIONS to the command string after the value stored in the event handler's OPTIONS. This effectively allows an event definition to change the behavior of an event handler by overriding command line parameters (assuming that the last duplicate parameter specified is the one used by the process).

Usage

Specification Method	Parameter / Value	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-options *string*			√	√	
Definition File Keyword	options *string*			√	√	

Value

string is the value passed as a command line argument.

A value of *none* can be used to remove the options string from an existing event handler record.

If *string* contains spaces, it must be enclosed in double (**"**) quotation marks.

UNIX

If *string* contains spaces, it must be enclosed either in single (**'**) or double (**"**) quotation marks.

Windows

If quotes are to be passed as part of the parameter's value (for example, an argument that contains spaces is to be passed to the handler process and treated as a single argument), the required syntax depends on whether the parameter is specified from the command line or from a definition file.

From the command line, use a backslash (\) to escape the quotes that need to be preserved (for example, **`-options “parm1 \“parm2a parm2b\” parm3”`**).

From a definition file, use an extra double quote to escape the quotes (for example, **`options “parm1 ““parm2a parm2b”” parm3”`**).

In both cases, three arguments will be passed to the event handler process. The portion of the string enclosed in double quotes (that is, **`“parm2a parm2b”`**) is treated as a single argument.

UNIX

If quotes are to be passed as part of the parameter's value (for example, an argument that contains spaces is to be passed to the handler process and treated as a single argument), enclose *string* in single quotes, and use a set of double quotes to enclose the quoted value.

For example, specifying **`-options 'parm1 “parm2a parm2b” parm3'`** will cause three arguments to be passed to the event handler process. The portion of the string enclosed in double quotes (that is, **`“parm2a parm2b”`**) is treated as a single value.

Default

There is no default value for this option for new event handler records.

16.25 REJECTED_HANDLER

Description

The REJECTED_HANDLER option specifies the event handler that UEM Server will execute for a tracked event occurrence that it is set to a **`rejected`** state.

The specified event handler must match the ID of an existing record in the event handler database. Execution of the event handler will fail if the UEM Server cannot find the specified handler record.

Usage

Specification Method	Parameter / Value	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-rejected_id *handlerid*			√	√	
Definition File Keyword	rejected_id *handlerid*			√	√	

Value

handlerid is the event handler to be executed.

If *handlerid* contains spaces, it must be enclosed in double (**"**) quotation marks.

UNIX

If *handlerid* contains spaces, it must be enclosed either in single (**'**) or double (**"**) quotation marks.

To remove the assignment of an event handler for a particular event definition, use the *-update* command operation (see ACTION) and specify a value of *none* for *handlerid*.

Default

If REJECTED_HANDLER is not used when an event definition record is added, the following default values are used:

Windows	empty string
UNIX	*none*

16.26 RENAME_FILE

Description

The RENAME_FILE option specifies whether or not a file that is being monitored - as specified via an EVENT_TYPE of **`FILE`** - should be renamed by UEM when the event occurrence is set to a **`triggered`** state.

Renaming a file ensures that UEM will not treat a file that it just finished tracking as a new event occurrence.

Usage

Specification Method	Parameter / Value	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-rename_file *option*			√	√	
Definition File Keyword	rename_file *option*			√	√	

Value

option is the specification for whether or not a file should be renamed.

Valid values for *option* are:

- **yes** File is renamed according to the RENAME_FILE_SPECIFICATION option.
- **no** File is not renamed.

If *option* is set to **no**, in order to prevent multiple detection of the file by the UEM Server, rename the file in a script invoked by an event handler when the occurrence is triggered.

If RENAME_FILE is not used, a default value of **yes**, as set by the UEM Server configuration, is used.

16.27 RENAME_FILE_SPECIFICATION

Description

The RENAME_FILE_SPECIFICATION option specifies the file name that the UEM Server should use when both:

- An event occurrence for an event with an EVENT_TYPE of **`FILE`** is set to a **`triggered`** state.
- RENAME_FILE option is set to **yes**.

Usage

Specification Method	Parameter / Value	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-rename_filespec *renamefile*			√	√	
Definition File Keyword	rename_filespec *renamefile*			√	√	

Value

renamefile is the file name to be used by the UEM Server.

Valid values for *renamefile* can include an absolute path or a path that is relative to the location of the file monitored by UEM. If no path information is provided, the monitored file simply is renamed. The renamed file will reside in the same location as the original file.

If *renamefile* contains spaces, it must be enclosed in double (**"**) quotation marks.

UNIX

If *renamefile* contains spaces, it must be enclosed in either single (**'**) or double (**"**) quotation marks.

Variables

Variables that UEM will substitute with actual runtime values can be included as part of *renamefile*. These variables, and the values that UEM will substitute for them, are:

- **`$(compname)`**
  Component name
- **`$(compid)`**
  Component ID
- **`$(date)`**
  Current date, in the format *YYYYMMDD*
- **`$(time)`**
  Current time, in the format *HHMMSS*
- **`$(origname)`**
  Original base file name, minus its last extension
- **`$(origext)`**
  Original file extension
- **`$(seqnum)`**
  Sequence number that is initialized to 0 when the demand-driven UEM Server is started, and is then incremented by 1 for each file renamed.

If this option is not used, the following default value, set by the UEM Server configuration, is used:

```
$(compname).$(compid).$(date).$(seqnum)
```

16.28 TRACKING_INTERVAL

Description

The TRACKING_INTERVAL option specifies the frequency with which the UEM Server will test for the completion of a tracked event occurrence.

