NetIQ® Module Builder

User Guide

November 2015

www.netiq.com/documentation

Legal Notice

For information about NetIQ legal notices, disclaimers, warranties, export and other use restrictions, U.S.Government restricted rights, patent policy, and FIPS compliance, see https:/www.netiq.com/company/legal/.

Copyright © 2015 NetIQ Corporation. All Rights Reserved.

For information about NetIQ trademarks, see http://www.netiq.com/company/legal/. All third-party trademarks are the property of their respective owners.

Contents

About this Book and the Library

The NetIQ AppManager product (AppManager) is a comprehensive solution for managing, diagnosing, and analyzing performance, availability, and health for a broad spectrum of operating environments, applications, services, and server hardware.

AppManager provides system administrators with a central, easy-to-use console to view critical server and application resources across the enterprise. With AppManager, administrative staff can monitor computer and application resources, check for potential problems, initiate responsive actions, automate routine tasks, and gather performance data for real-time and historical reporting and analysis.

Intended Audience

This guide provides information for individuals responsible for installing an AppManager module and monitoring specific applications with AppManager.

Other Information in the Library

The library provides the following information resources:

Installation Guide for AppManager

Provides complete information about AppManager pre-installation requirements and step-by-step installation procedures for all AppManager components.

User Guide for AppManager Control Center

Provides complete information about managing groups of computers, including running jobs, responding to events, creating reports, and working with Control Center. A separate guide is available for the AppManager Operator Console.

Administrator Guide for AppManager

Provides information about maintaining an AppManager management site, managing security, using scripts to handle AppManager tasks, and leveraging advanced configuration options.

Upgrade and Migration Guide for AppManager

Provides complete information about how to upgrade from a previous version of AppManager.

Management guides

Provide information about installing and monitoring specific applications with AppManager.

Help

Provides context-sensitive information and step-by-step guidance for common tasks, as well as definitions for each field on each window.

The AppManager library is available in Adobe Acrobat (PDF) format from the NetIQ Web site: www.netiq.com/support/am/extended/documentation/default.asp?version=AMDocumentation.

Conventions

The library uses consistent conventions to help you identify items throughout the documentation. The following table summarizes these conventions.

Convention	Use
Bold	◆ Window and menu items ◆ Technical terms, when introduced
Italics	◆ Book and CD-ROM titles ◆ Variable names and values ◆ Emphasized words
`Fixed Font`	◆ File and folder names ◆ Commands and code examples ◆ Text you must type ◆ Text (output) displayed in the command-line interface
Brackets, such as *[value]*	◆ Optional parameters of a command
Braces, such as *{value}*	◆ Required parameters of a command
Logical OR, such as *value1*|*value2*	◆ Exclusive parameters. Choose one parameter.

About NetIQ Corporation

NetIQ, an Attachmate business, is a global leader in systems and security management. With more than 12,000 customers in over 60 countries, NetIQ solutions maximize technology investments and enable IT process improvements to achieve measurable cost savings. The company's portfolio includes award-winning management products for IT Process Automation, Systems Management, Security Management, Configuration Audit and Control, Enterprise Administration, and Unified Communications Management. For more information, please visit www.netiq.com.

Contacting Sales Support

For questions about products, pricing, and capabilities, please contact your local partner. If you cannot contact your partner, please contact our Sales Support team

Worldwide:	www.netiq.com/about_netiq/officelocations.asp
United States and Canada:	888-323-6768
Email:	info@netiq.com
Web Site:	www.netiq.com

Contacting Technical Support

For specific product issues, please contact our Technical Support team.

Worldwide:	www.netiq.com/Support/contactinfo.asp
North and South America:	1-713-418-5555
Europe, Middle East, and Africa:	+353 (0) 91-782 677
Email:	support@netiq.com
Web Site:	www.netiq.com/support

Contacting Documentation Support

Our goal is to provide documentation that meets your needs. If you have suggestions for improvements, please email Documentation-Feedback@netiq.com. We value your input and look forward to hearing from you.

Contacting the Online User Community

Qmunity, the NetIQ online community, is a collaborative network connecting you to your peers and NetIQ experts. By providing more immediate information, useful links to helpful resources, and access to NetIQ experts, Qmunity helps ensure you are mastering the knowledge you need to realize the full potential of IT investments upon which you rely. For more information, please visit http://community.netiq.com.

1 Introducing NetIQ Module Builder

Module Builder extends the coverage of AppManager by using the knowledge of subject matter experts to create customized AppManager modules specific to a company’s set of applications.

Module Builder enables users to:

- Monitor applications for which AppManager does not provide a commercial module
- Monitor custom applications created by your own company or a consulting company
- Facilitate communication between an application’s subject matter expert, who might not know AppManager, and the AppManager expert, who might not know the monitored application

1.1 AppManager Modules

AppManager provides system administrators with a central, easy-to-use console to view critical server and application resources across the enterprise. With AppManager, administrative staff can monitor computer and application resources, check for potential problems, initiate responsive actions, automate routine tasks, and gather performance data for real-time and historical reporting and analysis.

AppManager modules provide Knowledge Scripts specific to the application being monitored by AppManager. You run Knowledge Scripts on a set of computers to track performance, health, and other data relevant to keeping the application functional.

Using the Module Builder Editor, a subject matter expert creates the foundation for a new, customized AppManager module. A typical Module Builder custom module includes the following items:

- Up to five custom Knowledge Scripts that track the health and status of processes, services, performance counters, log files, and event log data for the application you are managing.
- A Discovery Knowledge Script that finds all relevant objects on a particular computer that you can monitor with AppManager.

1.2 Module Builder Components

Module Builder includes the following components:

Module Builder Editor

The Module Builder Editor is a stand-alone application that an application’s subject matter expert uses to gather information about the application to be monitored. The subject matter expert then uses the Module Builder Editor to select and configure a set of Windows components to monitor for that application. These settings create the basis of a new module for the application.

Module Builder managed object

The managed object is a group of files that support the custom Knowledge Scripts and the Discovery script generated by the Module Builder Editor. The managed object also helps generate the Knowledge Scripts set up with the Module Builder Editor.

Module Builder custom module

The custom module contains the Knowledge Scripts generated by the Module Builder Editor and supported by the managed object. An AppManager expert uses these scripts to manage and monitor the specified application in AppManager.

1.3 Module Builder Roles

Module Builder users perform two expert roles. The same person might fill both roles.

Subject matter expert

The subject matter expert is the person responsible for the application being managed. The subject matter expert uses the Module Builder Editor to identify and configure the monitoring parameters for the application. The subject matter expert does *not* need to know AppManager.

The subject matter expert first runs the Module Builder Editor to collect information about the application to be monitored. The expert uses the Module Builder Editor on the computer where that application is running. As the interview progresses, the Module Builder Editor generates a `.mob` file that the subject matter expert can give to the AppManager expert. For more information, see Chapter 3, “Identifying Application Conditions to Monitor,” on page 19.

AppManager expert

The AppManager expert receives the new `.mob` file from the subject matter expert and reviews the monitoring parameters chosen by the subject matter expert for the application. The AppManager expert does not need to be familiar with the subject matter expert’s application, and the AppManager expert does not require access to the application. In most cases, the AppManager expert will also be the AppManager administrator.

The AppManager expert reviews the monitoring parameters in the `.mob` file and generates a set of AppManager Knowledge Scripts, which includes a Discovery Knowledge Script. For more information, see Section 4.1, “Generating Knowledge Scripts for a New Module,” on page 37.

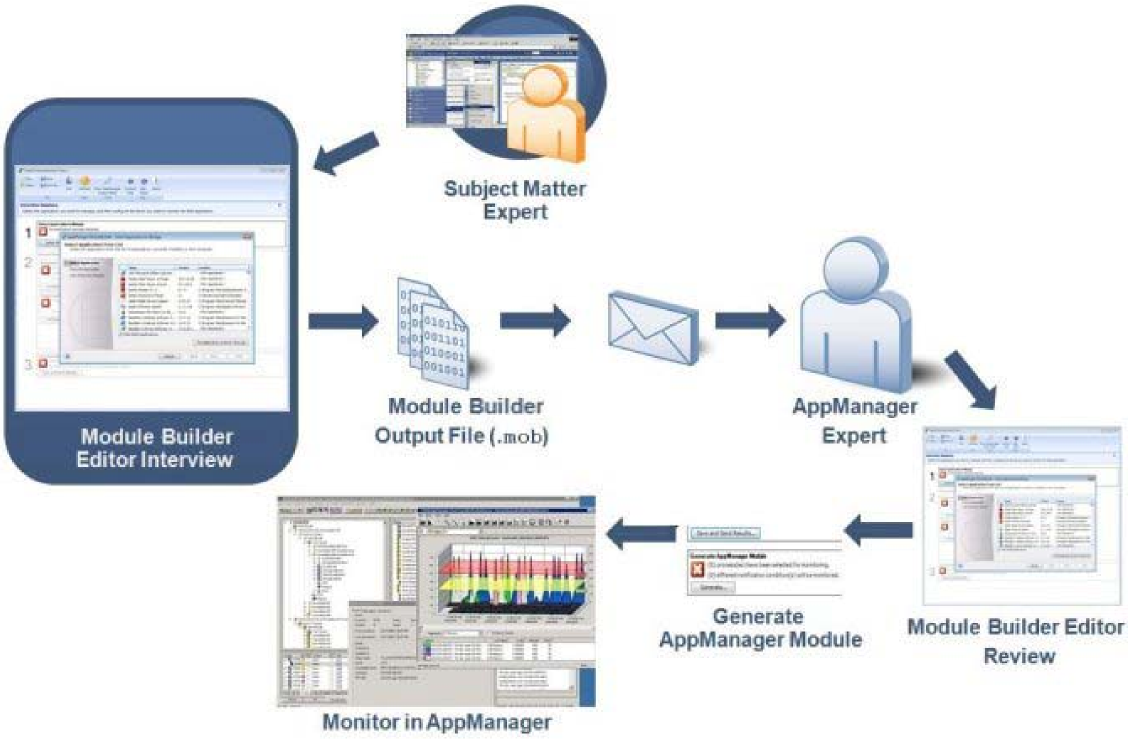
The default mode for the Module Builder Editor is Subject Matter Expert mode. For more information, see Section 3.11, “Changing Module Builder Editor Modes,” on page 34.

1.4 Module Builder Process

The subject matter expert and the AppManager expert work together to create a module, as illustrated in the following process.

The subject matter expert runs the Module Builder Editor on the computer where the application to be monitored is running.

1. The subject matter expert answers a series of questions about the application to be monitored. As the interview progresses, the Module Builder Editor gathers the information into a `.mob` file.
2. After completing the interview in the Module Builder Editor, the subject matter expert shares the `.mob` file with the AppManager expert.


Subject Matter Expert
Module Builder Editor Interview
Module Builder Output File (.mob)
AppManager Expert
Module Builder Editor Review
Generate AppManager Module
Monitor in AppManager

3. The AppManager expert uses the AppManager Expert mode in the Module Builder Editor to review the `.mob` file. The AppManager expert can review the file on the same computer used by the subject matter expert, or on a different computer.
4. As needed, the AppManager expert makes changes to the monitoring parameters in the `.mob` file and sends the file back to the subject matter expert for more information.
5. The AppManager expert clicks the **Generate** button in the Module Builder Editor to create a set of customized AppManager Knowledge Scripts from the `.mob` file.
6. The AppManager expert checks the customized Module Builder Knowledge Scripts into the AppManager repository and can begin monitoring the application.

1.5 Module Builder Licensing

Each computer that will run Knowledge Scripts created with Module Builder requires a separate license. You can use custom modules created using Module Builder to monitor as many applications as you want on the same computer, without additional licensing fees.

No license is required to run the Module Builder Editor on any computer.

2 Installing Module Builder

This chapter describes system requirements and installation instructions for the Module Builder Editor and the Module Builder managed object. This release of the Module Builder Editor, the Module Builder managed object, and the Module Builder custom module is supported only on Microsoft Windows operating systems.

2.1 System Requirements for the Module Builder Editor

The Module Builder Editor requires the following software.

Category	Requirement
Microsoft Windows operating system	One of the following: ◆ Windows Server 2012 R2 ◆ Windows Server 2012 ◆ Windows Server 2008 R2, including SP1, Enterprise or Standard Edition ◆ 32-bit or 64-bit Windows Server 2008, including SP1 or SP2, Enterprise or Standard Edition ◆ 32-bit or 64-bit Windows 7, including SP1 or SP2 ◆ 32-bit or 64-bit Windows Vista, including SP1, Business or Enterprise Edition ◆ 32-bit or 64-bit Windows Server 2003 R2, including SP2, Enterprise or Standard Edition ◆ 32-bit or 64-bit Windows XP with SP3
Microsoft .NET Framework	2.0.50727, at minimum
Network Permission	Subject matter expert must be a member of the local Administrators group.

NOTE: If you are using Windows Vista or Windows Server 2008 with User Account Controls (UAC) enabled, you might need to install Module Builder Editor using the **Run as administrator** command, or you need to update your credentials to install the Module Builder Editor. For more information, see technet.microsoft.com/en-us/library/cc709628(WS.10).aspx.

