Chapter 3

Application Monitors

AppMetrics utilizes *application monitors* to organize data collection and analysis per application server. An application monitor is defined on the AppMetrics manager computer and collects data from an AppMetrics agent running on a remote computer or optionally the local computer. An agent will collect COM+ instrumentation events and Windows performance data from specified COM+ and .NET serviced component applications and send that data to its associated monitor for processing and analysis.

Depending on the type of monitor selected, an application monitor can provide detailed or aggregated performance data along with real time alerts based upon application performance and user selected thresholds. Alerts can be generated as Windows event log entries, SNMP traps, or email/pager messages using SMTP. They can also be used to trigger Windows Script Components in order to recycle or shutdown problem applications.

Two crucial functions provided by AppMetrics are production and diagnostics monitoring. Each type of monitoring has unique advantages and limitations. Production monitors are used to observe and record the health of an application in an operational environment. They are designed to place minimal load on the application server allowing for 24 hour monitoring of critical applications. If a production monitor detects problems with a specific component or transaction, a diagnostics monitor can then be used to generate more in-depth information about the component or transaction in question down to the component and method-call level. A diagnostics monitor will however place a higher load on the application server, thus should only be used when tracking down a specific problem detected by a production monitor.

This chapter will cover the core application monitor concepts: application monitor management, configuration, and use; agent management and configuration; the use of *AppMetrics Console* instances to manage AppMetrics monitors and agents on remote computers.

Remote Data Collection

Remote Data Collection enables AppMetrics to monitor applications while using a minimum amount of system overhead. It achieves this by performing only what is necessary on the application server while performing the more system-intensive tasks on the AppMetrics manager computer. Figure 3-1 illustrates the remote-collection model.

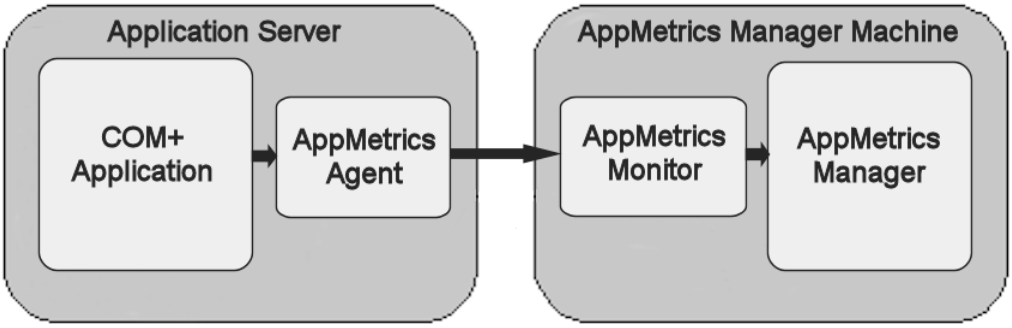


Figure 3-1 Remote Collection Model

In the remote-collection model an *AppMetrics Agent* runs on the application server and performs the following tasks:

- Collects instrumentation data on the monitored COM+ application(s)
- Reports this data to the associated *AppMetrics Monitor* running on the AppMetrics manager computer

As COM+ relays instrumentation data to the agent by way of events, the agent's work on the application computer is event driven and kept to a minimum, freeing up system resources that an administrator can devote to running and managing production applications on the server.

In turn, the *AppMetrics Monitor* collects, analyzes, logs, and reports the application data. As the monitor performs all this on a separate computer from the application server, AppMetrics performs a minimum amount of its work on the application computer while performing the bulk of its work on the manager computer.

Relationship between a Monitor and Agent

A one-to-one relationship exists between a monitor and an agent. This means that a monitor collects data from only one agent, and in turn, that agent sends data to only one monitor.

However, as many monitors as necessary can be created on the manager computer to monitor the various COM+ applications throughout your network. Likewise, you can create as many agents as needed on the application server as long as each agent reports to its own monitor.

Monitors, Agents, and Operating Systems

AppMetrics agents may be run on Windows 2000 Advanced Server, Windows Server 2003, Windows Server 2003 R2, Windows Server 2008, and Windows Server 2008 R2. AppMetrics monitors can run on any of the above server operating systems in addition to the workstation operating systems, Windows XP Professional, Windows Vista, and Windows 7.

Both 32-bit and 64-bit versions of the operating systems are supported.

Application Monitor Types

AppMetrics implements two ***Application Monitor Types***:

- **COM+ Production Monitor**
- **COM+ Diagnostics Monitor**

The ***COM+ Production Monitor*** is designed for the monitoring of applications running in actual production environments, as the name implies. It uses minimal resources on the application server and fewer resources on the AppMetrics manager. Production monitors display process information (such as CPU, Memory, Threads, etc) for each monitored application. Runtime screens display COM+ metrics related to *Transactions* (root method or component depending on the **Transactional Detail Levels** setting) , and *Components*.

Process, transaction, and component thresholds may be set and configured in order to notify operations staff of abnormal conditions. Notifications can be delivered via the Windows Event Log, SMTP, SNMP, or Windows Script Components.

Additionally, ***Microsoft System Center Operations Manager*** can be used to monitor the health of the COM+ applications monitored by AppMetrics. The AppMetrics SCOM management pack will gather AppMetrics notifications delivered to the Windows Event Log and present them as SCOM alerts.

The ***COM+ Diagnostics Monitor*** should be chosen when maximum detail is required. If the application under investigation has problems that otherwise eludes standard analysis, the diagnostics monitor provides a user an opportunity to record time-correlated data about the application. A diagnostics monitor can generate large, detailed data files. These data sets enable the user (with the aid of AppMetrics Reports or third-party reporting software) to review the state of the system at any point during the monitor's operation.

AppMetrics Console

The ***AppMetrics Console*** is used to create and manage application monitors and agents. It is implemented as a Microsoft Management Console "snap-in".

To open the AppMetrics Console:

1. Log in to Windows with an account that has been added to the AppMetrics Administrators local group on the computers that you plan to access through the console.
2. From the **Start** menu, click **Programs → Xtremesoft → AppMetrics for Transactions → AppMetrics for Transactions**.