The test for completeness depends upon the type of event that the tracked occurrence represents. For example, an occurrence tracked for an event with an EVENT_TYPE of **`FILE`** is considered complete when two consecutive checks on a file's size return the same value.

Usage

Specification Method	Parameter / Value	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-tracking_int *seconds*			√	√	
Definition File Keyword	tracking_int *seconds*			√	√	

Value

seconds is the frequency (in seconds) with which the UEM Server tests for completion of a tracked occurrence.

If TRACKING_INTERVAL is not used, a default value of *10* seconds, set by the UEM Server configuration, is used.

16.29 TRIGGERED_HANDLER_ID

Description

The TRIGGERED_HANDLER_ID option specifies the event handler that the UEM Server will execute for a tracked event occurrence that is set to a **`triggered`** state.

This event handler must match the ID of an existing record in the event handler database. Execution of the event handler will fail if the UEM Server cannot find the specified handler record.

Usage

Specification Method	Parameter / Value	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-triggered_id *handlerid*			√	√	
Definition File Keyword	triggered_id *handlerid*			√	√	

Value

handlerid is the event handler to be executed.

If *handlerid* contains spaces, it must be enclosed in double (**"**) quotation marks.

UNIX

If *handlerid* contains spaces, it must be enclosed in either single (**'**) or double (**"**) quotation marks.

To remove the assignment of an event handler for a particular event definition, use the *-update* command operation (see ACTION) and specify a value of *NONE* for *handlerid*.

Default

If TRIGGERED_HANDLER_ID is not used when an event definition record is added, the following default values are set by the UEM Server:

• Windows	empty string
• UNIX	*NONE*

16.30 USER_COMMAND

Description

The USER_COMAND option specifies a system command that is stored in an event handler record with a handler type of **`CMD`**. It identifies the handler process, which is executed by UEM Server on behalf of the stored event handler.

Usage

Specification Method	Parameter / Value	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-cmd *command*			√	√	
Definition File Keyword	cmd *command*			√	√	

Value

command is the system command stored in an event handler record. If *command* contains spaces, it must be enclosed in double (**"**) quotation marks.

Windows

If quotes are to be saved as part of *command*, the required syntax depends on whether the parameter is specified from the command line or from a definition file.

From the command line, use a backslash (\) to escape the quotes that need to be preserved (for example, **`-cmd “c:\someapp.exe \“quoted string\”`**”).

From a definition file, use an extra double quote to escape the quotes (for example, **`cmd “c:\someapp.exe ““quoted string”””`**).

In both cases, the parameter “**`quoted string`**” will be passed to **`c:\someapp.exe`** and treated as a single command line argument.

UNIX

If *command* contains spaces, it must be enclosed in either single (**’**) or double (**"**) quotation marks.

If quotes are to be saved as part of *command*:

- Enclose *command* in single quotes.
- Use a set of double quotes to enclose the quoted value.

For example, specifying **`-cmd '/usr/someapp “quoted string”'`** will save everything between the single quotes, including the spaces and double quotes, in the event handler record.

16.31 USER_FILE_ENCRYPTED

Description

The USER_FILE_ENCRYPTED option specifies a file that contains the parameters and encrypted values for the USER_ID and/or USER_PASSWORD options.

Storing these parameters and values in a file can be used in situations where it is not desirable to specify them explicitly on the command line. These parameters must be specified in their respective command line formats.

Any file specified by this USER_FILE_ENCRYPTED option must be encrypted with the Universal Encrypt utility. (For more information on Universal Encrypt, see the Stonebranch Solutions Utilities 4.2.0 Reference Guide.

Note: This file should not be specified additionally with the USER_FILE_PLAIN option. If both USER_FILE_ENCRYPTED and USER_FILE_PLAIN specify this file, the file specified by USER_FILE_PLAIN will be used.

It is strongly recommended that any user file (including those that are encrypted) which contains sensitive data should be further protected from unauthorized access using file level security.

UEM Server uses this information to execute an event handler process in the security context of the specified user account. An event-driven UEM Server requires this information to be stored in an event handler record if the USER_SECURITY option is enabled in the UEM Server configuration. This is the only means by which an event-driven UEM Server can access this information. Because an event-driven UEM Server is started without user interaction, it cannot prompt for user account information if it is required but not provided.

To simplify database administration, the file specified in USER_FILE_ENCRYPTED is stored in the event handler record. If this file contains a user ID and password, the UEMLoad utility saves those values in the event handler's USER_ID and USER_PASSWORD fields, respectively.

However, UEM does not rely strictly on the values stored in those fields. UEM re-reads the encrypted file stored in USER_FILE_ENCRYPTED, just before executing the event handler process, in order to obtain any changes made to the account information stored in the file. This makes it possible to update account information for several event handler processes without having to change the event handler records themselves.

Usage

Specification Method	Parameter / Value	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	-x *filename*					
Command Line, Long Form	-encryptedfile *filename*			√	√	
Definition File Keyword	encryptedfile *filename*			√	√	

Value

filename is file containing the encrypted values.

filename can contain either:

- Complete path
- Path that is relative to the directory from which the UEMLoad utility is invoked

If *filename* contains spaces, it must be enclosed in double (**"**) quotation marks.

UNIX

If *filename* contains spaces, it must be enclosed in either single (**'**) or double (**"**) quotation marks.

16.32 USER_FILE_PLAIN

Description

The USER_FILE_PLAIN option specifies a plain text file that contains the parameters and their values for the USER_ID and/or USER_PASSWORD options.