NOTE: Microsoft .NET Framework 3.5 (includes .NET 2.0 and 3.0) features are not enabled by default on Windows Server 2012 and 2012 R2. Before installing Module Builder Editor on Windows Server 2012 or 2012 R2, ensure that Microsoft .NET Framework 3.5 is enabled. For more information on how to enable Microsoft .NET Framework 3.5, see technet.microsoft.com/en-in/library/dn482071.aspx.

2.2 Installing the Module Builder Editor

You can install the Module Builder Editor using a standard installation or a silent installation.

2.2.1 Performing a Standard Installation

For a standard installation of the Module Builder Editor, you can run the `setup.exe` file from the download folder.

To install the Module Builder Editor:

1. Navigate to the download folder and double-click the `setup.exe` for the Module Builder Editor: ModuleBuilderEditor-1.0.***xxx***.0-setup.exe.
2. In the `\Program Files\NetIQ\Module Builder\Editor` folder, double-click the installation file for the Module Builder Editor: ModuleBuilderEditor-1.0.***xxx***.0.msi.

If any installation problems occur, an installation log file is available in the `\Windows\Temp` folder.

2.2.2 Performing a Silent Installation

Perform a silent installation to install the Module Builder Editor without user intervention. From a command prompt, instruct the setup program associated with the Module Builder Editor to perform the installation. The command you enter can contain all of the installation options you have selected, or it can allow the installation to use the default options.

To silently install the Module Builder Editor on the local computer, run the following command from the directory where the Module Builder Editor installation file is located. This command is provided as an example:

```
msiexec /qn /l* C:\WINDOWS\Temp\ModuleBuilderInstall.txt /i ModuleBuilderEditor-
x.x.x.0.msi INSTALLDIR="C:\Program Files\NetIQ\Module Builder"
```

The following table describes the commands used in the silent installation.

Keyword or Parameter	Description
`/qn`	Specifies a silent installation without the Module Builder Editor setup interface; no feedback is provided on the silent installation process.
`/l*`	Specifies that you want to create an installation log in `C:\WINDOWS\Temp\ModuleBuilderInstall.txt`
`/i`	Specifies the name of the Module Builder Editor installation package in `.msi` format.
`INSTALLDIR`	Specifies where you want to install the Module Builder Editor. **Note** This is the only option you can configure for the Module Builder Editor installation.

2.3 System Requirements for the Module Builder Managed Object

The Module Builder managed object is a group of files that support the custom Knowledge Scripts and the Discovery script generated by the Module Builder Editor. To use Module Builder Knowledge Scripts in AppManager, you must install the managed object.

The Module Builder managed object requires the following software.

Category	Requirement
AppManager installed on the AppManager repository (QDB) computers, the agent computers, and the console computers	7.0, at minimum **Notes** ◆ The AppManager agent services (`NetIQmc` and `NetIQccm`) can be running under the LocalSystem account or under an account in the local Administrators group. Both services must be running under the same account. ◆ For support of Windows Server 2008, hotfix 71704 is required. For more information, see the AppManager Suite Hotfixes Web page.
Microsoft Windows operating system on the managed computer (agent)	One of the following: ◆ Windows Server 2012 R2 ◆ Windows Server 2012 ◆ Windows Server 2008 R2, including SP1, Enterprise or Standard Edition ◆ 32-bit or 64-bit Windows Server 2008, including SP1 or SP2, Enterprise or Standard Edition ◆ 32-bit or 64-bit Windows 7, including SP1 or SP2 ◆ 32-bit or 64-bit Windows Vista, including SP1, Business or Enterprise Edition ◆ 32-bit or 64-bit Windows Server 2003 R2, including SP2, Enterprise or Standard Edition ◆ 32-bit or 64-bit Windows XP with SP3 ◆ 32-bit Windows 2000, including SP2, SP3, or SP4 (AppManager console or repository only)
Microsoft .NET Framework on the managed computer (agent)	2.0.50727, at minimum
The software application or applications you want to monitor on the managed computer (agent)	Determined by subject matter expert and AppManager expert

NOTE: The Module Builder Editor is not supported on a Windows 2000 computer, but the Module Builder managed object in AppManager supports an AppManager repository (QDB) or console computer on Windows 2000.

2.4 Installing the Module Builder Managed Object

Install the Module Builder managed object files on each agent, console, and repository computer. The Module Builder managed object files are in the `Install` folder of the download folder. This folder contains the `.msi` file for the Module Builder managed object: `AM70-ModuleBuilder-1.`*`x`*`.`*`xxx`*`.0.msi`.

If you are using Control Center to deploy the installation package, check in and then configure the installation package, `AM70-ModuleBuilder-1.`*`x`*`.`*`xxx`*`.0.xml`, which is included with the module setup program. For more information about the `.xml` file, see the NetIQ ModuleBuilder Readme.

For more information about deploying modules on agents, see the *Control Center User Guide for AppManager*.

The installed files are in the following location: `\Program Files\NetIQ\AppManager\bin`.

NOTE: For information about performing a silent installation of the Module Builder managed object, see the *Installation Guide for AppManager*.

2.4.1 Running the Setup Program on Windows Vista, Windows 7, or Windows Server 2008

The process for installing Module Builder on Windows Vista, Windows 7, or Windows Server 2008 (including R2) computers is slightly different from the installation process for other versions of Windows.

To run the setup program:

1. From the Start menu, click **All Programs**, and then click **Accessories**.
2. Right-click **Command Prompt** and select **Run as administrator**.
3. In the command prompt window, navigate to the location of the Module Builder managed object setup program.
4. At the prompt, type `AM70-ModuleBuilder-1.0.`*`xxx`*`.0.msi` and press `Enter`.

2.5 Configuring Settings for the Module Builder Editor

You can change the default configuration settings that control the behavior of the Module Builder Editor by modifying `MoBEditor.exe.config`. This configuration file is located in the `\NetIQ\Module Builder\Editor` folder.

The following table describes the configuration options stored in the `MoBEditor.exe.config` file.

Configuration Option and Default	Description
TraceLevel Default value=Warning	Specify the level of tracing detail that is added to the log file. The most detailed setting is Verbose, and the least detailed is Off. The options include Verbose, Info, Warning, Error, and Off.
TraceFileMaxLength Default value=10000000	Specify the size in bytes of each log file.

Configuration Option and Default	Description
TraceFileBackupCount Default value=10	Specify the number of log files to keep.
TraceFileAutoFlush Default value=true	Set to **true** if you want the Module Builder Editor to write the tracing text to the log file immediately, also called *flushing*, so the text is available in case of a loss of functionality. Set to **False** if you want the Module Builder Editor to write the tracing text at a later time. For either option, the tracing text will be written when the Module Builder Editor closes. Flushing the text immediately might affect the performance of the Module Builder Editor if the TraceLevel is set to Verbose or Info, where many lines of tracing text are written in quick succession.
DisplayExceptions Default value=false	Set to **true** if you want the Module Builder Editor to display a dialog box stating when an exception occurred and has been logged.
MaxLogfileReadSize Default value=210000	Specify the amount of the log file in bytes that will be shown in the Log File Preview text area on the Identify Log File Details to Monitor window in the Module Builder Editor.
MaxLogfileMatchingLinesKB Default value=20	Specify the number of KB that the Module Builder Editor should use in the `.mob` file for each log file that contains lines that match a search string. Multiple strings in any one log file will be stored together, up to this specified amount.
LogfileChunkSize Default value=500000	Specify how much of a log file, in bytes, that the Module Builder Editor should read at a time. This option allows the Module Builder Editor to read smaller chunks of a log file instead of reading the entire file all at once, which could cause performance issues with large log files.
LogfileSpecifications Default value=*.log, *.txt	Specify the default file extension shown in the Add Log File to Monitor dialog box. A user can add more types of file extensions for monitoring.
MaxChartPointsStored Default value=10	Specify the number of data points for each of the graphs stored in the `.mob` file.
MaxChartPointsDrawn Default value=10	Specify the maximum number of data points that each graph will display.
LoadAndSaveIcons	Optional key you can add to the `MoBEditor.exe.config` file. Set to **false** to prevent the Module Builder Editor from saving and loading icons for processes and applications. Too many icons in a `.mob` file may cause the Module Builder Editor to fail when attempting to open the `.mob` file. The key syntax is as follows. Add it to the `MoBEditor.exe.config` file in the <`appSettings`> section. `<add key="LoadAndSaveIcons" value="false"/>`

2.6 Discovering ModuleBuilder Resources

Use the Discovery_ModuleBuilder Knowledge Script to discover all known components for your Module Builder application on your selected computer. When AppManager discovers a component for your application, that component displays as a discovered resource for management in AppManager.

The default setting for this script is to run once.

Set the Values tab parameters as needed:

Description	How To Set It
Discovery Failure	
Event severity when discovery fails	Set the event severity level, from 1 to 40, to indicate the importance of an event that is raised when the discovery job fails. The default is 5.
Partial Discovery	
Allow partial discovery?	Select Yes to allow a partial discovery, in which some objects were discovered successfully, but others could not be discovered.The event details for a partial discovery lists the items that were not discovered. The default is Yes.
Event severity when discovery partially succeeds	If you set this Knowledge Script to allow a partial discovery, set the event severity level for a partial discovery. The default is 15.
Successful Discovery	
Raise event if discovery fully succeeds?	Select **Yes** to raise an event when the discovery process fully succeeds, without any warnings. The default is unselected.
Event severity when discovery fully succeeds	If you set this Knowledge Script to raise an event when the job fails, set the event severity level for a failed discovery. The default is 25.
Log File	**NOTE:** This section lists any log files selected in the Log Files component of the Module Builder Editor. If you did not define log files for monitoring, this section will not appear in the Discovery Knowledge Script parameters.
[Log File Name]	The name of each log file selected in the ModuleBuilder Editor is listed in this section, with the following two parameters associated with that log file.
Create log file object if file is not present	In the Module Builder Editor, the subject matter expert or the AppManager expert can set up conditions for a log file that currently does not exist. Select **Yes** to create the log file object even if the log file is not found during discovery. Clear this option if you do not want to create the log file object if the log file is not found. If the discovery process does not find a specified log file, a partial discovery notification is created. The default is Yes.
Log file path	Specify a new path for the location of the log file, as needed. You should modify this log file path on agent computers where the log file might be located in a different directory.

3 Identifying Application Conditions to Monitor

As the subject matter expert, you use the Module Builder Editor to identify the Windows components of your application that you want to manage. These components include processes, services, performance counters, event log events, or log files. After you select what you want to monitor, you determine the various conditions in those components that will generate notifications called *events* in AppManager. For more information, see the Section 3.2, “Interview Summary Window,” on page 20.

After you establish the criteria for your application, the Module Builder Editor generates a `.mob` file that the AppManager expert can configure using the Module Builder Editor. The AppManager expert then integrates the `.mob` file into AppManager as a new module designed to help manage your application. For more information, see Section 4.1, “Generating Knowledge Scripts for a New Module,” on page 37.

3.1 Launching the Module Builder Editor

Run the Module Builder Editor to provide application-specific information for the AppManager expert, who uses the information to generate customized Knowledge Scripts for AppManager. You can launch the Module Builder Editor on a local computer, or you can install and run the Module Builder Editor from a network share. You can also run it from a portable memory device, such as a memory stick.

3.1.1 Launching the Module Builder Editor Locally

Complete the following steps to launch the Module Builder on a local computer.

To launch the Module Builder Editor locally:

1. Ensure your user account is a member of the local Administrators group.
2. Launch the application you want to monitor on the same computer on which the Module Builder Editor is installed, also called the *current* computer.
3. ***If you are using Windows XP or Windows Server 2003**,* click the **Module Builder Editor** shortcut in the `NetIQ/Module Builder` Programs menu folder.
4. ***If you are using Windows Vista or Windows Server 2008***, right-click the **Module Builder Editor** shortcut in the `NetIQ/Module Builder` Programs menu folder and select **Run as administrator**.

3.1.2 Launching the Module Builder Editor from a Network Share

If you want to launch the Module Builder Editor from a share located on a different computer, configure Microsoft .NET 2.0 Framework on the computer where you created the share and on all computers where you want to launch Module Builder Editor.

To change the settings for network-hosted drive:

1. On the computer where you created the share, open **Administrative Tools** and start the **Microsoft .NET 2.0 Framework Configuration**.
2. Expand **My Computer** > **Runtime Security Policy** > **Machine** > **Code Groups**.
3. Right-click **All_Code** and select **New**.
4. Select **Create a new code group**.
5. Type a name, such as the name of the share, and then click **Next**.
6. In the **Choose the condition type for this code group** list, select **URL**.
7. In the **URL** field, type the path of the share in the following format:
   `file:////\\`*computername*`\`*sharename*`\*`

   Replace *computername* with the name of the computer where you created the share. Replace *sharename* with the name of the share
8. Click **Next**.
9. Select **Use an existing permission set**, and then select **FullTrust**.
10. Click **Next**, and then click **Finish**.
11. Restart Microsoft Internet Information Services (IIS).
12. On the computer where you created the share, launch the Module Builder Editor. For more information, see "Launching the Module Builder Editor Locally" on page 19.