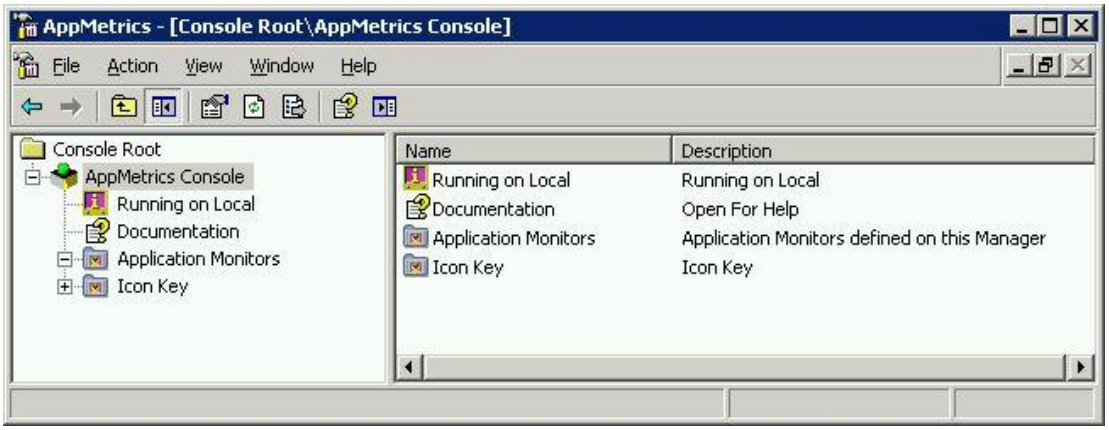


Figure 3-2 AppMetrics Console

Note: If starting the console for the first time on a computer where AppMetrics was installed with the *Reports and Console* setup type, the user will be prompted to specify the AppMetrics Manager server name. Enter the name of the manager computer and click OK.

Each instance of an AppMetrics Console may manage one or more monitors running on an AppMetrics Manager computer. The AppMetrics Console connects to the local server by default.

Connecting to a Remote Manager or Agent Computer

To connect to a remote AppMetrics Manager or Agent computer, follow the steps below:

1. Right-click the *AppMetrics Console* node in the navigation tree.
2. Select *Properties* from the pop-up menu.

   The ***AppMetrics Console Properties*** dialog is used to enter the server name of the AppMetrics Manager or Agent computer.

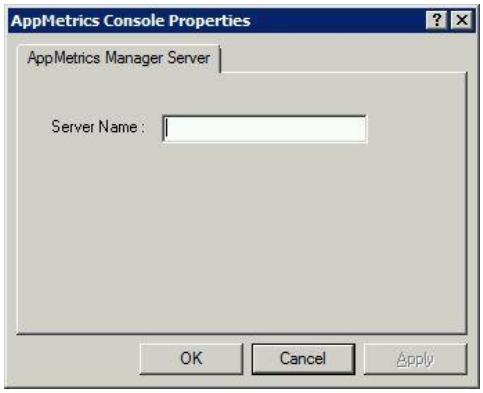


Figure 3-3 AppMetrics Console Properties Dialog

3. Enter the server name of the desired AppMetrics Manager or Agent computer.

The user may optionally add additional AppMetrics Console instances in the MMC to manage AppMetrics agents running on remote servers. See *Multiple AppMetrics Console Instances* on Page 3-68 for more information.

Managing Application Monitors and Agents

Creating Application Monitors

1. In the left-pane navigation tree under **Console Root** → **AppMetrics Console**, right click the **Application Monitors** folder and navigate to the **New** → **Application Monitor** menu item.

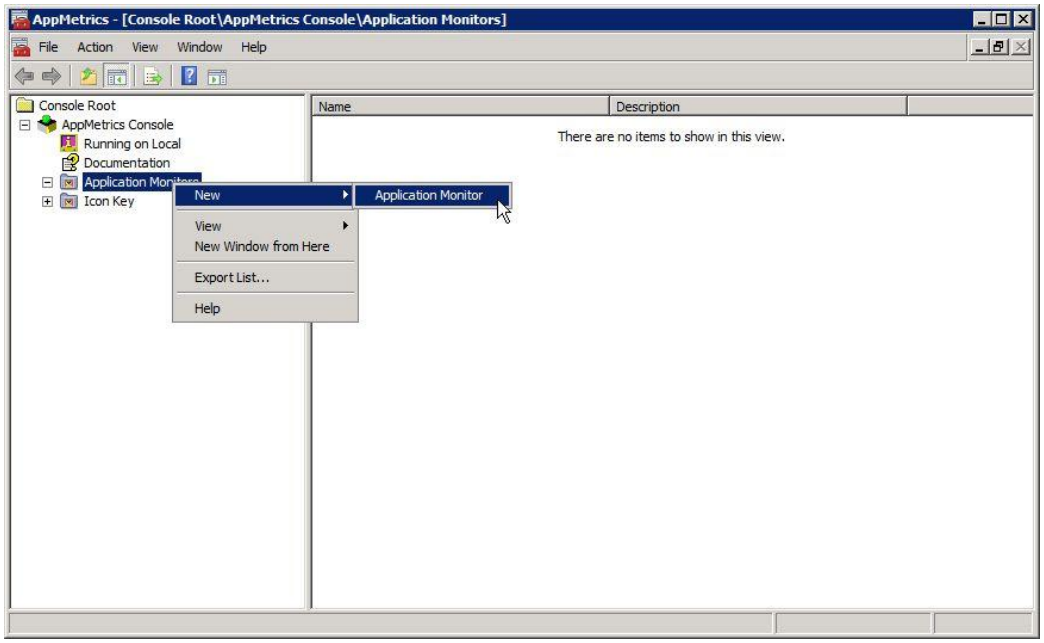


Figure 3-4 Creating an Application Monitor

2. The ***Create New Monitor*** dialog allows the user to choose between the two different monitor types. In the example below, the *COM+ Production Monitor* type is selected.

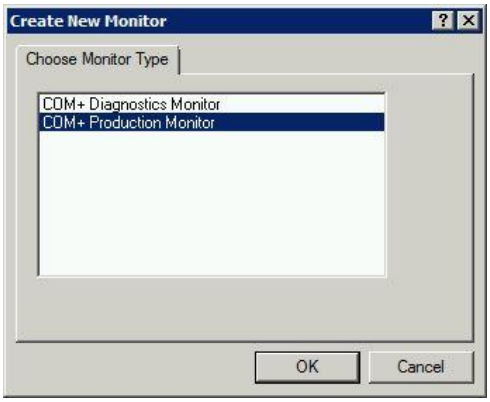


Figure 3-5 Create New Monitor Dialog

3. Select the desired *monitor type* and then hit **OK**.

4. The ***Monitor Properties*** dialog is used to enter the name, description, and start option for the monitor. In the example below, a monitor name of Sample Production Monitor is given.