Storing these parameters and values in a file can be used in situations where it is not desirable to specify them explicitly on the command line. These parameters must be specified in their respective command line formats.

Note: This file should not be specified additionally with the USER_FILE_ENCRYPTED option. If both USER_FILE_ENCRYPTED and USER_FILE_PLAIN specify this file, the file specified by USER_FILE_PLAIN will be used.

It is strongly recommended that any user file which contains sensitive data should be further protected from unauthorized access using file level security.

UEM Server uses this information to execute an event handler process in the security context of the specified user account. An event-driven UEM Server requires this information to be stored in an event handler record if the USER_SECURITY option is enabled in the UEM Server configuration. This is the only means by which an event-driven UEM Server can access this information. Because an event-driven UEM Server is started without user interaction, it cannot prompt for user account information if it is required but not provided.

Usage

Specification Method	Parameter / Value	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	-f *filename*			√	√	
Command Line, Long Form	-file *filename*			√	√	
Definition File Keyword	file *filename*			√	√	

Value

filename is file containing the values.

filename can contain either:

- Complete path
- Path that is relative to the directory from which the UEMLoad utility is invoked

If *filename* contains spaces, it must be enclosed in double (**"**) quotation marks.

UNIX

If *filename* contains spaces, it must be enclosed in either single (**'**) or double (**"**) quotation marks.

16.33 USER_ID

Description

The USER_ID option specifies the ID of a user account directly from the command line or definition file.

This is an alternative to specifying the ID from a file.

UEM Server uses this information to execute an event handler process in the security context of the specified user account. An event-driven UEM Server requires this information to be stored in an event handler record if the USER_SECURITY option is enabled in the UEM Server configuration. This is the only means by which an event-driven UEM Server can access this information. Because an event-driven UEM Server is started without user interaction, it cannot prompt for user account information if it is required but not provided.

Note: If UEMLoad specifies an encrypted file (via USER_FILE_ENCRYPTED), it extracts the **`-userid`** value from that file and stores it in the USER_ID field. However, UEMLoad also saves a reference to the encrypted file itself.

When UEM Server executes a process on behalf of the event handler, it re-extracts the **`-userid`** option from the file. This means that if the contents of the encrypted file change after the UEMLoad, the USER_ID value reported in a list or export may not match the actual USER_ID that UEM Server uses when it executes a secured process on behalf of that event handler.

Usage

Specification Method	Parameter / Value	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	-u *userid*			√	√	
Command Line, Long Form	-userid *userid*			√	√	
Definition File Keyword	userid *userid*			√	√	

Value

userid is the ID of a user account.

Valid values for *userid* are valid user accounts that are known to the target system.

16.34 USER_PASSWORD

Description

The USER_PASSWORD option specifies the password for the user account identified by the USER_ID option.

This is an alternative to specifying the password from a plain text or encrypted user file.

UEM Server uses this information to execute an event handler process in the security context of the specified user account. An event-driven UEM Server requires this information be stored in an event handler record if the USER_SECURITY option is enabled in the UEM Server configuration. This is the only way that an event-driven UEM Server can access this information. Because an event-driven UEM Server is started without user interaction, it cannot prompt for user account information if it is required but not provided.

Note: If UEMLoad specifies an encrypted file (via USER_FILE_ENCRYPTED), it extracts the **-pwd** value from that file and stores it in the USER_PASSWORD field. However, UEMLoad also saves a reference to the encrypted file itself.

When UEM Server executes a process on behalf of the event handler, it re-extracts the **-pwd** option from the file. This means that if the contents of the encrypted file change after the UEMLoad, the stored USER_PASSWORD may not match the actual USER_PASSWORD that UEM Server uses when it executes a secured process on behalf of that event handler.

Usage

Specification Method	Parameter / Value	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	-w password			√	√	
Command Line, Long Form	-pwd *password*			√	√	
Definition File Keyword	pwd *password* OR password *password*			√	√	

Value

password is the password for the user account.

16.35 USER_SCRIPT

Description

The USER_SCRIPT option specifies a path to a text file that contains one or more script statements that UEM Server will execute on behalf of the event handler.

UEMLoad will store each of these statements in the action field of the event handler record. As UEM Server prepares to execute the handler, it will write each of these statements to a temporary script file and construct a command line to execute that file.

Usage

Specification Method	Parameter / Value	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-script *filename*			√	√	
Definition File Keyword	script *filename*			√	√	

Value

filename is the script file containing the statements to be executed.

filename can contain either:

- Complete path
- Path that is relative to the directory from which the UEMLoad utility is invoked

If *filename* contains spaces, it must be enclosed in double (**"**) quotation marks.

UNIX

If *filename* contains spaces, it must be enclosed in either single (**'**) or double (**"**) quotation marks.

When specified from the command line, *filename* must be for that of an existing file. If a DEFINITION_FILE option is used to store the event handler record, script statements can be specified in-line.

16.36 USER_SCRIPT_TYPE

Description

The USER_SCRIPT_TYPE option describes the type of statements contained in the script specified by the USER_SCRIPT option.

Usage

Specification Method	Parameter / Value	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	n/a					
Command Line, Long Form	-script_type *type*			√	√	
Definition File Keyword	script_type *type*			√	√	

Value

type is the type of statements in the script specified by USER_SCRIPT.

Windows

When UEM Server executes an event handler with a HANDLER_TYPE of **script**, it writes the statements contained in the Handler Actions field of the handler record to a temporary file. The USER_SCRIPT_TYPE **type** value is used as that file's extension.