3.2 Interview Summary Window

Use this window to specify the application you want to manage, and then select the Windows components you want to monitor for the application.

On the Interview Summary window, use the following panes as needed.

Pane	Description
1: Select Application to Manage	Locate the application on the current computer that you want to manage and monitor.
2: Review Results and Add Details	Set the event conditions for the processes, services, performance counters, event log events, and log files you want to monitor with AppManager. The AppManager expert at your company can help you resolve any incomplete items or settings.
3: Save Results for AppManager Expert	Save the results of your work with the Module Builder as a `.mob` file. You can then share that file with your AppManager expert.

Pane	Description
4: Generate AppManager Module (AppManager Expert mode only)	The AppManager expert reviews and edits the settings of the `.mob` file and sends the file back to the subject matter expert for more information, if needed, and then generates the customized Knowledge Scripts for the selected application. For more information about using the Module Builder Editor in AppManager Expert mode, see Section 3.11, “Changing Module Builder Editor Modes,” on page 34.

3.3 Selecting the Application to Manage

The Select Application table displays all applications on the current computer that use the Windows Installer service or an executable installation file.

NOTE: When the Module Builder Editor is in AppManager Expert mode, the Select Applications table lists only the applications installed on the subject matter expert’s computer. The AppManager expert cannot overwrite this list.

In some cases, you may notice differences between the list of applications in the Select Application table for the Module Builder Editor and the Windows Add or Remove Programs list. These variations exist because Add or Remove Programs tries to correlate registry keys with entries in the Programs menu of the Start menu. Because these correlations are not always consistent, the Module Builder Editor does not make these correlations, and as a result, all programs might not be listed. Also, some applications do not display in the Select Application table due to their installation and uninstallation methods.

TIP: The name you type in the **Application name** field of the Discover Application window is the default file name for the `.mob` file, which by default is located in the `\My Documents\Module Builder Projects` folder on the current computer. The default folder for computers running Windows Vista or Windows Server 2008 is `\Documents\Module Builder Projects`.

3.4 Identifying Processes to Monitor

Use the Module Builder Editor to select the processes you want to monitor, and then configure how you want to monitor those processes for your application. For example, you can have AppManager send an event to an AppManager administrator if a process is running or not running.

If the Module Builder Editor discovers any running processes associated with your application when you selected the application, it sets those processes to *Process should be running* by default. Process are associated with an application based on the application’s installation directory. You can change these settings later.

You can also monitor selected processes to make sure they do not use too much processing bandwidth (CPU) or memory. You can set a *threshold* amount, such as CPU usage should never be more than 20%, and if that threshold is exceeded, AppManager notifies an AppManager administrator. Threshold values are “greater than” only, not “greater than or equal to” the specified value.

NOTE: Module Builder determines the CPU utilization value by dividing the percentage of processor time by the number of CPUs. This function is the same way that Microsoft Task Manager gets CPU utilization values.

3.4.1 Identify Process Details to Monitor Window

This window allows you to specify the conditions that will result in an AppManager notification for a selected process. Complete the following fields as needed.

Field	Description
It doesn't matter	Select this option if you do not want to monitor this aspect of the process.
I don't know	Select this option if you are not sure you want to monitor this option. Your AppManager expert can help you make a decision later.
Process should be running	Select this option to have AppManager raise an event if the selected process stops running.
Process should NOT be running	Select this option to have AppManager raise an event if the selected process starts running.
How severe is it when the wrong state is encountered?	Specify a severity level for when AppManager discovers that the selected process is not in the specified normal state. For more information, see Section 3.6, "Setting Severity Levels," on page 24.
Never more than (CPU utilization)	Specify the percentage of CPU usage this process can use before it generates an event. This value is your upper threshold value. The performance counter used for this option is `Process\% Processor Time`, divided by the number of CPUs on the system. When the Module Builder Editor is in AppManager Expert mode, the chart shows historical information collected by the subject matter expert instead of live data.
Never more than (physical memory utilization)	Specify the percentage of physical memory usage this process can use before it generates an event. The performance counter used for this option is `Process\Working Set`. When the Module Builder Editor is in AppManager Expert mode, the chart shows historical information collected by the subject matter expert instead of live data.

3.5 Identifying Services to Monitor

Use the Module Builder Editor to select the Windows services you want to monitor and configure how to monitor those services for your application. For example, you can have AppManager send an event or alert if a service is running, hung, paused, disabled, not found, or recently started.

If the Module Builder Editor discovers any running, auto-started Windows services associated with your application when you selected the application, by default the Module Builder Editor specifies that AppManager sends warning notifications if that service is not running, hung, disabled or not found.

The Module Builder Editor also specifies by default that AppManager sends an informational notification if a service is shut down normally. Services are associated with an application based on the application's installation directory. You can change these settings later.

NOTE: In some instances, the executable file (.exe) that corresponds to a service can also be monitored as a process. The executable file for a service may have been added automatically as a process. If you want to monitor the CPU usage and memory for that executable file, you should also monitor the file with the Processes component in the Module Builder Editor. If you do not want to monitor the service as a process, you can remove that service using the Processes component. For more information, see Section 3.4, "Identifying Processes to Monitor," on page 21.

3.5.1 Identify Service Details to Monitor Window

This window allows you to specify the conditions that will result in an AppManager notification for a selected service. You can also specify what, if any, actions you would like to perform on the service, depending on its status.

Complete the following fields as needed.

Field	Description
Not running	If the selected service is not running, you can specify the type of notification to send to the AppManager administrator. You also have the option to start a service that is not running.
Hung	If the selected service is not responding and cannot be started or stopped, you can specify the type of notification to send to the AppManager administrator. You also have the option to terminate the service, as well as to restart the service after termination.
Paused	If the selected service is paused, you can specify the type of notification to send to the AppManager administrator. If you cannot pause the service, this option is not available.
Disabled	If the selected service is disabled, you can specify the type of notification to send to the AppManager administrator.
Shutdown normally	If the selected service shuts down normally, you can specify the type of notification to send to the AppManager administrator. You also have the option to start the service again. If you cannot stop the service, this option is not available.
Not found	If the selected service is not found, you can specify the type of notification to send to the AppManager administrator.
Started	If the selected service is started while you are monitoring it, you can specify the type of notification to send to the AppManager administrator. You also have the option to stop the started service.
Create what kind of notification?	In this column, select the kind of notification you want to send to the AppManager administrator when the service has a specific status. The notification types correspond to the severity levels used in AppManager. For more information, see Section 3.6, "Setting Severity Levels," on page 24.
Take what action on the service?	In this column, select what, if any, actions you would like to perform on the service when it has a specific status. You can select actions regardless of what kind of notification you select for the service.

NOTE: The default notifications and actions settings depend on the status of the service when it was selected. For example, if a service has a Service Status of Running and a Startup Type of Automatic, the Module Builder Editor will set the notification types to Warning notification or No notification, and the Module Builder Editor will also select the option to start a service that is not running, and terminate and then restart a service that is hung.

In addition, some of the fields on this window might be disabled based on the configuration of the Windows component, such as a service that is Not Pausable. You can edit any of these settings as needed.

3.6 Setting Severity Levels

As you set the event conditions or notification types for processes, services, performance counters, event log events, and log files, you also specify the severity level for the event. A severity level indicates the importance of an event in AppManager, on a scale of 1 to 40, with 1 being the most important.

Complete the following fields as needed.

Field	Description
No notification	Select this option or leave it at the default setting if you do not want to send an event to the AppManager administrator for this particular event condition.
I don’t know	Select this option if you are not sure you want to monitor this option. Your AppManager expert can help you make a decision later.
Diagnostic	Select this option to send an event that corresponds to a default severity of 35 in AppManager.
Informational	Select this option to send an event that corresponds to a default severity of 25 in AppManager.
Warning	Select this option to send an event that corresponds to a default severity of 15 in AppManager.
Severe	Select this option to send an event that corresponds to a default severity of 5 in AppManager.

3.7 Identifying Performance Counters to Monitor

Use the Module Builder Editor to select the performance counters you want to monitor, and then configure how to monitor those performance counters for your application. When a performance counter value goes above or below a certain amount, or threshold, AppManager generates an event.

You can also set a threshold that generates an event if a counter falls below a specified amount, or you can use a combination of high and low thresholds. Threshold values are “greater than” or “less than” only, not “greater than or equal to” or “less than or equal to” the specified value.

3.7.1 Identify Performance Counter Details to Monitor Window

This window allows you to specify the conditions that will result in an AppManager notification for a selected performance counter.

Complete the following fields as needed.

Field	Description
Units	If needed, select the unit of measurement for the performance monitor chart, such as seconds, MB, or megahertz. The chart displays the most recent performance statistics for a selected instance of a performance counter. The option you select only affects the chart, and it does *not* affect the performance counter values or the thresholds you set below. The default is `<none>`. **NOTE:** You can also create a customized unit by highlighting the text in the Units field, such as `<none>`, and then typing your preferred type of unit. Any unit you select or type for the Units field is added to the text of the related notification sent by AppManager. When the Module Builder Editor is in AppManager Expert mode, the chart shows historical information collected by the subject matter expert instead of live data.
Scale	If needed, adjust the scale of the graph so you can view the chart data more effectively. When you set upper or lower performance counter thresholds, the scale value is multiplied by your performance counter threshold when comparing it to the raw performance counter value. For example, you are monitoring a performance counter whose raw unscaled value is 5,000,000,000. To scale that value down to a more manageable number, select a Scale value of 100,000, which causes the graph to display the charted values as divided by this amount. The scaled value is 50,000. Also, if you want to set a upper threshold of 51,000, the specified threshold value will be multiplied by the Scale value (51,000 times 100,000) so it can be compared against the raw unscaled performance counter value. If your performance counter value is scaled, thresholds and collected data will use the scaled value.
It doesn’t matter	Select this option if you do not want to create a threshold for this performance counter.
I don’t know	Select this option if you are not sure you want to create a threshold for this performance counter. Your AppManager expert can help you make a decision later.
Performance counter should never be more than	Specify a numeric value to set an upper threshold. When a performance counter exceeds this value, AppManager generates an event for the AppManager administrator.
How severe is it when the counter exceeds this value?	Specify a severity level for when the performance counter exceeds the threshold. This option is required if you set an upper threshold. For more information, see Section 3.6, “Setting Severity Levels,” on page 24.

Field	Description
Enter any additional information you want to send with this notification	Specify any additional message that you want to display along with an upper threshold notification.
Performance counter should never be less than	Specify a numeric value to set a lower threshold. When a performance counter falls below this value, AppManager generates an event for the AppManager administrator.
How severe is it when the counter goes below this value?	Specify a severity level for when the counter falls below the threshold. This option is required if you set a lower threshold. For more information, see Section 3.6, “Setting Severity Levels,” on page 24.
Enter any additional information you want to send with this notification	Specify any additional message that you want to display along with a lower threshold notification.

NOTE

- You can select an upper threshold, a lower threshold, or both. Only one threshold is required
- If you added or enabled new performance counters while the Module Builder Editor was running, the Module Builder Editor will not recognize these counters until you close and then open the Module Builder Editor again.

3.8 Identifying Event Log Events to Monitor

Use the Module Builder Editor to select the Windows event log events you want to monitor, and then configure how to monitor those events for your application. For example, you can monitor the event logs for Warning or Error events specific to your application, or you could monitor the event logs for events that have a description containing the words *warning* or *error*. The Module Builder Editor presents all events the applications has registered, even if those events do not currently appear in an existing event log.

Use any of the following methods for defining event log events for monitoring:

- Browsing the existing events on the current computer
- Using event log message files, which are the Windows event message files registered on the current computer
- Manually defining the attributes for the event log event

NOTE: Module Builder does not support event log event strings that span more than one line in the log file.

3.8.1 Add Event to Monitor Window

This window allows you to select the event log events you want to monitor with AppManager.

Complete the information on the following tabs as needed.

Tab	Description
Existing Events	This tab allows you to browse and select existing event log events from a real-time viewer. To change to a different set of event log events, select a different log from the Log name list. To change the time frame for the list of events, select an option from the Show Events For Last list. When adding event log events on a computer running Windows Vista or Windows Server 2008, the user group might need to be specified. For example, specifying the `SYSTEM` user by itself in the **User** field might not yield any matches even if matching events are present. You must add the group name, such as `SYSTEM`, before a match is displayed. After you add an event using this tab, the Module Builder Editor populates most of the fields with the corresponding event properties. You can modify or unselect fields such as **User** and **Computer** to make them less specific; if not, you may miss important event log data. This tab does not display if the Module Builder Editor is in AppManager Expert mode.
Application-specific Event Messages	This tab allows you to select from a list of registered application event messages that may be similar to your application. Select **Show all registered event log applications** to view all event messages that have been registered.
Ad-hoc Events	This tab allows you to specify the event conditions for a log event. To change the event log type, select a different log from the Log name list. Select at least one of the event properties from the list and specify the details for the event property. You can use regular expressions for the event properties; for more information, see Section 3.10, “Searching for Text with Regular Expressions,” on page 32.