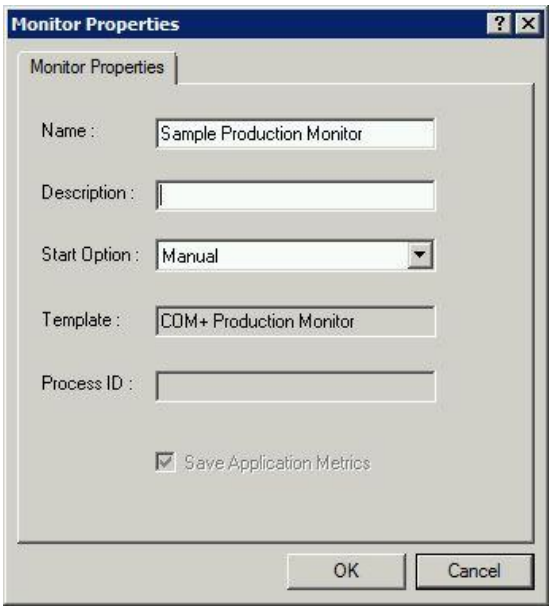


Figure 3-6 Monitor Properties Dialog

5. Enter the monitor name in the **Name** field using only alphanumeric characters and optionally, a separating space between words. Names already in use by other monitors are not accepted.

6. (Optional) Enter a description for the monitor in the **Description** field.

7. Select the desired **Start Option**. The *Manual* setting (default) will start the monitor when the AppMetrics Console starts. The *Automatic* setting starts the monitor when the AppMetrics service is started.

8. Click **OK**.

Note:

When creating a production monitor, the **Save Application Metrics** checkbox will be checked and disabled upon monitor creation. The checkbox is enabled on subsequent invocation of the ***Monitor Properties*** dialog. The dialog is accessible by right-clicking the **Monitor** node, then selecting **Properties**. When creating a diagnostic monitor, the **Save Application Metrics** checkbox will not be present.

Please refer to **Production Monitor Shutdown Option** on page 3-57 for more information on this feature.

Once the new monitor is added to the console it is listed under the ***Application Monitors*** folder as shown in Figure 3-7.

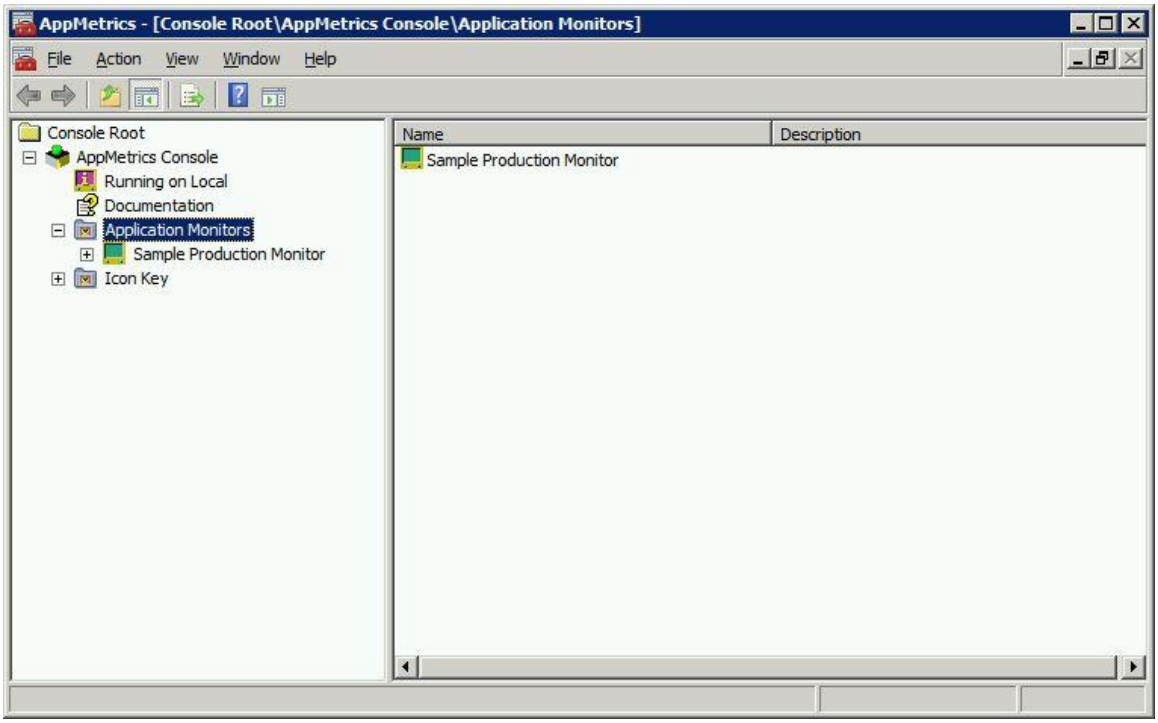


Figure 3-7 New Application Monitor

Before a new monitor can collect metrics an agent must first be added to the monitor.

The following page describes how the *AppMetrics Console* is used to create an agent on a selected application server from which the new monitor will collect application metrics.

Adding an Agent to a Monitor

After creating a monitor an agent will need to be added to it in order to collect instrumentation data from the desired server.

Note:

- Perform this procedure from the AppMetrics Manager computer (locally or remotely).
- Ensure that the AppMetrics Agent software is properly installed and configured on the application server which is to be monitored. Please refer to the AppMetrics Installation Guide for more details.
- You must be logged into the AppMetrics Manager computer with a Windows account which resides in the AppMetrics Administrators group on both the manager and agent computers. Failure to do so will result in “Access Denied” errors during the procedure.

To add an agent to a monitor:

1. Under the ***Applications Monitors*** folder, expand the monitor.
2. Right-click the ***Agents*** node and navigate to the **New** → **Agent** menu item.