On the target system, a file association between *type* and an application responsible for opening files with that extension must be defined. While this association is not required, when the event handler record is added with UEMLoad, it must be defined prior to any attempt by UEM Server to execute the handler. Otherwise, execution of the script will fail.

UNIX

While this option can be specified from the command line, its value will be ignored when UEM Server executes the script.

Default

If this option is not used, a default value of **bat**, set by the UEM Server configuration, is used.

16.37 VERSION

Description

The VERSION option instructs the UEMLoad utility to display version and copyright information.

Usage

Specification Method	Parameter / Value	IBM i	NonStop	UNIX	Windows	z/OS
Command Line, Short Form	-v			√	√	
Command Line, Long Form	-version			√	√	
Definition File Keyword	n/a					

Value

(There are no values for this option.)

CHAPTER 17

Additional Information for Universal Event Monitor

17.1 Overview

This chapter provides additional information used by or specific to Universal Event Monitor.

Table 17.1, below, identifies the type of information provided in this chapter and provides a link to each section.

Information	Description	Page
Character Code Pages	Character code pages for use with Universal Event Monitor.	291
UTT Files	Universal Translate Table (UTT) files are used to translate between Unicode and the local single-byte code page.	293
SSL Cipher Suites	SSL cipher suites for use with Universal Event Monitor.	294
Event Definition Parameters - General	General parameters for event definitions.	295
Event Definition Parameters - Event Type FILE	Parameters that are available in event definitions for an Event Type of FILE.	297
Event Handler Parameters	Parameters for event handlers.	299

Table 17.1 Universal Event Monitor - Additional Information

17.2 Character Code Pages

Table 17.2 identifies the character code pages provided by Stonebranch Inc. for use with Stonebranch Solutions on each supported operating system.

Code Page	CCSID	z/OS	UNIX	Windows	IBM i		HP NonStop
					HFS	LIB	
IBM037	037	√			√	√	
IBM273	273	√			√	√	
IBM277	277	√			√	√	
IBM278	278	√			√	√	
IBM280	280	√			√	√	
IBM284	284	√			√	√	
IBM500	500	√			√	√	
IBM875	875	√					
IBM1047							
IBM1140	1140	√			√	√	
IBM1141	1141	√			√	√	
IBM1142	1142	√			√	√	
IBM1143	1143	√			√	√	
IBM1144	1144	√			√	√	
IBM1145	1145	√			√	√	
IBM1146	1146	√			√	√	
IBM1147	1147	√			√	√	
IBM1148	1148	√			√	√	
IBM4971	4971	√					
ISO8859-1	819		√	√	√		√
ISO8859-2	912		√	√	√		√
ISO8859-3	913		√	√	√		√
ISO8859-4	914		√	√	√		√
ISO8859-5	915		√	√	√		√
ISO8859-6	1089		√	√	√		√
ISO8859-7	813		√	√	√		√
ISO8859-8	916		√	√	√		√
ISO8859-9	920		√	√	√		√
ISO8859-10			√	√	√		√
ISO8859-13	921		√	√	√		√
ISO8859-14			√	√	√		√
ISO8859-15	923		√	√	√		√
PC437	437			√	√		

Code Page	CCSID	z/OS	UNIX	Windows	IBM i		HP NonStop
					HFS	LIB	
PC737	737			√	√		
PC775	775			√	√		
PC850	850			√	√		
PC852	852			√	√		
PC855	855			√	√		
PC857	857			√	√		
PC860	860			√	√		
PC861	861			√	√		
PC862	862			√	√		
PC863	863			√	√		
PC864	864			√	√		
PC865	865			√	√		
PC866	866			√	√		
PC869	869			√	√		
PC874	874			√	√		
WIN1250	1250			√	√		
WIN1251	1251			√	√		
WIN1252	1252			√	√		
WIN1253	1253			√	√		
WIN1254	1254			√	√		
WIN1255	1255			√	√		
WIN1256	1256			√	√		
WIN1257	1257			√	√		
WIN1258	1258			√	√		

Table 17.2 Character Code Pages

17.3 UTT Files

Table 17.3 identifies the Universal Translate Table (UTT) files that are used to translate between Unicode and the local single-byte code page.

Operating System	UTT File Location
z/OS	UTT files are located in the library allocated to the **UNVNLS** ddname. *codepage* is the member name of the UTT file.
UNIX	UTT files are located in the **nls** subdirectory of the installation directory. *codepage* is the base file name of the UTT file. All UTT files end with an extension of **.utt**.
Windows	UTT files are located in the NLS subdirectory of the installation directory. *codepage* is the base file name of the UTT file. All UTT files end with an extension of **.utt**.

Table 17.3 UTT File Locations

17.4 SSL Cipher Suites

Table 17.4, below, identifies all of SSL cipher suites provided by Stonebranch Inc. for use with Universal Enterprise Monitor.

Cipher Suite	Description
RC4-SHA	128-bit RC4 encryption and SHA-1 message digest
RC4-MD5	128-bit RC4 encryption and MD5 message digest
AES256-SHA	256-bit AES encryption and SHA-1 message digest
AES128-SHA	128-bit AES encryption and SHA-1 message digest
DES-CBC3-SHA	128-bit Triple-DES encryption and SHA-1 message digest
DES-CBC-SHA	128-bit DES encryption and SHA-1 message digest
NULL-SHA	No encryption and SHA-1 message digest
NULL-MD5	No encryption and MD5 message digest

Table 17.4 SSL Cipher Suites

17.5 Event Definition Parameters - General

Table 17.5, below, describes the general event definition parameters.