Related Topics

- Section 3.8.2, “Identify Event Log Events to Monitor Window,” on page 27
- Section 3.10, “Searching for Text with Regular Expressions,” on page 32

3.8.2 Identify Event Log Events to Monitor Window

This window allows you to specify the conditions that will result in an AppManager notification for a selected event log event.

NOTE

- If the event logs on the current computer are large or numerous, the value in the Number of Occurrences column might display Calculating for a period of time as the values are collected.

- Click **Refresh** to recalculate the number of matching event log entries per filter in the Events to Monitor table. This information is updated in the Number of Occurrences column of the Events to Monitor table.
- The Refresh option does not work in AppManager Expert mode.

Complete the following fields as needed.

Field	Description
How should matching events be referenced?	If needed, provide a new label for the event.
Event ID	The event number that identifies the event type.
Description contains	A particular string of text or keywords in the data.
Event type	The type of event includes Information, Warning, Error, Critical, Audit Success (shown in Windows XP and Windows Server 2003 as Success Audit), or Audit Failure (shown in Windows XP and Windows Server 2003 as Failure Audit).
Source	The source of the event. This can be the name of a program, a system component, or an individual component of a large program.
User	The account name associated with the event. Specify an appropriate search string, such as `<domain name>\<user name>`. The Knowledge Script will look for matching entries in the Event Log's **User** field.
Computer	The name of the computer where the event occurred.
Category	The further classification of the event by the event source, such as Server or Logon.
Data contains	Specify a particular number of bytes in hexadecimal. This field is useful only if the event contains binary data. A match occurs if any portion of the event contains the specified value. In other words, this match is not a literal match.
How severe are events that match these properties?	Select a severity level for when the event message criteria are met. This option is required. For more information, see Section 3.6, "Setting Severity Levels," on page 24.

NOTE: For all of the condition fields except for **Data contains**, you can add regular expressions instead of text. After you type your regular expression search criteria, select **Regular expression**. For more information, see Section 3.10, "Searching for Text with Regular Expressions," on page 32.

3.9 Identifying Log Files to Monitor

Use the Module Builder Editor to choose which text log files you want to monitor for your application. For example, you can monitor log files for specific strings of text that the application writes to the file when a critical event or an error occurs, and you can then set up AppManager events based on this information.

You can monitor log files that are specific to your application, log files that are on the hard drive of the current computer, or you can monitor a log file that currently does not exist, with the expectation that AppManager will send an event when that log file is created.

If the Module Builder Editor discovers any log files in the installation folder for your application when you identified the application, the Module Builder Editor creates a default search string for the word *error* in those log files. You can change these default settings later.

NOTE: Module Builder does not support regular-expression search strings that span more than one line in the log file.

3.9.1 Add Log File to Monitor Window

This window allows you to select the log files you want to monitor with AppManager.

Complete the information on the following tab as needed.

Tab	Description
Application-specific	This tab lists the log files found in the installation folder for the your application.
On Disk	This tab allows you to browse and add relevant log files from the current computer. Click Up to navigate to a higher folder level, or click Browse to navigate to a specific folder. This tab is not available in AppManager Expert mode.
Non-existent	This tab allows you to add a log file for monitoring that currently does not exist, but is created in certain circumstances. When the log file is created, you can then specify that AppManager send an event to the AppManager administrator. You can specify the same monitoring options for this type of log file. For example, for the **Folder** field, you might browse to the folder where other logs are located for your application, and in the **File name** field, you can type the file name for a log file similar to the other log files in the folder. You can also use a wildcard, such as `myapp*.log`. In this case, the `*` represents a number for a log file that is incremented when it reaches a certain size, such as `myapp3.log` or `myapp4.log`. The **Create a wildcard specification based on the selected log file** option is not available on this tab.

TIP

- To expand your log file search parameters, click Modify File Specifications and add more file types to the list of log files.
- To add a group of log files with similar names, select a log file and select **Create a wildcard specification based on the selected log file**. For more information, see Section 3.9.2, “Change Wildcard Specification Window,” on page 30.

3.9.2 Change Wildcard Specification Window

You can create a wildcard log file specification that allows you to select multiple log files with similar names. For example, if you type `MoBEditor*.txt`, the Module Builder Editor selects all files that started with `MoBEditor` and ended with `.txt`.

To create wildcard specifications for log files:

1. From the list of log files on the Identify Log File Details to Monitor window, select a log file and click **Change Wildcard Specification**.

   TIP: You can also select a log file on the Add Log File to Monitor window, by selecting **Create a wildcard specification based on the selected log file** and clicking **OK**.

2. Specify the name of the related log files you want to monitor, using the `?` or `*` as wildcards.
3. Click OK.

3.9.3 Identify Log File Details to Monitor Window

This window allows you to select a log file event and specify the conditions that will result in an AppManager event.

TIP: Click **Refresh** to recalculate the number of matches in the Number of Matching Lines column for each search word per log file.

Complete the following fields as needed.

Field	Description
Only show lines that contain a search string	Select this option to preview the search strings you have created for this log file. The default setting for this option is unselected, which shows the entire contents of the log file in the preview pane.
Log file encoding	Select a different option from this list if you want to change how the log file is interpreted. If you can't read the log file data, try a different type of encoding until you can read it. The default is UTF-8.
Word Wrap	Toggle the option to make the file easier to read in the text box below this field.
Zoom	Use this tool to change the size of the text in the text box below.
It doesn't matter	Select this option if you do not want to search for a particular word or string in this log file.
I don't know	Select this option if you are not sure you want to search for a particular word or string in this log file. Your AppManager expert can help you make a decision later.
The following words	Specify that you want search for a specific string of text in a log file. Click **Add Word** to define the string of text. For more information, see Section 3.9.4, "Add Log File Search Word Window," on page 31.
Log file should NOT exist	Specify that you want AppManager to raise an event when your application creates this log file.
Log file should always exist	Specify that you want AppManager to raise an event when this log file does not exist.
How severe is it when the wrong state is encountered?	Specify a severity level for when the state for the log file either exists or does not exist incorrectly. For more information, see Section 3.6, "Setting Severity Levels," on page 24.

- Section 3.9.4, "Add Log File Search Word Window," on page 31

3.9.4 Add Log File Search Word Window

This window allows you to specify that AppManager send an event if it finds a specific search word or string in a log file.

Complete the following fields as needed.

Field	Description
Search word(s)	Type the text you want to find, such as the words `error` or `warning`. You can also type a regular expression in this field. This field is required.
Match case	Select this option to make the search case-sensitive. The default is unselected.
Use Regular Expression	Select this option to search using a regular expression. The default is unselected. For more information, see Section 3.10, "Searching for Text with Regular Expressions," on page 32.

Field	Description
Severity of error	Select an option from this list to specify the severity of the notification that is sent in AppManager when your specified search word or words is encountered. This field is required. For information, see Section 3.6, “Setting Severity Levels,” on page 24.
No additional text	Select this option if you do not want to include any more text after the search string when reporting your search results.
The entire line	Select this option to include the entire line of text that contains the search string when reporting your search results.
The rest of the line	Select this option include the rest of the line of text that comes after the search string when reporting your search results.
Surrounding characters	Select this option and then specify the number of characters before or after the search string to include when reporting your search results.

Related Topics

- “Add Log File to Monitor Window” on page 29
- “Identify Log File Details to Monitor Window” on page 30
- “Add Log File to Monitor Window” on page 29
- Section 3.9.4, “Add Log File Search Word Window,” on page 31

3.10 Searching for Text with Regular Expressions

You can perform advanced text pattern searching using *regular expressions*. Regular expressions describe a specific portion of text for which you want the Module Builder Editor to search in an application’s event log entries or log files. The Module Builder Editor uses the Microsoft .NET Framework Software Developer’s Kit (SDK) for processing regular expressions.

The following table lists regular expression operators and their definitions.

Operator	Definition
\	Matches the exact character following it.
.	Matches any single character except line-break characters.
[ ]	Matches any single character from within the bracketed list.
^	Matches a position before the first character in a string.
$	Matches right after the last character in a string.
|	Matches either the regular expressing preceding it or the regular expression following it.
( )	Groups one or more regular expressions together logically.
?	Indicates that the preceding character in the expression is optional.
*	Functions as a wildcard in the regular expression string to omit some variable nature of the event log event or the log file.
!	Indicates that the regular expression following it should *not* match.

Operator	Definition
`\d`	Matches any digits 0-9.
`{n}`	Specifies that the preceding regular expression is matched exactly n number of times.

3.10.1 Regular Expressions for Event Log Events

You can use regular expressions in almost all of the event log fields on the Identify Event Log Events to Monitor window. The following table lists some sample regular expressions you can use to filter the data in the event log events.

Example	Matches

`^2\d{3}$`	`2000`
	`2021`
	`2999`

`^(5084|5603|8128)$`	`5084`
	`5603,8128`

`.*User.*`	`UserEnv` `RemoteUser`

`^(?!CORPSERVER)`	`01CORPSERVER`
	`SERVER02`

`winlogon\.exe. *Upgrade.*Windows setup has completed`	`The process winlogon.exe has initiated the restart of computer RALAMDCV04 on behalf of user NT AUTHORITY\SYSTEM for the following reason: Operating System: Upgrade (Planned)`
	`Reason Code: 0x80020003`
	`Shutdown Type: restart`
	`Comment: Windows setup has completed, and the computer must restart.`

3.10.2 Regular Expressions for Log Files

On the Add Log File Search Word window, you can use regular expressions to create very specific search strings for log files.

The following table lists some sample regular expressions you can use to search for data in log files.

Example	Match
`error:(1[0-9]|20)\s`	`The error:15 has occurred`
`^Error`	`Error in filename.`

Example	Match
`alert.*system fault`	`A subsystem alert was raised indicating a system fault in the health monitoring process.`
`postmaster@(domain-one\.local|domain-two\.local|domain-three\.local)`	`A system generated message was sent to postmaster@domain-one.local.`

3.11 Changing Module Builder Editor Modes

The Module Builder Editor uses the following modes:

Subject Matter Expert mode

Used by the subject matter expert who will identify and set up the various conditions for the application that will be monitored. For more information, see Section 3.11.1, “Using the Subject Matter Expert Mode,” on page 34.

AppManager Expert mode

Used by the AppManager expert who will deploy AppManager modules based on information from the Module Builder Editor. This mode does not display live data, but instead shows data from the time frame in which the subject matter expert created the `.mob` file. For more information, see Section 3.11.2, “Using the AppManager Expert Mode,” on page 34.

3.11.1 Using the Subject Matter Expert Mode

The default setting for the Module Builder Editor is Subject Matter Expert mode. However, the Module Builder Editor remembers the mode setting. If you change to AppManager Expert mode and close the Module Builder Editor, the next time you launch the Module Builder Editor, the mode setting will be AppManager Expert mode.

Enable the Subject Matter Expert mode when by clicking Hide AppManager Expert Mode on the ribbon. Subject Matter Expert mode should be used by the subject matter expert who will identify and set up the various conditions for the application that will be monitored.

While you are in Subject Matter Expert mode, the Module Builder Editor actively collects new data from the current computer and stores it in the `.mob` file. If you open an existing `.mob` file, the Module Builder Editor starts collecting new data from the various Windows components, such as current service status and performance counter data. The Module Builder Editor could potentially overwrite existing data in this file as a result.

3.11.2 Using the AppManager Expert Mode

AppManager Expert mode should be used by the AppManager expert who will deploy AppManager modules based on information from the Module Builder Editor that has been provided by an application subject matter expert.

Enable the AppManager Expert mode by clicking Show AppManager Expert Mode. If you open a `.mob` file from a previous AppManager Expert mode session with the Module Builder Editor, the Module Builder Editor remembers the mode that was used.

While you are in AppManager Expert mode, the Module Builder Editor does not actively collect new data from the current computer. When you access a Module Builder file in AppManager Expert mode, you will only see the data that the Module Builder Editor collected while in Subject Matter Expert mode.

As a result, process and performance counter charts will not gather new data while you are in AppManager Expert mode. The lists of processes, services, log files, and other data are all specific to the computer that was used to gather information on the application while in Subject Matter Expert mode. Also, clicking **Refresh** will not update the data while you are in AppManager Expert mode.

4 Module Builder Knowledge Scripts

As the AppManager expert, you work with the custom AppManager Knowledge Scripts created by the subject matter expert with the Module Builder Editor. These scripts allow you to monitor the processes, services, performance counters, event log events, and log files that are critical to the performance of the managed application.

Before you can run the Knowledge Scripts for your new module, install the Module Builder managed object, and then generate the custom Knowledge Scripts. For more information, see Section 4.1, "Generating Knowledge Scripts for a New Module," on page 37.

A separate Module Builder Knowledge Script group exists for each module you generated with the Module Builder Editor. For more information, see Section 4.2, "Working with Module Builder Knowledge Scripts," on page 39.