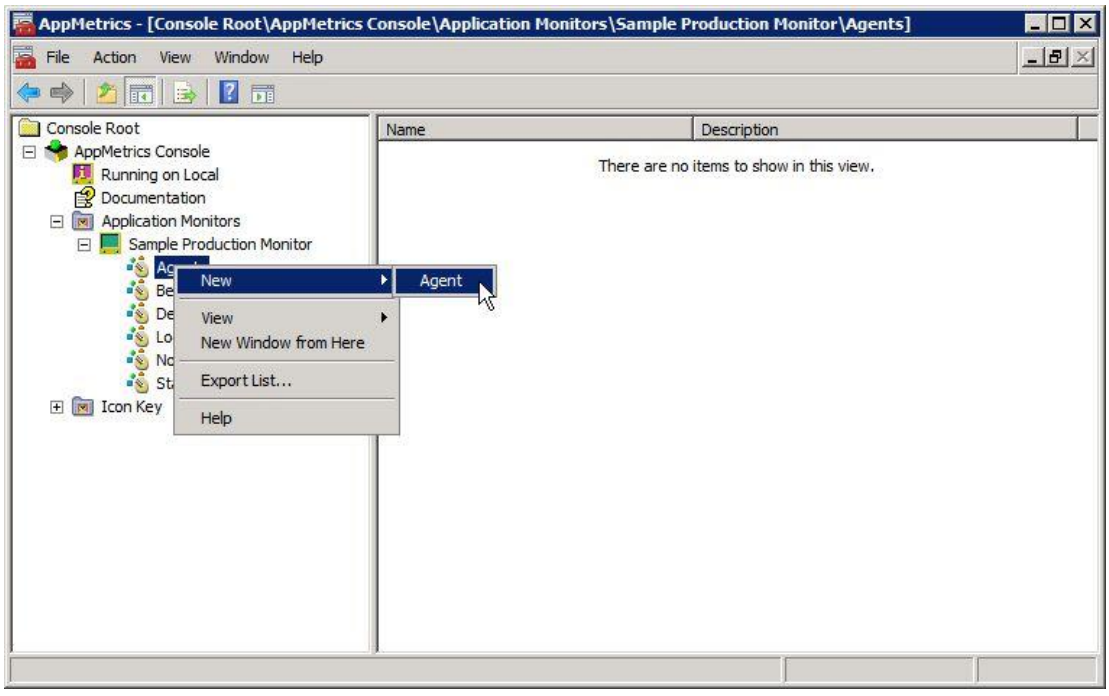


Figure 3-8 Adding an Agent to a Monitor

3. The ***Add Agent*** dialog is used to specify the desired application server on which the agent will be created (or selected if one already exists). The **Browse** button must be used to enter the *agent name*.

Figure 3-9 Add Agent Dialog

4. In the **Server Name** field, type the name of the desired application server to monitor.
5. Click the **Browse** button.
6. The ***Browse Agents*** dialog will display a list of any unused agents of the correct type which already exist on the server.

   Note: This would occur if agents had previously been created on the server but were not manually deleted after the associated monitor was removed.

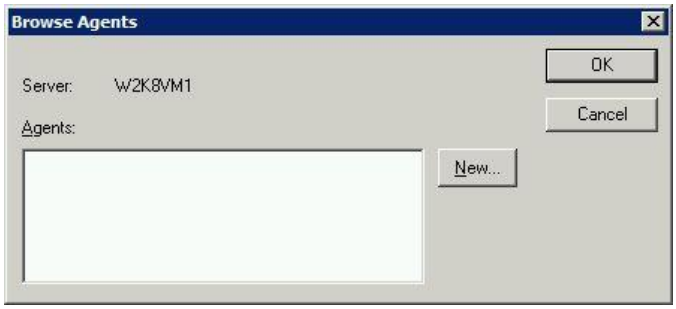


Figure 3-10 Browse Agents Dialog

   At this point, you can do one of the following:

   - Choose an agent from the list of available **Agents** on the specified server that are present but are not connected to another monitor. To choose one of these agents, click it in the list, then skip to step 13.

     OR

   - Create a new agent if no other agents are available. To do so, proceed with the next step.

7. Click the **New** button.

8. The ***Create New Agent*** dialog is used to specify the name, description, and start option for the agent. In the example below, an agent name of Prod Agent has been entered.

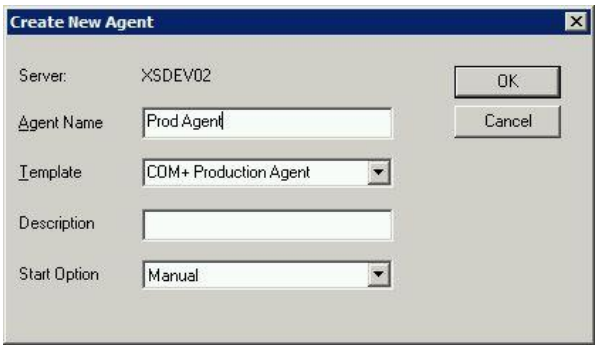


Figure 3-11 Create New Agent Dialog

9. Enter a name in **Agent Name** field to identify the new agent.
10. (Optional) Enter a description for the agent in the **Description** field.
11. Select the desired **Start Option**. The *Manual* setting (default) will cause the agent to start when its associated monitor starts. The *Automatic* setting starts the agent when the AppMetrics service is started on the application server.
12. When done, click **OK**. This returns you to the ***Browse Agents*** dialog.

    Note: AppMetrics chooses the *Template* for the agent based on the monitor type.
13. In the ***Browse Agents*** dialog, click **OK**. This returns you to the ***Add Agent*** dialog.
14. Click **OK** in the ***Add Agent*** dialog.

Once the agent is created it is added to the console and listed under the monitor's *Agents* node.