The parameters that make up the record's key are <u>underlined</u>.

Parameter	Description	Remarks
<u>Event ID</u>	Unique identifier for the event definition.	Length of the handler ID cannot be greater than 32 characters. This value is case-sensitive.
<u>Assigned UEM Component</u>	Name of an event-driven Universal Event Monitor Server component responsible for monitoring the event.	
Enabled Flag	A *true* / *false* value that determines whether an event-driven UEM Server processes the event definition.	If *true*, the event is checked periodically by its assigned event-driven UEM Server to see if it should be made active or inactive. If *false*, the event, while still included in a UEM Server's list of assigned components, is not checked to see if it should be made active. [Default is *true*.]
Active Flag	A *yes / no* value that indicates whether UEM Server is monitoring the event.	If *yes*, the current date/time is greater than the defined **Activation Date/Time**, but less than the defined **Inactivation Date/Time**. If *no*, the current date/time is greater than the defined **Inactivation Date/Time**. This parameter is set only through UEM Server. It cannot be set using the UEMLoad utility. [Default is *no*.]
Event Type	System event represented by the event definition.	Value corresponds to one of the supported event types. (Currently, **`FILE`** is the only support event type.)
Activation Date	Date on which UEM will begin checking for the occurrence of the system event represented by the event definition.	Specified using the format *YYYY.MM.DD*. [Default is current date.]
Activation Time	Time on the **Activation Date** at which UEM will begin checking for the occurrence of the system event represented by the event definition.	Specified using the format *HH:MM*. [Default is current time.]
Inactivation Date	Date on which UEM will stop checking for the occurrence of the system event represented by the event definition.	Specified using the format *YYYY.MM.DD*. [Default is *2038.01.16* for event definitions stored with the UEMLoad utility.
Inactivation Time	Time on the **Inactivation Date** at which UEM will stop checking for the occurrence of the system event represented by the event definition.	Specified using the format *HH:MM*. [Default is *23:59*.]

Parameter	Description	Remarks
Tracking Interval	Frequency with which UEM tests for the completion of an occurrence of the system event represented by the event definition.	Value is expressed in seconds. Value depends on the event type. Some system events will be considered complete as soon as their occurrence is detected by UEM. For these events, this parameter should be *0* (zero). [Default is *10*.]
Triggered Event Handler ID	ID of an event handler database record whose specified actions should be taken when an event occurrence satisfies its defined completion criteria.	Value is case sensitive. Value can be blank, in which case the default value specified in the UEM Server configuration (empty string for Windows, *NONE* for UNIX) will be used.
Rejected Event Handler ID	ID of an event handler database record whose specified actions should be taken if one or more tracked event occurrences fail to complete before the event is made inactive by UEM.	Value is case sensitive. Value can be blank, in which case the default value specified in the UEM Server configuration (empty string for Windows, *NONE* for UNIX) will be used.
Expired Event Handler ID	ID of an event handler database record whose specified actions should be taken if an event is made inactive with no occurrence of the system event represented by the event definition being detected by UEM.	Value is case sensitive. Value can be blank, in which case the default value specified in the UEM Server configuration (empty string for Windows, *NONE* for UNIX) will be used.
Event Handler Options	String that contains one or more parameters that UEM adds to the command line it constructs to execute an event handler's specified command or script.	Added to the command line for every process executed for the event handlers referenced by this event definition. This includes those event handlers referenced by the Triggered Event Handler ID, Rejected Event Handler ID, and Expired Event Handler ID fields. If a value is stored in the Options field of any event handler that this event definition references, the value stored in this field is placed after the value of that field when UEM constructs the command line.
Last Update Date/Time	Date and time the event definition was created or last modified.	
Last Update User	ID of the user account that created or most recently updated the event definition.	

Table 17.5 Event Definition Parameters - General

17.6 Event Definition Parameters - Event Type FILE

Table 17.6, below, describes the parameters that are available in event definitions for an Event Type of **`FILE`**.

Parameter	Description	Remarks
File Specification	Complete path of the file to watch.	File specification can contain wildcards: • ? is available to match up to 1 character. • * can be used to match 0 or more characters.
Minimum File Size	Smallest size that a file can be in order to be considered complete.	Value of 0 indicates that a file of any size is acceptable. To specify a storage unit for this value, add either of the following after the file size: • (b)ytes • (k)ilobytes • (m)egabytes • (g)igabtyes If a storage unit is not specified, UEM assumes the file size is given in bytes.
Rename File	Indicates whether or not UEM Server will rename the file prior to executing the triggered event handler. This will prevent multiple detections of the same file.	Acceptable values are *yes* or *no*. [Default is *yes*.]

Parameter	Description	Remarks
Rename File Specification	Format used by UEM to rename a file.	Complete file name or a file mask that accepts specific well-known variables that will be substituted by UEM Server at run time. When specified, variables must be in the format **$(var)**. The following variables are available: • **$(compname)** – component name • **$(compid)** - component ID • **$(date)** - current date in the format YYYYMMDD • **$(time)** - current time, n the format HHMMSS • **$(origname)** - original base file name, minus its last extension • **$(origext)** - original file extension • **$(seqnum)** - sequence number that starts at 0 when a UEM Server component is started, and is then incremented by 1 for each file renamed. If no path is specified, the file is simply renamed. Otherwise, the file can be renamed, moved from its original location, and placed in the path specified by this parameter. If no value is specified, a default value of **$(compname).$(compid).$(date).$(seqnum)**, specified in the UEM Server configuration, will be used.