Depending on the settings you and the subject matter expert specified in the Module Builder Editor, Module Builder creates one or more of the following Knowledge Scripts:

Knowledge Script	What It Does
EventLogCheck	Monitors a set of event log events for the application.
LogFileCheck	Monitors a set of log files for the application.
PerformanceMetrics	Monitors the status of the performance counters selected for the application.
ProcessHealthCheck	Monitors the health and status of processes selected for the application.
ServiceHealthCheck	Monitors the health and status of services related to the application.

NOTE: To view generic Help for Module Builder Knowledge Scripts, click **Help > Help Topics**, and then go to the Knowledge Script Reference folder on the Contents tab and select the **Module Builder Knowledge Scripts** folder.

4.1 Generating Knowledge Scripts for a New Module

As the AppManager expert, your responsibilities begin when you receive the `.mob` file created with the Module Builder Editor by the subject matter expert. Install the ModuleBuilder managed object files on each agent, console, and repository computer.

After reviewing the `.mob` file in the Module Builder Editor and making any changes needed to the settings and conditions, use the `.mob` file to generate the new module and its Knowledge Scripts.

The generation process also creates an application monitoring contract, a PDF file that describes the Windows components and settings specified by the subject matter expert in the Module Builder Editor. The Knowledge Scripts and the contract can only be generated if all the relevant settings are complete, with no required settings left as *I don't know* and no required severity levels left undefined.

You need Adobe Acrobat Reader or another PDF reader application to read the PDF file, which uses the following naming convention: `ModuleBuilder-[`*ApplicationName*`]_Agreement.pdf`.

NOTE

- You do not need to open the `.mob` file on the same computer used by the subject matter expert when the `.mob` file was created.
- Data gathered on the subject matter expert's computer is displayed in the Module Builder Editor, but no data is gathered from your computer while you are in AppManager Expert mode.

To generate scripts for a new module:

1. Double-click the `.mob` file to open the Module Builder Editor.
2. If the Module Builder Editor does not open in AppManager Expert mode, click **Show AppManager Expert Mode**.
3. Review the contents of the `.mob` file by clicking **Configure** for each component on the Review Results and Add Details pane of the Interview Summary window.
4. Where needed, update the `.mob` file if information is incorrect or missing, including places where the subject matter expert left an option set to *I don't know*. Send the file back to the subject matter expert for more information, if needed.
5. In the Generate AppManager Module pane of the Interview Summary window, click **Generate**.
6. ***If you want to change the default severity settings for all generated Knowledge Scripts***, edit the four severity levels as needed. The smaller the number, the higher the importance in AppManager.
7. ***If you want to modify the default KS Category name for the generated Knowledge Scripts***, specify the name in the **KS Category** field. The default name is "ModuleBuilder".
8. ***If you want to view a list of changes made to this file****,* click **View Change History**. Edits made by the subject matter expert are listed under the *changes by SME* section of the `.txt` file, while edits by the AppManager expert are listed under *changes by AME*.
9. Click **Generate Module** and select the folder where you want to place the generated scripts and the monitoring contract.

   NOTE: If you are generating a module for an application you previously used with the Module Builder Editor, the Module Builder Editor will overwrite any existing Knowledge Scripts in the default folder. If you do not want to overwrite existing scripts, navigate to a different folder or click **Make New Folder.**

10. Click **OK** to close the Module Generation Complete dialog box.

    TIP: If the generation process generated any errors and you want to copy the error information into another application, press **Ctrl** while you select the errors, and then press **Ctrl+C**.

11. Click **Finish**.
12. Copy the folder containing the newly generated scripts from the `\Module Builder Projects\` folder onto a shared drive or a removable drive, or email the folder and its contents to the console computer where AppManager is installed. If you are running the Module Builder Editor and AppManager on the same computer, you can skip this step.
13. On the console computer where AppManager is installed, paste the new Module Builder folder and its contents into the `\NetIQ\AppManager\qdb\kp` folder.
14. In AppManager, check in the Knowledge Scripts. For more information about checking in scripts, see the user guides for Operator Console or Control Center.
15. At this point, the Module Builder custom module functions just like any other AppManager module. You can add computers to monitor and run the discovery process as needed.

NOTE: If the subject matter expert makes any changes to the module settings in the ModuleBuilder Editor, launch the Module Builder Editor and generate the scripts again using the above process. Check in the new or updated Knowledge Scripts, run the discovery process again, and update any existing jobs as well.

4.2 Working with Module Builder Knowledge Scripts

The `Discovery_ModuleBuilder-[`*ApplicationName*`]` appears in the Discovery Knowledge Script group, where `[`*ApplicationName*`]` is the application name given by the subject matter expert in the Select Application process of the Module Builder Editor.

The names for custom Knowledge Scripts created with the Module Builder Editor are structured like this: `ModuleBuilder-[`*ApplicationName*`]_[`*ScriptName*`]`, where `[`*ScriptName*`]` is the name of the specific Knowledge Script.

The name for a Module Builder Discovery Knowledge Script is structured like this: `Discovery_ModuleBuilder-[`*ApplicationName*`]`.

The names of the objects in the TreeView are structured like this: `ModuleBuilder-[ApplicationName]:[`*ComputerName*`]`, where `[`*ComputerName*`]` is the name of the computer on which the Module Builder Editor was run.

NOTE

- When deploying a Module Builder Knowledge Script on one or more computers that have more than one unique Module Builder application, the script automatically associates all Module Builder application objects, not just the objects based on the script. When deploying a Module Builder script in this scenario, use the Objects tab to unselect Module Builder applications that are not applicable.
- For monitoring policies, AppManager automatically creates jobs without control over the object matching, which could lead to failure events, such as job type mismatches.
- For rule-based management groups, you cannot use the Compare Object with Name object rule because of the way in which Module Builder objects are named. The unique portion of the object name will be to the right of the colon. As a result, you must instead use a rule-based management group, such as Match Detail.

4.3 Revising an Existing Module Builder Custom Module

As the AppManager expert, you can revise the settings or Windows components selected in the Module Builder Editor if those settings are not sufficient or accurate. You can also revise the custom module when you upgrade to a newer release of the managed application.

When a subject matter expert or an AppManager expert updates a module and the AppManager expert regenerates the resulting Knowledge Scripts, the new scripts must be checked in, a new discovery must occur, and any existing jobs should be updated.

To revise an existing module:

1. Decide if you want to create a new Module Builder custom module with its own set of Knowledge Scripts and object tree in the TreeView, or if you want to use the existing application and preserve currently running jobs and existing data streams.

2 ***If you want to create a new custom module with its own set of Knowledge Scripts***, complete the following steps:

   2a Use the Module Builder Editor to specify the application and the application components to monitor.

   2b Generate the new Knowledge Scripts.

   2c In AppManager, stop and existing jobs running with the old version of the module.

   2d Check in the new scripts.

   2e Run discovery and deploy new jobs.

3 ***If you want to use the existing Knowledge Scripts***, complete the following steps:

   3a Use the Module Builder Editor to update the existing `.mob` file or simply create a new `.mob` file with the same application name as the old version.

   3b Generate the new Knowledge Scripts.

   3c In AppManager, check in the new scripts.

   NOTE: When you revise an existing .mob file, if you change the Application name field on the Discover Application Properties dialog box for the Select Application to Manage process, you get a new object tree and a new set of Knowledge Scripts.

4 For jobs that are currently running with the previous version of the module, you need to propagate the updated Knowledge Scripts. For more information, see the “Running Monitoring Jobs” chapter of the *AppManager Operator Console User Guide*.

4.4 Using the Browse Button to Set Parameters

Some Knowledge Script parameters can only be set by clicking the **Browse (...)** button in the Value column. Clicking **Browse (...)** launches a dialog box that lists each item individually. The dialog box allows you to edit the parameter-specific settings for each item or items to be monitored. This option is also called the Knowledge Script parameter extension.

If you want to override the settings created by the subject matter expert with the Module Builder Editor, you can change the values in the Browse dialog box as needed.

Depending on the Knowledge Script parameter, you will encounter one of the following sets of options in the resulting dialog box:

- If you want to raise an event if a certain condition is met, select **Yes** in the Raise Event? column for each item in the list you want to update. Select **No** if you do not want to raise an event.
- If you want to change the default severity level, type a new severity level number in the Severity column for each item in the list you want to update.
- If you want to edit the threshold amounts or units for a process, type a new number in the Threshold column or select an option from the Unit column for each item in the list you want to update.
- If you want to edit the threshold amounts for a performance counter, specify the new amounts for each item in the list you want to update. The Unit and Scale values are listed in the dialog box, but you cannot edit these parameters.

TIP: When relevant, click **Apply default to All** to change all parameters to the value set in the Default row. The default setting for each object is determined by the settings selected by the subject matter expert in the Module Builder Editor.

4.5 EventLogCheck

This automatically generated Knowledge Script monitors a set of event log events for the application, based on the criteria set up in the Module Builder Editor. With this Knowledge Script you can track Windows event log entries that match a filtering criterion.

You can set AppManager to send an event or alert if the specified text or conditions appear in an event log, and then collect data on those events.

NOTE

- The EventLogCheck script uses the local repository on the agent computer to store the last scanned information. As a result, locally stored data for a job persists unless you remove it from the agent. If you stop an existing EventLogCheck job and then restart it later, you might not receive an event, and your data stream value might be set to zero. EventLogCheck only looks for new entries based on the last time the log file was analyzed. Because the job was an existing job, the last scan time was saved on the agent computer being monitored. When the job was restarted, the last scan time was used as the starting point.
- If you change the Scan back parameter for an active running EventLogCheck job, you might not get an event and your data stream value could be set to zero. By default EventLogCheck will only look for new entries based upon the last time the event log was analyzed. Because the job was an existing job, the last scan time was saved on the agent computer being monitored. When the job was restarted, the last scan time was used as the starting point.

4.5.1 Resource Object

Event Log object

4.5.2 Default Schedule

The default interval for this script is every 30 minutes.

4.5.3 Setting Parameter Values

Set the following parameters as needed:

Description	How to Set It
Event Log	
Scan back event log N hours on first iteration	Set this parameter to control checking for the first interval (after which checking is incremental): ◆ **-1** for all the existing entries ◆ **N** for the past n hours (8 for the past 8 hours, 50 for the past 50 hours, etc.) ◆ **0** for no previous entries (only search from this moment on) The default is 0. **NOTE:** This parameter value is only used on the first iteration of the job. If you stop the job and then start it again, this parameter will not be considered.

Description	How to Set It
Threshold: Maximum number of matching entries	Click **Browse (...)** and specify the threshold for the number of matching entries for each event log. The default is 1.
Maximum number of entries per event report	Specify the maximum number of entries to be recorded in each event's detail message. If the Knowledge Script finds more entries from the log than can be put into one event message, it will return multiple events to report all the outstanding entries in the log. The default is 30 entries.
Data Collection	
Collect data for event log?	Click **Browse (...)** and select **Yes** for each event for which you want to collect data. If enabled, data collection returns the number of matches found during the monitoring period.The default is No.
Maximum number of entries in data detail	Specify the maximum number of event log entries you want to list in the data detail. The data stream value will have the total count, but this parameter will limit the number of matches that are listed in the data detail. The default is 100 entries.
Event log data stream legend	Edit the name of the legend for the event log data stream. The default text is: "Event [EVENT_DESCRIPTION]". [EVENT_DESCRIPTION] will be substituted with the event filter description specified in the Module Builder Editor.
Event Notification	
Event Found in Event Log	
Raise event if entry is found in event log?	Click **Browse (...)** and select **Yes** for each event for which you want to raise an AppManager event when an event condition is met. The default is based on the options selected in the Module Builder Editor.
Event severity when entry is found in event log	Click **Browse (...)** and specify the event severity for each event in which an entry is found in the event log. The default severity level is based on the options selected in the Module Builder Editor.
Event message when entry is found in event log	Edit the event message text used when an entry is found in the event log. The default text is: "Event [EVENT_DESCRIPTION] was found in the [EVENT_LOG] event log". [EVENT_DESCRIPTION] will be substituted with the event filter description specified in the Module Builder Editor, and [EVENT_LOG] will be substituted with the Windows Event Log associated with the event filter (such as Application, System, or Security).
Additional Settings	
Event Details	
Event detail format	Specify how you want the event detail information formatted. Your options include: ◆ **HTML Table**: Displays the information in an HTML-formatted table. ◆ **Plain Text**: Displays the information in a table that uses plain text. The default is HTML Table.
Job Timeout	
Elapsed time for job timeout	Specify the length of system inactivity that designates a job timeout. The default is 30 minutes.

Description	How to Set It
Event severity when job timeout occurs	Specify the event severity, between 1 and 40, to indicate the importance of an event raised when a job timeout occurs. The default is 10.
Event message when job timeout occurs	Edit the text of the job timeout message. The default text is: “Job timeout”.
Job Failure Event Notification	
Event severity when job fails	Set the event severity level, from 1 to 40, to indicate the importance of an event that is raised when the job fails. The default is 10.

4.6 LogFileCheck

This automatically generated Knowledge Script monitors a set of text log files for the application, based on the criteria set up in the Module Builder Editor.

You can use this Knowledge Script to raise AppManager events or notifications for the following situations:

- If a log file is found
- If a log file is not found
- If a specified string is found in a log file

LogFileCheck can also collect data for a specified log file after the log file has been found.