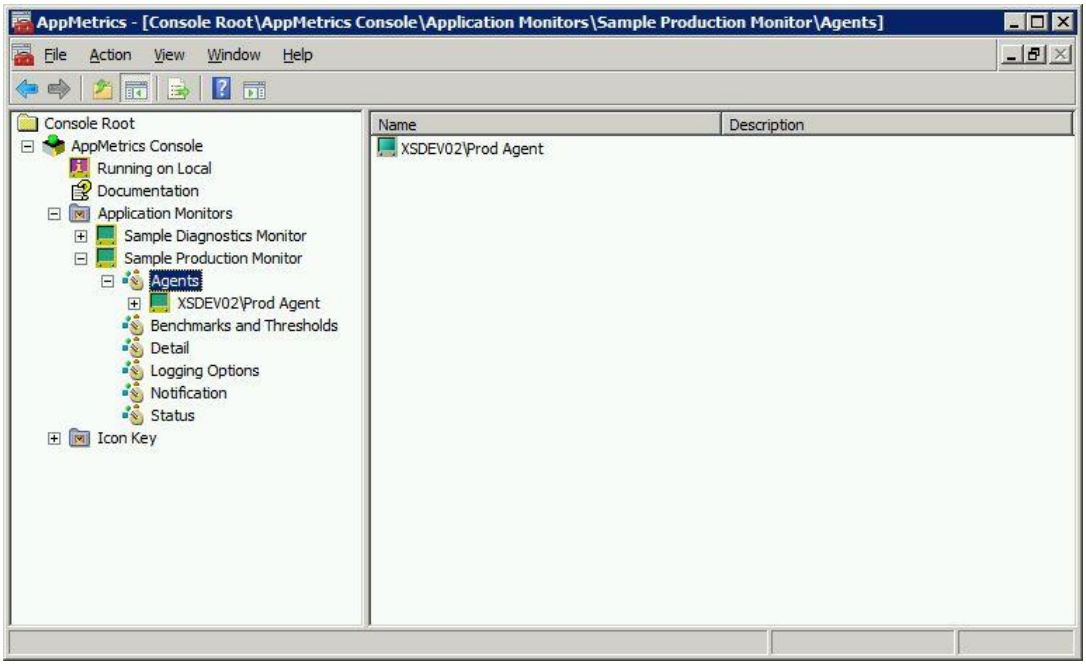


Figure 3-12 New Agent Added to Monitor

The new agent will be also displayed under the *AppMetrics Agents* folder on an *AppMetrics Console* connected to the application server.

Note: The *AppMetrics Console* connected to the application server may need to be refreshed in order to view a new agent added from another computer. You can refresh the view by right-clicking the AppMetrics Console node and selecting **Refresh** in the popup menu.

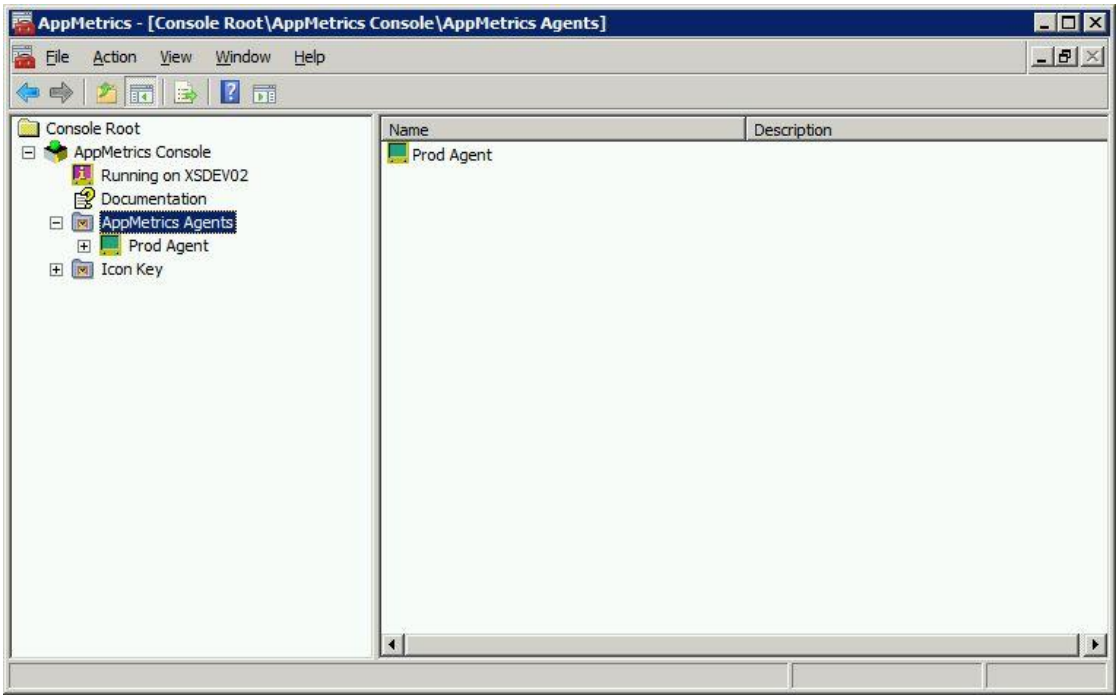


Figure 3-13 New Agent Displayed on Application Server

Configuring an Agent

Once an agent has been added to the new monitor it will need to be configured by selecting which applications to monitor on that server. The ***COM+ Applications Configuration View*** is accessible by clicking the *COM+ Applications* node located under a monitor's associated agent node (Figure 3-14).