Table 17.6 Event Definition Parameters - Event Type **FILE**

17.7 Event Handler Parameters

Table 17.7, below, describes the event handler parameters.

The parameters that make up the record’s key are underlined.

Parameter	Description	Remarks
Encrypted File	Complete path to a file encrypted with Universal Encrypt. UEM reads the contents of this file to obtain the User ID and Password of a user account, which is then used to establish a security context in which the event handler process is executed.	A Password may not be necessary on UNIX, depending on the existing EVENT_HANDLER_ACL entries. Storing the path to an encrypted file allows the file to be shared between Stonebranch Solutions applications. This makes it unnecessary to update individual event handler records whenever account information is updated or security requirements change.
Handler Actions	Command or script executed by UEM Server for this handler.	For records with a Handler Type of *cmd*, this parameter contains the syntax required to execute a system command or other command line application. For records with a Handler Type of *script*, this parameter contains one or more system commands that are executed collectively as a batch script.
Handler ID	Unique identifier for the event handler.	Length of the handler ID cannot be greater than 32 characters. Value is case-sensitive.
Handler Type	Type of action that will be executed.	If the Handler Actions field contains a system command or other command line application, this field has a value of *cmd*. If the Handler Actions field contains a series of system commands that should be executed as a script file, this field has a value of *script*.
Last Update Date/Time	Date and time that the event handler was created or last modified.	
Last Update User	ID of the user account that created or most recently updated the event handler.	
Maximum Acceptable Return Code	Highest value that the handler process can return to still be considered as having executed successfully.	If the value returned by the action is less than or equal to the value specified by this parameter, UEM will report a successful completion of the handler process. If the value returned by the action is greater than the value specified by this parameter, UEM will report a failure of the handler process.

Parameter	Description	Remarks
Options	String with one or more parameters that UEM adds to the command line that it constructs in order to execute a specified command or script.	This parameter serves the same purpose as the event definition's Handler Options parameter, but differs in scope. An event definition's Handler Options are applied to every handler process executed on its behalf. An event definition's Handler Options may specify global handler execution options (for example, output redirection) or may customize event handler behavior for that specific event. An event handler's Options value, on the other hand, is applied every time the handler is executed, regardless of which event definition causes it to execute. When command line options are specified in the event definition (via Handler Options) and in the event handler (via Options), both are used. However, the event handler's Options value is added first.
Password	Password of the user account specified in the User ID parameter.	If the UEM Server processing this record is configured to run without security, this parameter is ignored. On UNIX-based systems, a UACL entry can be defined that permits handler execution for some user accounts without requiring authentication. In such a situation, this parameter is ignored. This parameter is encrypted prior to being stored in the database.
Script Type	Script interpreter to invoke to execute script statements contained in the Handler Actions parameter.	On UNIX-based systems, this value may be blank, in which case it will default to *sh*. A script's execution shell also can be specified as the first statement contained in the Handler Actions parameter. On Windows-based systems, this value will be used as a file extension for the temporary script constructed by UEM Server in order to execute the script statements specified in the Handler Actions parameter. To insure proper execution of the script, a file association should exist on the target system that specifies the application with which the script should be executed. This parameter is ignored if the Handler Type has a value of *cmd*.
User ID	ID of the user account in whose security context the handler actions will be run.	If the UEM Server processing this record is configured to run without security, the value in this parameter is ignored.

Table 17.7 Event Handler Parameters

APPENDIX A

Standard I/O Redirection and Event Handler Processes

A.1 Overview

Universal Event Monitor currently does not provide any native support for redirecting standard input and output to or from a process. However, it is possible to instruct the operating system to redirect standard I/O during process execution. I/O redirection is done using one of the following redirection symbols:

- **< -** Redirect standard input using an existing file. If an application supports it, standard input redirection can be used as a batch alternative to input that is normally obtained interactively from the user.
- **> -** Redirect standard output to a specified file. When a single **>** is used and the specified file exists, its contents are overwritten. The file is created if it does not exist.
- **>>** - Redirect standard output and append it to a specified file. If the specified file exists when this symbol is used, output generated by the process is added to the end of the specified file, preserving the file's current contents. The file is created if it does not exist.
- **2>** - Redirect standard error to a specified file. If the specified file exists, its contents are replaced. The file is created if it does not exist. If standard output also is being redirected to a file, an ampersand (**&**) followed by a **1** (which the operating system recognizes as an identifier for standard output) can be used in place of a file name. This causes standard error to be written to the same file used for standard output. For example, **2>&1** instructs the operating system to send all messages generated by the application that are destined for standard output and standard error to be written to the same file.
- **2>>** - Redirect standard error and append it to a specified file. If the specified file exists when this symbol is used, messages written by the application to standard error are added to the end of the specified file. If standard output also is being redirected to a file, an ampersand (**&**) followed by a **1** (which the operating system recognizes as an identifier for standard output) may be used in place of a file name. This causes standard error to be appended to the same file used for standard output.

For example, **2>>&1** instructs the operating system to send all messages generated by the application that are destined for standard output and standard error to be written to the same file.

This section describes how to redirect standard I/O for event handlers that execute a command or a script. It also explains how an event definition's *HANDLER_OPTIONS* option can be used to control standard i/o redirection.

A.2 Command Execution

If an event handler executes a command, standard I/O redirection can be specified with the handler's *USER_COMMAND* option. The value of this option would be the same as it would be if you were issuing the command and redirecting I/O from the command prompt.