NOTE

- The LogFileCheck script uses the local repository on the agent computer to store the last scanned information. As a result, locally stored data for a job persists unless you remove the data from the agent. If you stop the LogFileCheck job, and then start it again later, the monitoring continues from where it left off.
- If you stop a LogFileCheck job and then restart it later, you might not receive an event, and your data stream value might be set to zero. LogFileCheck only looks for new entries based on the last time the log file was analyzed. Because the job was an existing job, the last scan time was saved on the agent computer being monitored. When the job was restarted, the last scan time was used at the starting point.

4.6.1 Resource Object

Log Files object

4.6.2 Default Schedule

The default interval for this script is every hour.

4.6.3 Setting Parameter Values

Set the following parameters as needed:

Description	How to Set It
Log File Found	
Data Collection	
Collect data for log file found?	Click **Browse (...)** and select **Yes** for all log files for which you want to collect data. If enabled, data collection returns the following: ◆ 0 - the log file is not found ◆ 100 - the log file is found. The default is No.
Log file found data stream legend	Edit the name of the legend for the log file found data stream. The default text is: “Log File [OBJECT]”. [OBJECT] will be substituted with the name of the log file.
Event Notification	
Log File Found	
Raise event if log file found?	Click **Browse (...)** and select **Yes** for the log files for which you want to raise an event if the log file is found. The default is based on the options selected in the Module Builder Editor.
Event severity when log file is found	Click **Browse (...)** and set the event severity for when a log file is found. The default severity level is based on the options selected in the Module Builder Editor.
Event message when log file is found	Edit the event message text used when a log file is found. The default text is: “Log File [OBJECT] was found”. [OBJECT] will be substituted with the name of the log file.
Log File Not Found	
Raise event if log file is not found?	Click **Browse (...)** and select **Yes** for all the log files for which you want to raise an event if the log file is not found. The default is based on the options selected in the Module Builder Editor.
Event severity when log file is not found	Click **Browse (...)** and set the event severity for when a log file is not found. The default severity level is based on the options selected in the Module Builder Editor.
Event message when log file is not found	Edit the event message text used when a log file is not found. The default text is: “Log File [OBJECT] was not found” [OBJECT] will be substituted with the name of the log file.
String Found in Log File	
Scan entire file or files on first iteration?	Click **Browse (...)** and select **Yes** for all the log files you want Module Builder to completely scan the first time you run the this Knowledge Script. If you have a large number of log files to scan, you might want to select No to avoid any potential performance issues on the first iteration. The default is No.

Description	How to Set It
Rescan updated files?	Click **Browse (...)** and select **Yes** for the log files you want to completely rescan. When set to No, each monitored log file that changes during the monitoring interval will be scanned for new entries from the last scan position. When set to Yes, the *entire* file will be rescanned for matching entries when the file changes. The default is No.
Display additional log file text with data and event?	Click **Browse (...)** and select **Yes** for all the log files for which you want to display additional log file text with data and event. The additional log file text could be the entire line of log file data that includes the search string, the rest of the line, or the surrounding characters. This option is specified in the Module Builder Editor.
Data Collection	
Collect data for string found in log file?	Click **Browse (...)** and select **Yes** for all the log files for which you want to collect data for when Module Builder finds that log file. The default is No.
Maximum number of entries in data detail	Specify the maximum number of log files you want to list in the data detail. The data stream value will have the total count, but this parameter will limit the number of matches that are listed in the data detail. The default is 100.
String found in log file data stream legend	Edit the name of the legend for the data stream created when Module Builder finds a log file with the specified text string. The default text for the legend is: “String [SEARCH_STRING] [CASE_SENSITIVE] [REGULAR_EXPRESSION] in log file [OBJECT]”. [SEARCH_STRING] will be substituted by the search word or words specified in the Module Builder Editor. If the *Match Case* option in the Module Builder Editor was checked, [CASE_SENSITIVE] will be replaced by “Case Sensitive” in the legend. If the *Use Regular Expression* option in Module Builder is checked, [REGULAR_EXPRESSION] will be replaced by “Uses Regular Expression” in the legend. [OBJECT] will be substituted with the name of the log file.
Event Notification	
Raise event if string is found?	Click **Browse (...)** and select **Yes** for all the log files for which you want to raise an event if the search string is found. The default setting is based on the options selected in the Module Builder Editor.
Maximum number of entries in event detail	Specify the maximum number of log files you want to list in the event detail. The data stream value will have the total count, but this parameter will limit the number of matches that are listed in the event detail. The default is 100. **NOTE:** The matches are listed from newest to oldest.
Event message when string is found	Edit the event message text used when the search string is found. The default text for the legend is: “New [SEARCH_STRING] [CASE_SENSITIVE] [REGULAR_EXPRESSION] entry was found in log file [OBJECT]”. [SEARCH_STRING] will be substituted by the search word or words specified in the Module Builder Editor. If the *Match Case* option in the Module Builder Editor was checked, [CASE_SENSITIVE] will be replaced by “Case Sensitive” in the legend. If the *Use Regular Expression* option in Module Builder is checked, [REGULAR_EXPRESSION] will be replaced by “Uses Regular Expression” in the legend. [OBJECT] will be substituted with the name of the log file.

Description	How to Set It
Additional Settings	
Event Details	
Event detail format	Specify how you want the event detail information formatted. Your options are: ◆ **HTML Table**: Displays the information in an HTML-formatted table. ◆ **Plain Text**: Displays the information in a table that uses plain text. The default is HTML Table.
Job Timeout	
Elapsed time for job timeout	Specify the length of system inactivity that designates a job timeout. The default is 30 minutes.
Event severity when job timeout occurs	Specify the event severity for when a job timeout occurs. The default is 10.
Event message when job timeout occurs	Edit the text of the job timeout message. The default text is: “Job timeout”.
Job Failure Event Notification	
Event severity when job fails	Set the event severity level, from 1 to 40, to indicate the importance of an event that is raised when the job fails. The default is 10.

4.7 PerformanceMetrics

This automatically generated Knowledge Script monitors a set of performance counters for the application, based on the criteria set up in the Module Builder Editor.

You can use this Knowledge Script to raise AppManager events for the following situations:

- If the specified counter exceeds a threshold
- If the specified counter falls below a threshold

PerformanceMetrics can also collect data for the specified performance counter.

TIP: To create a dynamic view based on the name of a performance counter that has special characters such as \ or % in its name, use the wildcard symbol * in place of the special character.

4.7.1 Resource Object

Performance Counters object

4.7.2 Default Schedule

The default interval for this script is five minutes.

4.7.3 Setting Parameter Values

Set the following parameters as needed:

Description	How to Set It
Performance Counter	
Data Collection	
Collect data for performance counter?	Click **Browse (...)** and select **Yes** for the performance counters for which you want to collect counter value data, modified by the Scale value selected in the Module Builder Editor, where relevant. The default is No.
Performance counter data stream legend	Edit the name of the legend for the performance counter data stream. The default text is: “Performance counter [PERFORMANCE_COUNTER]”. [PERFORMANCE_COUNTER] will be substituted with the name of the performance counter.
Event Notification	
Counter Exceeds Threshold	
Raise event if counter exceeds threshold?	Click **Browse (...)** and select **Yes** for all the performance counters for which you want to raise an event if the threshold is exceeded. The default setting is based on the options selected in the Module Builder Editor.
Threshold: Maximum performance counter value	Click **Browse (...)** and edit the threshold amount for the selected performance counters. The default threshold is based on the options selected in the Module Builder Editor.
Event severity when counter exceeds threshold	Click **Browse (...)** and edit the severity level for when a selected performance counter exceeds the threshold. The default severity level is based on the options selected in the Module Builder Editor.
Event message when counter exceeds threshold	Edit the name of the legend for the performance counter data stream for when the counter exceeds the threshold. The default text is: “Performance Counter [PERFORMANCE_COUNTER] exceeds threshold”. [PERFORMANCE_COUNTER] will be substituted with the name of the performance counter.
Counter Below Threshold	
Raise event if counter falls below threshold?	Click **Browse (...)** and select **Yes** for all the performance counters for which you want to raise an event if the counter falls below the threshold. The default setting is based on the options selected in the Module Builder Editor.
Threshold: Minimum performance counter value	Click **Browse (...)** and edit the threshold amount as needed for the selected performance counters. The default threshold is based on the options selected in the Module Builder Editor.
Event severity when counter falls below threshold	Click **Browse (...)** and edit the severity level for when a selected performance counter falls below the threshold. The default severity level is based on the options selected in the Module Builder Editor.
Event message when counter falls below threshold	Edit the name of the legend for the performance counter data stream for when the counter falls below the lower threshold. The default text is: “Performance Counter [PERFORMANCE_COUNTER] falls below threshold”. [PERFORMANCE_COUNTER] will be substituted with the name of the performance counter.

Description	How to Set It
Additional Settings	
Event Details	
Event detail format	Specify how you want the event detail information formatted. Your options include: ◆ **HTML Table**: Displays the information in an HTML-formatted table. ◆ **Plain Text**: Displays the information in a table that uses plain text. The default is HTML Table.
Job Timeout	
Elapsed time for job timeout	Specify the length of system inactivity that designates a job timeout. The default is 5 minutes.
Event severity when job timeout occurs	Specify the event severity for when a job timeout occurs. The default is 10.
Event message when job timeout occurs	Edit the text of the job timeout message. The default text is: “Job timeout”.
Job Failure Event Notification	
Event severity when job fails	Set the event severity level, from 1 to 40, to indicate the importance of an event that is raised when the job fails. The default is 10.

4.8 ProcessHealthCheck

This automatically generated Knowledge Script monitors the status, memory usage, and CPU usage of selected processes for your application, based on the type of information gathered in the Module Builder Editor.

You can use this Knowledge Script to raise AppManager events for the following situations:

- If the process is running, not running, no longer running, or recently started
- If the process exceeds a certain level of physical memory usage
- If the process exceeds a certain level of CPU usage

ProcessHealthCheck can also collect data for the process state, for CPU usage for the process, and for memory utilization for the process.

4.8.1 Resource Object

Processes object or Processes folder

4.8.2 Default Schedule

The default interval for this script is every five minutes.

4.8.3 Setting Parameter Values

Set the following parameters as needed:

Description	How to Set It
Process State	
Data Collection	
Collect data for the state of the process?	Click **Browse (...)** and select **Yes** for each process for which you want to gather process state data. The default is No.
Process state data stream legend	Edit the name of the legend for the process state data stream. The default text is: “Process [OBJECT]”. [OBJECT] will be substituted with the name of the process.
Event Notification	
Process Not Running	
Raise event if process not running?	Click **Browse (...)** and select **Yes** for each process for which you want to raise an event if that process is not running. The default setting is based on the options selected in the Module Builder Editor.
Event severity when process is not running	Click **Browse (...)** and set the event severity level, from 1 to 40, to indicate the importance of an event that is raised if a process is not running. The default severity level is based on the options selected in the Module Builder Editor.
Event message when process is not running	Edit the existing message that displays when the job times out. The default text is: “Process [OBJECT] is not running”. [OBJECT] will be substituted with the name of the process.
Process Running	
Raise event if process is running?	Click **Browse (...)** and select **Yes** for each process for which you want to raise an event if that process is running. The default setting is based on the options selected in the Module Builder Editor.
Event severity when process is running	Click **Browse (...)** and set the event severity level, from 1 to 40, to indicate the importance of an event that is raised when the process is running. The default severity level is based on the options selected in the Module Builder Editor.
Event message when process is running	Edit the event message text that displays when the process is running. The default text is: “Process [OBJECT] is running”. [OBJECT] will be substituted with the name of the process.
Process No Longer Running	
Raise event if process is no longer running?	Click **Browse (...)** and select **Yes** for each process for which you want to raise an event if that process has stopped running since the last script iteration. The default setting is based on the options selected in the Module Builder Editor.
Event severity when process is no longer running	Click **Browse (...)** and set the event severity level, from 1 to 40, to indicate the importance of an event that is raised when the process is no longer running. The default severity level is based on the options selected in the Module Builder Editor.