COM+ Applications Configuration View

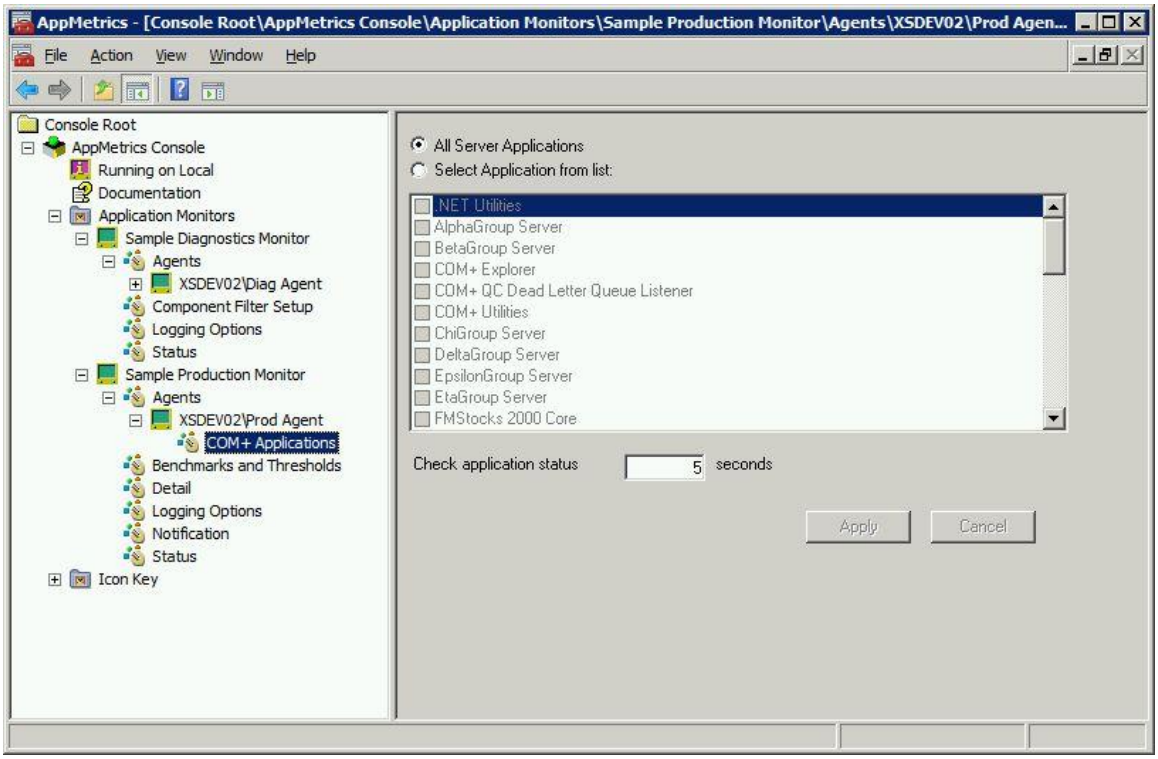


Figure 3-14 COM+ Applications Configuration View

By default all installed server applications are monitored. This view enables users to choose which server applications to monitor by selecting the **Select Applications from list** radio button and then selecting the desired applications to monitor.

AppMetrics will only list server applications In the ***COM+ Applications Configuration View***. Library applications may also be monitored if they run within the context of a monitored server application.

The interval for updating the application status is 5 seconds by default, and may be modified by changing the value in the **Check application status** field. This setting affects the ***Applications Runtime View*** refresh rate and differs from the production monitor sampling interval, which is configurable in the ***Setup Configuration View*** (see Figure 3-31).

Note: If the same name is used by more than one application on the computer, AppMetrics will not display these additional applications in this view.

Starting a Monitor and Agent

After configuring the monitor and agent they can be started by right clicking the monitor node and selecting **Start Monitor** in the popup menu.

An agent should always be started by starting its monitor and not by starting it directly from an *AppMetrics Console* running on the application server.

Once the monitor and agent are started the agent will begin collecting application data and send it to the monitor. The monitor will process the incoming data from the agent and periodically upload it to the SQL database.

Stopping a Monitor and Agent

A monitor and its agent can be stopped by right-clicking the monitor node and then selecting **Stop Monitor** from the popup menu.

An agent should normally only be stopped by stopping the monitor. If stopped directly from the agent node displayed on an *AppMetrics Console* running on the application server, unpredictable results may occur. In the event of a agent or monitor crash however, it may be necessary to stop the agent directly. To do so, use *AppMetrics Console* to connect to the application server, then right-click the agent node and select **Stop Agent** in the popup menu.

Deleting a Monitor

To delete a monitor, right-click on the monitor node then select **Delete** from the popup menu.

When a monitor is deleted all related transaction and component data is removed. Associated log files, folders, and database files are deleted.

Before deleting a monitor the user should first backup any desired data files for future analysis. This can be done by detaching the database and copying the database files to a safe location.

The agent is removed from the monitor automatically if not yet removed, but the agent itself will still need to be manually deleted from the agent computer (application server).

Removing an Agent from a Monitor

Before an agent can be deleted it must first be removed from any monitor it may be associated to.

Deleting a monitor will automatically remove the association between the monitor and agent, but can be done manually if desired. Removing the agent from a monitor does not delete the agent itself; it merely disconnects the agent from the monitor. As a result, the agent may be reused for a different monitor.

To remove an agent from a monitor, right-click the agent node under the monitor's *Agent* folder and select **Remove Agent** from the popup menu.

Deleting an Agent

An agent can be deleted through the *AppMetrics Console* on the agent computer or from any other computer with the *AppMetrics Console* installed if the console is able to remotely connect to the agent computer.

To delete an agent from the agent computer, right-click the agent node and select **Delete Agent** form the popup menu.

Using and Configuring Application Monitors

Monitor runtime views are available when a running monitor is selected in the left pane navigation tree. Configuration views are visible when one of the items under the monitor node is selected, but can only be modified when the monitor is stopped.

Right-clicking on a monitor's name will cause a pop-up menu to appear. Clicking on the **Start Monitor** item starts the monitor. The monitor icon changes appearance to indicate the monitor is in an active state and collecting metrics.

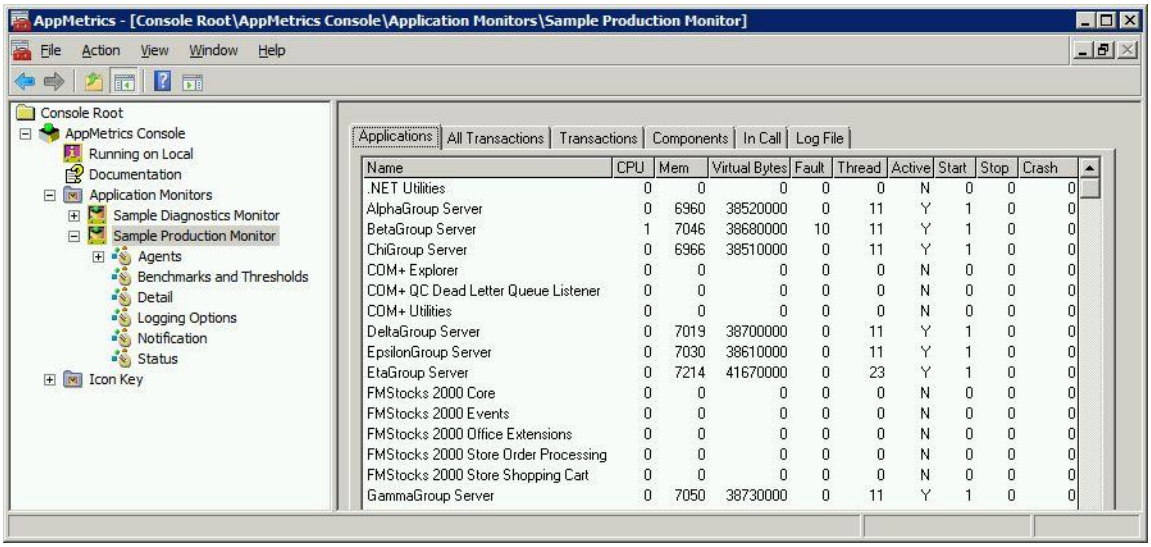


Figure 3-15 Monitor Run-Time Views

Monitors are stopped by right clicking on the monitor node and selecting **Stop Monitor**. Once stopped, the monitor icon will change back to its inactive state.