Figure A.1, below, provides examples.

```
Example 1
-cmd “dir >dirout.txt”

Example 2
-cmd “someapp <appinput.dat >applog.txt 2>&1”

Example 3
-cmd “dailybkup >>backup.log 2>>backup_err.log”
```

Figure A.1 Standard I/O redirection using the USER_COMMAND parameter.

Example 1 shows how to send the results of a `dir`ectory listing on a Windows system to a file in the current directory named **`dirout.txt`**.

Example 2 can be run on a Windows or UNIX system. It demonstrates the execution of an application named **`someapp`** that receives its input from a file named **`appinput.dat`**, and is redirecting all messages destined for **`stdout`** and **`stderr`** to a file named **`applog.txt`**. The contents of **`applog.txt`** are overwritten each time that the command executes.

Example 3 also can be run on a Windows or UNIX system. It executes an application named **`dailybkup`**, and appends all messages destined for **`stdout`** to **`backup.log`** and all messages destined for **`stderr`** to a file named **`backup_err.log`**.

Because the file name does not change with each invocation of the event handler, this method is best suited for event handler processes whose output does not need to be captured in unique files. If multiple event occurrences cause this event handler process to be executed at the same time, the output from each instance of the event handler process will be interleaved within the same file.

Windows

Special consideration should be given when executing several instances of an event handler process that redirects output to a single file. Sharing violations have been observed in this situation that prevent output from being properly captured. For these cases, it is recommended that a method be used that provides for assignment of a unique file name to each process instance.

A.3 Script Execution

If an event handler executes a script, standard I/O redirection can be done within the script itself. This method provides a little more flexibility with respect to assigning unique names to files that store captured output. If unique names are needed, logic can be set up within the script to construct the name and to ensure that the file does not already exist.

Figure A.2 and Figure A.3, below, illustrate sample scripts that contain logic for generating unique file names.

UNIX

This sample is a Bourne shell script that uses a combination of environment variables set by the UEM Server, the script's process ID, and the current system date/time in order to construct a unique file name used to store the output of the **ls -al** command.

```
#!/bin/sh

rc=0

## Set a date string in the format yyyy.mm.dd.
dt=`date +%Y.%m.%d`

## Set a time string in the format hh.mm.ss
tm=`date +%H.%M.%S`

## Save the process ID
pid=$$

## Remove the extension from the file tracked by UEM.
uemFName=`basename “$UEMORIGFILE” | sed 's/\(^.*\)\.\(.*$\)/\1/'`

## Construct a filename using the name of the file tracked by UEM, the
## current date, the current time, and the process ID.
fname=$uemFName.$dt.$tm.$pid.txt

## Execute the command. Redirect the output to the file name
## constructed above.
ls -al >$fname

## Set the return code and exit the script
rc=$?
exit $rc
```

Figure A.2 Logic to Generate Unique File Name in Bourne Shell Script

Windows

This sample is a Windows batch file that uses a combination of environment variables set by the UEM Server and the current system date/time in an attempt to construct a unique file name used to store the output of the **`dir`** command. Because multiple instances of this script can be executing at the same time, and a process ID is not available to a Windows batch file, a loop exists to retry the file name construction if a matching file is found.

```
@echo off

:: Main script flow
set rc=0

call :SetVariables
call :SetupOutputFile

:: Now, execute the command. Redirect the contents of this directory
:: and all subdirectories to the file identified by %fname%.

dir /o/-p/s >%fname%
set rc=%ERRORLEVEL%

:: Exit the script
goto Exit

::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
:: Subroutines
::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::

:SetVariables
:: Set a date string in the format yyyy.mm.dd.
for /f "tokens=2,3,4 delims=/ " %%a in ('date /t') do set dt=%%c.%%a.%%b

:: Set a time string in the format hh.mm.ss
for /f "tokens=1,2 delims=: " %%a in ('time /t') do set tm=%%a.%%b

:: Remove file extension
for /f "tokens=1,2 delims=." %%a in ("%UEMORIGFILE%") do set uemFName=%%a

:: Set the file name
set fname=%uemFName%.%dt%.%tm%.txt

:SetVariablesExit
goto :EOF

(continued)
```

```
(continued)

::
:: Test for the existence of the output file. If the file is found, loop
:: for a while then reset the time variable and check again.
::
:SetupOutputFile
:: If the file doesn't exist, create it to prevent another instance of this
:: script from grabbing it.
if not exist %fname% (
echo. >%fname%
goto SetupOutputFileExit
)

:: Otherwise, wait for a count of 10,000 (approx 1 sec) and try again.
if exist %fname% for /l %%I in (1,1,10000) do echo. >NUL

call :SetVariables
goto SetupOutputFile

:SetupOutputFileExit
goto :EOF

:Exit
exit %rc%
```

Figure A.3 Logic to generate unique file name in Windows batch file.

Note: In both of these examples, the redirection is done at the point at which the process is invoked. If the script itself also generates output that needs to be captured, this file would need to be provided via the event definition's handler options (see Section A.4 Handler Options).

A.4 Handler Options

An event's HANDLER_OPTIONS option can be used to establish standard I/O redirection for every event handler process executed for the event. These options are specified using the HANDLER_OPTIONS option of the event definition or the **`-handler_opts`** command line option of the UEM Manager. In both implementations, the value specified for this parameter is appended to the command line that the UEM Server constructs to execute the specified user command or script.

This option has the same issues regarding unique file names as those described above in the Command Execution section. If it is possible that multiple, concurrently tracked event occurrences will result in the simultaneous execution of the event's handlers, all output generated by all handlers will be written to the same file. If output from each handler process needs to be captured in unique files, the method described in the Script Execution section above is a better alternative.