Description	How to Set It
Event message when process is no longer running	Edit the event message text that displays when the process is no longer running. The default text is: “Process [OBJECT] is no longer running”. [OBJECT] will be substituted with the name of the process.
Process Recently Started	
Raise event if process was recently started?	Click **Browse (...)** and select **Yes** for each process for which you want to raise an event if that process was started since the last iteration of this script. The default setting is based on the options selected in the Module Builder Editor.
Event severity when process was recently started	Click **Browse (...)** and set the event severity level, from 1 to 40, to indicate the importance of an event that is raised when the process was recently started. The default severity level is based on the options selected in the Module Builder Editor.
Event message when process was recently started	Edit the event message text that displays when the process has recently started. The default text is: “Process [OBJECT] was recently started”. [OBJECT] will be substituted with the name of the process.
Monitor CPU Usage?	Select **Yes** to monitor the amount of CPU the process uses. The default is based on the options selected in the Module Builder Editor.
CPU usage threshold	Click **Browse (...)** and specify the level of CPU usage for each process that will raise an event. The default threshold setting is based on the options selected in the Module Builder Editor.
Data Collection	
Collect data for CPU usage?	Click **Browse (...)** and select **Yes** for each process for which you want to gather CPU usage data. The default is No.
CPU usage data stream legend	Edit the name of the legend for the CPU usage data stream. The default text is: “Process [OBJECT] CPU usage”. [OBJECT] will be substituted with the name of the process.
Data Collection for Additional Process Instances	
Collect data for all instances of this process, or for each individual instance?	Select the instances for which you want to collect data: ◆ **All**: Combined; collect data on all processes and generate a single data stream for each process, regardless of how many instances of that process are running. Each instance will be listed in the detail. ◆ **Individual**: Collect data on all instances of a process individually and display data in individual data streams. ◆ **Both**: Collect data on all instances of a process as well as each process, and generate data streams for each instance, each process, and all instances in each process combined. The default is All.
Data stream legend for CPU usage for individual instances of a process	Edit the name of the legend for the CPU usage data stream for individual instances of a process. The default text is: “Process [OBJECT]-[PID] CPU usage”. [OBJECT] will be substituted with the name of the process, and [PID] stands for the Process ID for each instance.

Description	How to Set It
Event Notification	
Raise event if CPU usage exceeds threshold?	Click **Browse (...)** and select **Yes** for each process for which you want to raise an event if the CPU usage exceeds threshold. The default setting is based on the options selected in the Module Builder Editor.
Event severity when CPU usage exceeds threshold	Click **Browse (...)** and set the event severity level, from 1 to 40, to indicate the importance of an event that is raised when the CPU usage exceeds the threshold. The default severity level is based on the options selected in the Module Builder Editor.
Event message when CPU usage exceeds threshold	Edit the event message text that displays when the CPU usage exceeds the stated threshold. The default text is: “Process [OBJECT] CPU usage above threshold”. [OBJECT] will be substituted with the name of the process, and [PID] stands for the Process ID for each instance.
Notification for Additional Process Instances	
Raise events on individual instances of a process or all at once?	Select the instances for which you want to raise events: ◆ **All**: Single event including all instances if any instance exceeds the threshold. ◆ **Individual**: Individual events for each instance that exceeds the threshold. The default is All.
CPU usage data stream legend for individual instances of a process	Edit the name of the data stream legend for when CPU usage exceeds the threshold for individual instances of a process. The default text is: “Process [OBJECT]-[PID] CPU usage above threshold”. [OBJECT] will be substituted with the name of the process, and [PID] stands for the Process ID for each instance.
Monitor Memory Usage?	Select **Yes** to raise an event if memory usage exceeds the threshold. The default is based on the options selected in the Module Builder Editor
Memory usage threshold	Click **Browse (...)** and specify the level of memory usage for each process that will raise an event. The default is based on the options selected in the Module Builder Editor.
Data Collection	
Collect data for memory usage?	Click **Browse (...)** and select **Yes** for each process for which you want to gather memory usage data. The default is No.
Memory usage data stream legend	Edit the name of the legend for the memory usage data stream. The default text is: “Process [OBJECT] memory usage”. [OBJECT] will be substituted with the name of the process.
Data Collection for Additional Process Instances	

Description	How to Set It
Collect data for all instances of this process, or for each individual instance?	Select the instances for which you want to collect data: - **All**: Collect data on all processes and generate a single data stream for each process, regardless of how many instances of that process are running. Each instance will be listed in the detail. - **Individual**: Collect data on all instances of a process individually and display data in individual data streams. - **Both**: Collect data on all instances of a process as well as each process, and generate data streams for each instance, each process, and all instances in each process combined. The default is All.
Data stream legend for memory usage for individual instances of a process	Edit the name of the legend for the memory usage data stream for individual instances of a process. The default text is:" Process [OBJECT]-[PID] memory usage". [OBJECT] will be substituted with the name of the process, and [PID] stands for the Process ID for each instance.
Event Notification	
Raise event if memory usage exceeds threshold?	Click **Browse (...)** and select **Yes** for each process for which you want to raise an event if the memory usage exceeds threshold. The default setting is based on the options selected in the Module Builder Editor.
Event severity when memory usage exceeds threshold	Click **Browse (...)** and set the event severity level, from 1 to 40, to indicate the importance of an event that is raised when the memory usage exceeds the threshold. The default severity level is based on the options selected in the Module Builder Editor.
Event message when memory usage exceeds threshold	Edit the event message text that displays when memory usage exceeds the threshold. The default text is: "Process [OBJECT] memory usage above threshold". [OBJECT] will be substituted with the name of the process.
Notification for Additional Process Instances	
Raise events on individual instances of a process or all at once?	Select the instances for which you want to raise events: - **All**: Single event including all instances if any instance exceeds the threshold. - **Individual**: Individual events for each instance that exceeds the threshold. The default is All.
Memory usage data stream legend for each individual instance of a process	Edit the name of the data stream legend for when memory usage exceeds the threshold for individual instances of a process. The default text is: "Process [OBJECT]-[PID] memory usage above threshold". [OBJECT] will be substituted with the name of the process, and [PID] stands for the Process ID for each instance.
Additional Settings	
Event Details	

Description	How to Set It
Event detail format	Specify how you want the event detail information formatted. Your options include: ◆ **HTML Table**: Displays the information in an HTML-formatted table. ◆ **Plain Text**: Displays the information in a table that uses plain text. The default is HTML Table.
Job Timeout	
Elapsed time for job timeout	Specify the length of system inactivity that designates a job timeout. The default is 5 minutes.
Event severity when job timeout occurs	Set the event severity level, from 1 to 40, to indicate the importance of an event that is raised when the job times out. The default is 10.
Event message when job timeout occurs	Edit the existing message that displays when the job times out. The default text is: “Job timeout”.
Job Failure Event Notification	
Event severity when job fails	Set the event severity level, from 1 to 40, to indicate the importance of an event that is raised when the job fails. The default is 10.

4.9 ServiceHealthCheck

This automatically generated Knowledge Script monitors the status of Windows services related to the application, based on the type of information gathered in the Module Builder Editor.

You can use this Knowledge Script to raise AppManager events for the following situations:

- If the service is started
- If the service is disabled or paused
- If the service was not found
- If the service is not running
- If the service is unresponsive or hung
- If the service shut down normally

For some of the above situations, this script can take automated actions on the service, including:

- Starting a service that is not running or that has been shut down normally
- Stopping a service that was started
- Terminating and restarting a service that is hung

ServiceHealthCheck can also collect data for the service status.

4.9.1 Resource Object

Services object

4.9.2 Default Schedule

The default interval for this script is every five minutes.

4.9.3 Setting Parameter Values

Set the following parameters as needed:

Description	How to Set It
Service State	
Data Collection	
Collect data for the state of the service?	Click **Browse (...)** and select Yes for each service for which you want to gather service state data. The default is No.
Collect data for dependent services?	Select **Yes** to gather service data for any other services that are affected by a change in the status of the selected service. The default is unselected.
Service state data stream legend	Edit the name of the legend for the service data stream. The default text is “Service [OBJECT]”. [OBJECT] will be substituted with the name of the service.
Event Notification	
Service Monitor Settings	
Service Started	
Raise event if service is started?	Click **Browse (...)** and select **Yes** for each service for which you want to raise an event if that service is started. The default setting is based on the options selected in the Module Builder Editor.
Event severity when service is started	Click **Browse (...)** and set the event severity level, from 1 to 40, to indicate the importance of an event that is raised when the service is started. The default severity level is based on the options selected in the Module Builder Editor.
Event message when service is started	Edit the event message text that displays when the service is started. The default text is “[OBJECT] Service is started”. [OBJECT] will be substituted with the name of the service.
Service Disabled	
Raise event if service is disabled?	Click **Browse (...)** and select **Yes** for each service for which you want to raise an event if that service is disabled. The default setting is based on the options selected in the Module Builder Editor.
Event severity when service is disabled	Click **Browse (...)** and set the event severity level, from 1 to 40, to indicate the importance of an event that is raised when the service is disabled. The default severity level is based on the options selected in the Module Builder Editor.
Event message when service is disabled	Edit the event message text that displays when the service is disabled. The default text is “[OBJECT] Service is disabled”. [OBJECT] will be substituted with the name of the service.
Service Paused	

Description	How to Set It
Raise event if service is paused?	Click **Browse (...)** and select **Yes** for each service for which you want to raise an event if that service is paused. The default setting is based on the options selected in the Module Builder Editor.
Event severity when service is paused	Click **Browse (...)** and set the event severity level, from 1 to 40, to indicate the importance of an event that is raised when the service is paused. The default severity level is based on the options selected in the Module Builder Editor.
Event message when service is paused	Edit the event message text that displays when the service is paused. The default text is “[OBJECT] Service is paused”. [OBJECT] will be substituted with the name of the service.
Service Not Found	
Raise event if service is not found?	Click **Browse (...)** and select **Yes** for each service for which you want to raise an event if that service is not found. The default setting is based on the options selected in the Module Builder Editor.
Event severity when service is not found	Click **Browse (...)** and set the event severity level, from 1 to 40, to indicate the importance of an event that is raised when the service is not found. The default severity level is based on the options selected in the Module Builder Editor.
Event message when service is not found	Edit the event message text that displays when the service is not found. The default text is “[OBJECT] Service is not found”. [OBJECT] will be substituted with the name of the service.
Service Not Running	
Raise event if service is not running?	Click **Browse (...)** and select **Yes** for each service for which you want to raise an event if that service is not running. The default setting is based on the options selected in the Module Builder Editor.
Event severity when service is not running	Click **Browse (...)** and set the event severity level, from 1 to 40, to indicate the importance of an event that is raised when the service is not running. The default severity level is based on the options selected in the Module Builder Editor.
Event message when service is not running	Edit the event message text that displays when the service is not running. The default text is “[OBJECT] Service is not running”. [OBJECT] will be substituted with the name of the service.
Service Unresponsive	
Raise event if service is unresponsive or hung?	Click **Browse (...)** and select **Yes** for each service for which you want to raise an event if that service is unresponsive or hung. The default setting is based on the options selected in the Module Builder Editor.
Number of iterations before considering service is unresponsive	Specify the number of times the script should run without getting a response from the service before the script should consider the service to be unresponsive. The default is two iterations.
Event severity when service is unresponsive	Click **Browse (...)** and set the event severity level, from 1 to 40, to indicate the importance of an event that is raised when the service is unresponsive. The default severity level is based on the options selected in the Module Builder Editor.

Description	How to Set It
Event message when service is unresponsive	Edit the event message text that displays when the service is unresponsive. The default text is “[OBJECT] Service is unresponsive or hung”. [OBJECT] will be substituted with the name of the service.
Service Shut Down Normally	
Raise event if service was shut down normally?	Click **Browse (...)** and select **Yes** for each service for which you want to raise an event if that service was shut down normally. The default setting is based on the options selected in the Module Builder Editor.
Event severity when service was shut down normally	Click **Browse (...)** and set the event severity level, from 1 to 40, to indicate the importance of an event that is raised when the service was shut down normally. The default severity level is based on the options selected in the Module Builder Editor.
Event message when service was shut down normally	Edit the event message text that displays when the service was shut down normally. The default text is “[OBJECT] Service was shut down normally”. [OBJECT] will be substituted with the name of the service.
Service Action Options	
Service Auto-start	
Auto-start the service if found stopped?	Click **Browse (...)** and select **Yes** for each service you want to start if that service was stopped due to an error.
Auto-start the service if shut down normally?	Click **Browse (...)** and select **Yes** for each service you want to start if that service is shut down normally.
Auto-start dependent services?	Select **Yes** if you want to start any services that are affected by a change in the status of the selected service. The default is unselected.
Auto-start timeout for services?	Specify the length of the time to wait for the service to start before the auto-start option is considered timed out. The default is 30 seconds.
Raise event if auto-start successful?	Click **Browse (...)** and select **Yes** for each service for which you want to raise an event if that service is started successfully.
Event severity when auto-start successful	Click **Browse (...)** and set the event severity level, from 1 to 40, to indicate the importance of an event when a service is started successfully.
Event message when auto-start successful	Edit the event message text that displays when the service is started correctly. The default text is “[OBJECT] Service auto-started successfully”. [OBJECT] will be substituted with the name of the service.
Raise event if auto-start failed?	Click **Browse (...)** and select **Yes** for each service for which you want to raise an event if AppManager could not start the service.
Event severity when auto-start failed	Click **Browse (...)** and set the event severity level, from 1 to 40, to indicate the importance of an event in which AppManager could not start the service.
Event message when auto-start failed	Edit the event message text that displays when AppManager could not start the service. The default text is “[OBJECT] Service auto-start failed”. [OBJECT] will be substituted with the name of the service.
Raise Event if Auto-start Disabled and Service is Down?	Select **Yes** if you want to raise an event if a service has auto-start disabled and the service is not running, so as a result, the service cannot be restarted. The default is Yes.