The following sections on **COM+ Production Monitors** and **COM+ Diagnostics Monitors** describe the runtime and configuration views for each.

COM+ Production Monitors

Production monitors may be used to observe, measure, and record the day-to-day health of deployed applications. The monitors collect and display metrics that are uploaded to a SQL Server database. These metrics represent aggregated or "rolled-up" information as compared to the more granular data recorded in diagnostics monitors.

Both production and diagnostics monitors contain an ***Applications Runtime View*** which display process information related to the list of monitored applications.

Performance threshold monitoring can be used to trigger alerts. These thresholds may be configured through use of the ***Benchmarks and Thresholds View***. On the runtime views, metrics which exceed designated thresholds are signaled by changing the color of the corresponding data field.

AppMetrics production monitors can also be configured to send notifications when any defined thresholds are exceeded. Notifications may be sent in a number of ways, such as invoking a Windows component script, sending email messages to specific individuals, firing SNMP traps to alert via enterprise management tools, or logging entries to the Windows Event Log which can concurrently be monitored by tools such as ***Microsoft Systems Center Operations Manager***.

Production Monitor Runtime Views

Applications Runtime View

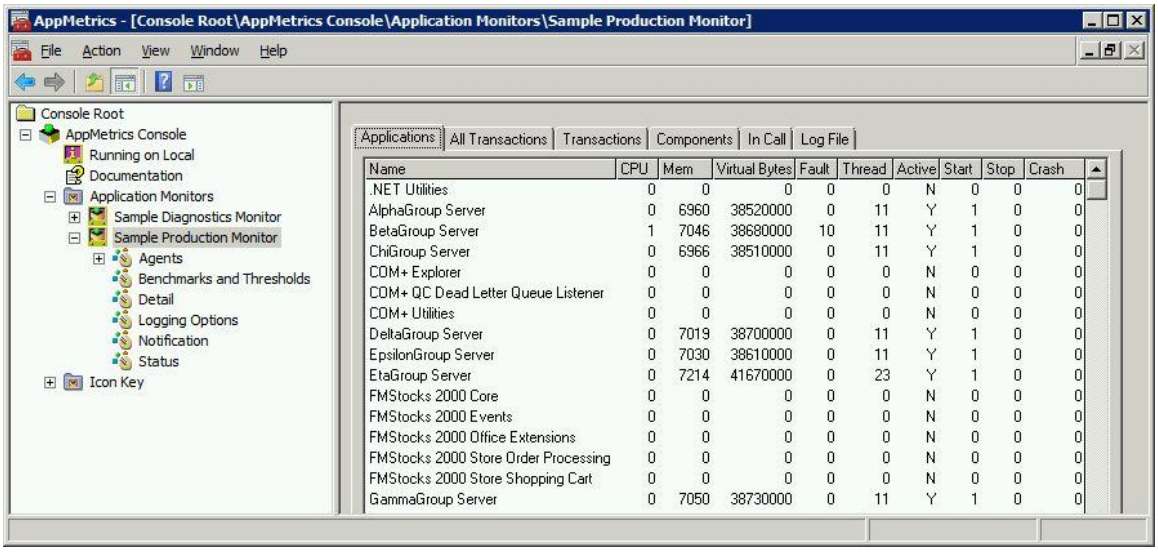


Figure 3-16 Applications Runtime View

When a production monitor node is selected, the **Applications** runtime view displays a list of COM+ applications currently being monitored along with process metrics and operational status for each. The metrics are updated every 60 seconds by default and can be sorted by clicking a column heading in the list.

All Transactions Runtime View

The ***All Transactions***, ***Transactions***, and ***Components*** views share common features. Each of them report metrics related to the current interval and those collected in the previous interval: Metrics for the most recent monitor interval are presented in the area labeled *This Session*, and metrics from the previous interval are presented in the area labeled *Last Interval*.

Metrics with configured thresholds display green, yellow, or red backgrounds based on their value with respect to their designated threshold settings.