One advantage to using an event's handler options parameter is that an event's triggered, rejected, and/or expired event handlers can be changed without changing the way output is captured by the respective event handler process.

An event's handler options parameter may also be used to redirect standard input from a file. Capturing output may then be handled with the handler options parameter or by using one of the other methods described above.

The examples in Figure A.4, below, demonstrate the use of this parameter for event definitions and event handlers added with the UEMLoad utility, and for events defined and executed from a UEM Manager's command line.

```
Example 1
UEMLoad utility:
uemload -add -event_id event001 -handler_opts “>netstat.txt”
-triggered_id handler001 -expired_id expiredhandler
-event_type file -filespec uemtest.dat
uemload -add -handler_id handler001 -cmd netstat
-encryptedfile userfile.enc
uemload -add -handler_id expiredhandler -cmd “echo Event expired”
-encryptedfile userfile.enc

UEM Manager command line:
uem -event_type file -filespec uemtest.dat
-host rmthost -user rmtuser -pwd password
-handler_opts “>dirout.txt” -triggered -cmd netstat
-expired -cmd “echo Event expired”

Example 2
UEMLoad utility:
uemload -add -event_id event002 -triggered_id handler002
-handler_opts “<appinput.dat >applog.txt 2>&1”
-event_type file -filespec uemtest.dat
uemload -add -handler_id handler002 -cmd someapp
-encryptedfile userfile.enc

UEM Manager command line:
uem -event_type file -filespec uemtest.dat
-host rmthost -user rmtuser -pwd password
-handler_opts “<appinput.dat >applog.txt 2>&1”
-triggered -cmd someapp

Example 3
UEMLoad utility:
uemload -add -event_id event003 -inact_date_time 2005.12.31,23:59
-handler_opts “>>backup.log 2>>backup_err.log”
-rejected_id handler003 -event_type file -filespec uemtest.dat
uemload -add -handler_id handler003 -cmd dailybkup

UEM Manager command line:
uem -event_type file -inact_date_time 2005.12.31,23:59 -filespec uemtest.dat
-host rmthost -user rmtuser -pwd password
-handler_opts “>>backup.log 2>>backup_err.log”
-rejected -cmd dailybkup
```

Figure A.4 Using handler options when storing event definitions

For each of the event definitions shown above, the OS installed on the host **`rmthost`** can be either Windows or UNIX. If the UEM Manager is executed from z/OS, the name of the executable (that is, **`uem`**) would be omitted. However, the rest of the parameters would be entered as shown in the JCL's SYSIN DD statement.

The first example shows an event of type *FILE* that will monitor a file named **`uemtest.dat`**. If an occurrence of this file is detected, the **`netstat`** command will be executed when the occurrence enters a triggered state. If the event expires, an “Event expired” message is issued. In both situations, the output from the command is written to a file named **`netstat.txt`**.

The second example also shows an event of type *FILE* that is set up to monitor a file named **`uemtest.dat`**. If an occurrence of this file is triggered, the application **`someapp`** will be executed. According to the **`-handler_options`** parameter, this application will receive input redirected from a file named **`appinput.dat`**. All output, including output destined for standard error, is written to a file named **`applog.txt`**.

Finally, the last example is also monitors a file named **`uemtest.dat`**. This event executes an event handler process only if an occurrence of the file is detected and that file fails to complete by *midnight* on *31 December 2005*. The handler process executed is named **`dailybkup`**. All messages issued by the application that are destined for standard output are written at the end of a file named **`backup.log`**. All messages destined for standard error are appended to a file named **`backup_err.log`**. One or both files are created if they do not exist when the handler process is executed.

APPENDIX B

Environment Variables Set by Universal Event Monitor

Before it executes an event handler process, Universal Event Monitor can set one or more environment variables that contain information regarding a monitored event and/or an event occurrence. The environment variables that are set depend on a monitored event's type and processing state.

The environment variables set for the different event types are listed in Table B.1, below. The three right-most columns represent the event processing states:

- (***T***)riggered
- (***R***)ejected
- (***E***)xpired

Event Type	Variable Name	Description	T	R	E
FILE	UEMFILESPEC	File specification from event definition	√	√	√
FILE	UEMORIGFILE *	Name of file monitored and tracked by UEM, including the complete path	√	√	
FILE	UEMORIGFNAME	Name of file monitored and tracked by UEM	√	√	
FILE	UEMORIGPATH	Location of file monitored and tracked by UEM	√	√	
FILE	UEMRENAMEDFILE *	Name of file after being renamed by UEM, including the complete path	√		
FILE	UEMRENAMEDFNAME	Name of file after being renamed by UEM	√		
FILE	UEMRENAMEDPATH	Location of file after being renamed by UEM	√		
FILE	UEMFILESIZE	Last recorded file size in space unit of bytes	√	√	

Table B.1 Environment Variables Set by Universal Event Monitor

APPENDIX C

Customer Support

Stonebranch, Inc. provides customer support, via telephone and e-mail, for Universal Event Monitor and all Stonebranch Solutions components.

E-MAIL

All Locations

support@stonebranch.com

Customer support contact via e-mail also can be made via the Stonebranch website:

www.stonebranch.com

TELEPHONE

Customer support via telephone is available 24 hours per day, 7 days per week.

North America

(+1) 678 366-7887, extension 6

(+1) 877 366-7887, extension 6 [toll-free]

Europe

+49 (0) 700 5566 7887

stonebranch

950 North Point Parkway, Suite 200

Alpharetta, Georgia 30005

U.S.A.