Description	How to Set It
Event severity when auto-start disabled and service is stopped	Click **Browse (...)** and set the event severity level, from 1 to 40, to indicate the importance of an event that is raised when a service has auto-start disabled and the service is not running.
Event message when auto-start disabled and service is stopped	Edit the event message text that displays when a service has auto-start disabled and the service is not running. The default text is “[OBJECT] Service stopped and auto-start disabled”.
Service Stop	
Stop the service if found started?	Click **Browse (...)** and select **Yes** for each service you want to stop if AppManager finds that the service has started. The default severity level is based on the options selected in the Module Builder Editor.
Stop dependent services?	Select **Yes** to stop any services that are dependent on the service that AppManager found that was started. The default is unselected.
Raise event if stop successful?	Click **Browse (...)** and select **Yes** for each service for which you want to raise an event if that running service is stopped successfully.
Event severity when stop successful	Click **Browse (...)** and set the event severity level, from 1 to 40, to indicate the importance of an event that is raised when the service is stopped successfully.
Event message when stop successful	Edit the event message text that displays when the service is stopped successfully. The default text is “[OBJECT] Service stopped successfully”. [OBJECT] will be substituted with the name of the service.
Raise event if stop failed?	Click **Browse (...)** and select **Yes** for each service for which you want to raise an event if AppManager cannot stop the service. The default is Yes.
Event severity when stop failed	Click **Browse (...)** and set the event severity level, from 1 to 40, to indicate the importance of an event that is raised when AppManager cannot stop the service.
Event message when stop failed	Edit the event message text that displays when AppManager cannot stop the service. The default text is “[OBJECT] Service stop failed”. [OBJECT] will be substituted with the name of the service.
Terminate Service	
Terminate if service is unresponsive?	Click **Browse (...)** and select **Yes** for each service you want to stop if AppManager considers the service to be unresponsive. The default setting is based on the options selected in the Module Builder Editor.
Terminate the dependents if service is unresponsive?	Select **Yes** if you want to stop any services that are dependent on the selected service that AppManager considers to be unresponsive. The default is Yes.
Raise event if terminated successfully?	Click **Browse (...)** and select **Yes** for each service for which you want to raise an event if AppManager stops the unresponsive service.
Event severity when terminate is successful	Click **Browse (...)** and set the event severity level, from 1 to 40, to indicate the importance of an event that is raised when AppManager stops the unresponsive service.

Description	How to Set It
Event message when terminate is successful	Edit the event message text that displays when AppManager stops the unresponsive service. The default text is “[OBJECT] Service terminated successfully”. [OBJECT] will be substituted with the name of the service.
Raise event if terminate failed?	Click **Browse (...)** and select **Yes** for each service for which you want to raise an event if the attempt to terminate that service failed. The default is Yes.
Event severity when terminate failed	Click **Browse (...)** and set the event severity level, from 1 to 40, to indicate the importance of an event that is raised when AppManager cannot stop the service.
Event message when terminate failed	Edit the event message text that displays when AppManager cannot stop the service. The default text is “[OBJECT] Service termination failed”. [OBJECT] will be substituted with the name of the service.
Restart Terminated Service?	
Restart the dependents?	Select **Yes** if you want to restart any services dependent on the selected service that AppManager previously terminated. The default is Yes.
Restart timeout for services?	Specify the length of the restart timeout. The default is 30 seconds.
Raise event if service restarted successfully?	Click **Browse (...)** and select **Yes** for each service for which you want to raise an event if AppManager successfully restarted that service.
Event severity when service restarted successfully	Click **Browse (...)** and set the event severity level, from 1 to 40, to indicate the importance of an event that is raised when AppManager successfully restarted that service.
Event message when service restarted successfully	Edit the event message text that displays when AppManager successfully restarted that service. The default text is “[OBJECT] Service restarted successfully”. [OBJECT] will be substituted with the name of the service.
Raise event if service restart failed?	Click **Browse (...)** and select **Yes** for each service for which you want to raise an event if AppManager failed to restart that service. The default is Yes.
Event severity when restart failed	Click **Browse (...)** and set the event severity level, from 1 to 40, to indicate the importance of an event that is raised when AppManager failed to restart that service.
Event message when restart failed	Edit the event message text that displays when AppManager failed to restart that service. The default text is “[OBJECT] Service failed to restart”. [OBJECT] will be substituted with the name of the service.
Additional Settings	
Event Details	
Event detail format	Specify how you want the event detail information formatted. Your options include: ◆ **HTML Table**: Displays the information in an HTML-formatted table. ◆ **Plain Text**: Displays the information in a table that uses plain text. The default is HTML Table.
Job Timeout	

Description	How to Set It
Elapsed time for job timeout	Specify the length of system inactivity that designates a job timeout. The default is 5 minutes.
Event severity when job timeout occurs	Set the event severity level, from 1 to 40, to indicate the importance of an event that is raised when the job times out. The default is 10.
Event message when job timeout occurs	Edit the existing message that displays when the job times out. The default text is “Job timeout”.
Job Failure Event Notification	
Event severity when job fails	Set the event severity level, from 1 to 40, to indicate the importance of an event that is raised when the job fails. The default is 10.

5 Troubleshooting Module Builder

This chapter describes how to troubleshoot NetIQ Module Builder. The following problems are discussed:

Section 5.1, “Problems with Module Builder Editor,” on page 61

Section 5.2, “Problems with Module Builder Managed Object,” on page 63

5.1 Problems with Module Builder Editor

5.1.1 The menu extensions added during installation do not appear in the Operator Console

Problem: The menu extensions used for the Browse dialog box that are newly added during the installation of Module Builder Editor do not appear in the Operator Console.

Solution: You must restart the Operator Console after installing the Module Builder Editor components to view the newly added extensions.

5.1.2 Uninstalling Module Builder Editor prompts to restart the console computer

Problem: If AppManager is running while you are uninstalling the Module Builder Editor, you might be prompted to restart the console computer. This problem occurs because AppManager Operator Console locks the menu extensions specific to Module Builder.

Solution: Close the Operator Console before you uninstall Module Builder Editor.

5.1.3 Configuring the performance counter options in Module Builder Editor adds erroneous Windows event log entries

Problem: Module Builder Editor adds unnecessary events to the Windows event logs when you configure the performance counter options in Module Builder Editor. This occurs because of mismatches between the performance counters for the application and the registry for the performance counters.

Solution: Perform the following steps after you have installed Module Builder Editor:

1. Open Registry Editor and navigate to `HKLM\System\CurrentControlSet\Services\EventLog\Application\NetIQmcCnt\`.
2. Change the registry value of `EventMessageFile` to `C:\Program Files\netiq\appmanager\bin` or the relevant AppManager Agent `bin` directory, not the `system32` directory.
3. Restart the NetIQ AppManager Client Resource Monitor (`netiqmc.exe`) and the NetIQ AppManager Client Communication Manager (`netiqccm.exe`) services.

4. From the command line, execute the `lodctr mccnt.ini` command.
5. Import `mccnt.reg` from `C:\Program Files\netiq\appmanager\bin\` or the relevant AppManager Agent `bin` directory.

NOTE: This workaround is not required if your environment does not have the `HKLM\System\CurrentControlSet\Services\EventLog\Application\NetIQmcCnt\` key.

5.1.4 Some installed applications may not display in Select Application from List dialog

Problem: Due to the way some applications handle their installation and uninstallation processes, those applications may not display in this list.

Solution: If the application you want to manage is not in the list, you must define the name and path manually.

5.1.5 Cannot set the severity level for success or failure of a service

Problem: When you are specifying the actions to take on a service, you cannot set the severity level for the success or failure of these actions in the Module Builder Editor.

Solution: The corresponding success or failure severities can only be updated in the AppManager Knowledge Script after the script has been checked into your AppManager Repository. You can view the corresponding severity information for the actions in the application monitoring contract PDF document.

5.1.6 Event Log Source name should be specified using the full source name or by using a regular expression

Problem: On Windows Vista and Windows 2008, Module Builder Editor requires that the Event Log Source name should be specified using the full source name or by using a regular expression. The source name that appears in the Windows Event Viewer may only contain a shortened version of the full source name.

Solution: When adding an event using the Existing Events tab in the Module Builder Editor, the source name appears in the full format. However, when editing an event or adding an event using the Ad-hoc Events tab, you must specify the Source value in this full format or use a Regular Expression when using this event property.

To find the Source value from a Windows Vista or Windows Server 2008 Event Viewer:

1. Select the Event.
2. Click **Details**.
3. In the Friendly View, Click **System** > **Provider** > **[Name]**

5.1.7 Excessive icons in .mob file might cause Module Builder Editor to fail

Problem: The Module Builder Editor might fail when attempting to load a saved `.mob` file that contains a high number of icons for processes or applications. The Module Builder Editor cannot display the icons if the number of icons exceeds the Microsoft Windows limit for GDI (Graphics Device Interface) Objects.

Solution: You can prevent Module Builder Editor from saving and loading these icons by adding the "LoadAndSaveIcons" key to the `MoBEditor.exe.config` file. For more information, see the "Configuring Settings for the Module Builder Editor" topic in the "Installing Module Builder" chapter of the *Module Builder User Guide*.

5.1.8 Error messages display when the name of a monitored service is more than 80 characters

Problem: The Module Builder Editor displays a message stating that a service no longer exists if the service name is greater than 80 characters. This situation only applies to the service name, not the service display name.

Solution: You can dismiss the error messages and still manually add the service to monitor.

5.2 Problems with Module Builder Managed Object

5.2.1 Application-specific online Help for Knowledge Scripts is unavailable

Problem: When you press **F1** or click the **Help** button from a Module Builder Knowledge Script Values tab, the following error message is displayed:

```
Page cannot be displayed
```

Solution: To view `generic` Help for Module Builder Knowledge Scripts:

1. Click **Help** > **Help Topic**.
2. Go to the `Knowledge Script Reference` folder on the `Contents` tab.
3. Select either the `Discovery` or the `Module Builder Knowledge Script` folder.

Help is also available in the *Module Builder User Guide*, a PDF file located in the `\Program Files\NetIQ\` folder on the agent computer.

5.2.2 Clicking the Browse button leads to a mismatch on case-sensitive AppManager repositories (QDBs)

Problem: AppManager converts the root object string to uppercase when matching a computer name, including when a computer name is used in the Module Builder application name. When you click the Browse button, an error message displays on case-sensitive QDBs.

Solution: Avoid using the computer name in the application name. If you need to use the computer name in the application name, edit the Knowledge Scripts by matching the case as needed.

5.2.3 Possible mismatches might occur when you run discovery on a module that was recently updated with the Module Builder Editor

Problem: By default, the Module Builder Discovery Knowledge Script adds a unique identifier to monitored object names that have the same name in the Tree View. For example, this naming convention may occur if the application uses log files found in separate folders. If you or another user reconfigure the Module Builder Editor settings that use this unique identifier, the subsequent re-discovery process in AppManager might cause the labels to mismatch from previous results.

Solution: You should deploy new monitoring jobs to prevent previous events and collected data from being associated with the revised labels.

5.2.4 Cannot create a view in Operator Console if a performance counter name contains special characters

Problem: You cannot create a view in Operator Console if a performance counter name contains special characters.

Solution: If you want to create a dynamic view based on the name of a performance counter that has special characters such as "\" or "%" in its name, you should use the wildcard symbol "*" in place of the special character. The use of special characters in the filters for views is not supported in AppManager.

5.2.5 Data collision when the monitored objects have very similar long names

Problem: Collected data might collide into a single graph data entry if a Knowledge Script collects data on monitored objects that have very similar long names.

Solution: Modify the data legend parameter for that Knowledge Script.

5.2.6 Running the PerformanceMetrics Knowledge Script adds erroneous Windows event log entries

Problem: Module Builder adds unnecessary events to the Windows event logs when you run the PerformanceMetrics Knowledge Script. This occurs because of mismatches between the performance counters for the application and the registry for the performance counters.

Solution: Perform the following steps after you have installed Module Builder:

1. Open Registry Editor and navigate to `HKLM\System\CurrentControlSet\Services\EventLog\Application\NetIQmcCnt\`.
2. Change the registry value of `EventMessageFile` to `C:\Program Files\netiq\appmanager\bin` or the relevant AppManager Agent `bin` directory, not the `system32` directory.
3. Restart the NetIQ AppManager Client Resource Monitor (`netiqmc.exe`) and the NetIQ AppManager Client Communication Manager (`netiqccm.exe`) services.

4. From the command line, execute the `lodctr mccnt.ini` command.
5. Import `mccnt.reg` from `C:\Program Files\netiq\appmanager\bin\` or the relevant AppManager Agent `bin` directory.

NOTE: This workaround is not required if your environment does not have the `HKLM\System\CurrentControlSet\Services\EventLog\Application\NetIQmcCnt\` key.

5.2.7 The QMob.exe process may stop prematurely if more than one EventLogCheck jobs are running and a timeout condition occurs

Problem: An application fault in `qmob.exe` occurs if you run multiple EventLogCheck Knowledge Script jobs on the same agent computer and the **Elapsed time for job timeout** parameter value is exceeded while one of the jobs is executing.

Solution: Increase the **Elapsed time for job timeout** parameter value in the EventLogCheck Knowledge Script parameter.