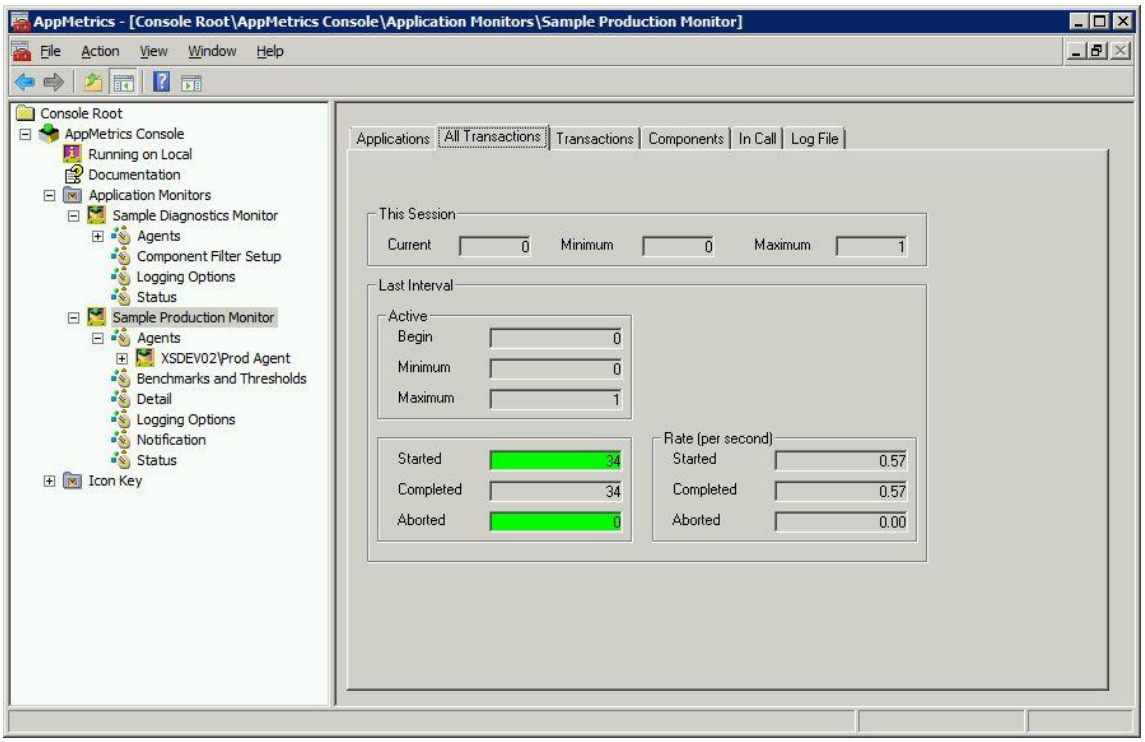


Figure 3-17 All Transactions Runtime View

The ***All Transactions Runtime View*** displays summary metrics from all transactions detected for all monitored applications configured for the current monitor.

Transactions Runtime View

The ***Transactions Runtime View*** displays metrics related to specific user transactions, selectable from a drop down list.

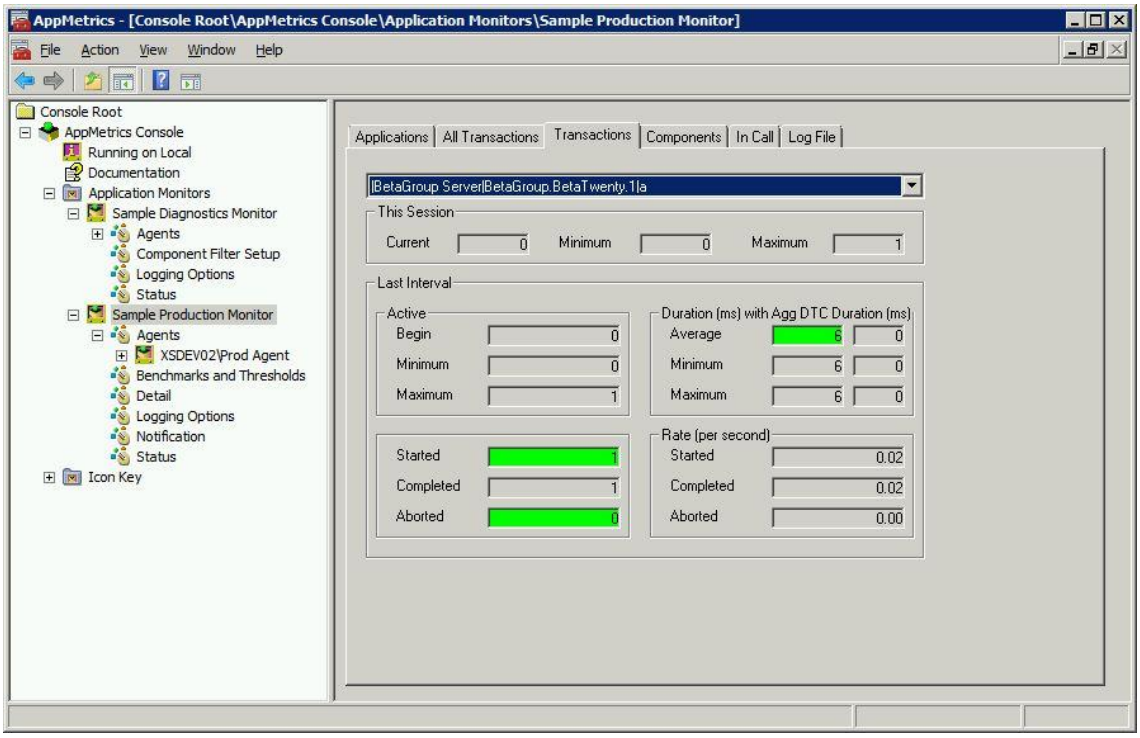


Figure 3-18 Transactions Runtime View

Transactional Detail Level Setting

In AppMetrics, user transactions are displayed in the **Transactions** runtime view depending on the current setting for the monitor's *Transactional Detail Level* setting. *Transactional Detail Levels* are defined in Table 3-3 on Page 39. Based on this setting, which is configurable on the ***Transactional Detail Level Configuration View*** (Page 37), the ***Transactions Runtime View*** will display data as follows:

Transaction Detail Level	Transactions Runtime View Behavior
Aggregate Only	No data is displayed. Data will continue to be collected and displayed on the ***All Transactions*** runtime view however.
Identify by Component	User transactions are based upon root components detected by the monitor. Metrics related to those transactions are displayed.
Identify by Component and Method	User transactions are based upon root methods and components detected by the monitor. Metrics related to those transactions are displayed.

Table 3-1 Transactional Runtime View Behavior

Detail Throttling Attributes

In addition to the monitor's *Transactional Detail Level* setting, the ***Transactions Runtime View*** also depends on the **Detail Throttling** settings on the ***Transactional Detail Level Configuration View*** (Figure 3-32). When detail throttling is enabled, the monitor stops collecting and displaying data for transactions. For more information, see *Detail Throttling for COM+ Production Monitors* on page 3-40.

User Transactions

In AppMetrics, a ***user transaction*** starts when a client process makes a call into a component. In turn, the instance of the called component becomes the root object. All the work to complete the transaction is performed within the context of this root object.

The root object may make further calls into other components, but when these latter calls return, the user transaction is not yet complete. The user transaction is complete only when the initial call returns.

As an illustration, consider a transaction for crediting a bank account. Assume the application has a component called "Account", and the component itself has a method called "Credit". The transaction starts when a client process calls the Credit method. This call results in an instance of the Account component. The instance serves as the root object for the transaction.

When performing the transaction, the root object may make a variety of calls, some of which may be to other components. For example, the root object may call a method named "Balance" to obtain the current balance in the account. Once these subsequent calls return, and the initial Credit method on the root object also returns, the transaction is complete.

User Transactions and DTC Transactions

As a note, user transactions have no relationship to DTC transactions in COM+. A user transaction may or may not be set as transactional in the DTC. Thus, a user transaction in AppMetrics is not the same as a DTC transaction.

Components Runtime View

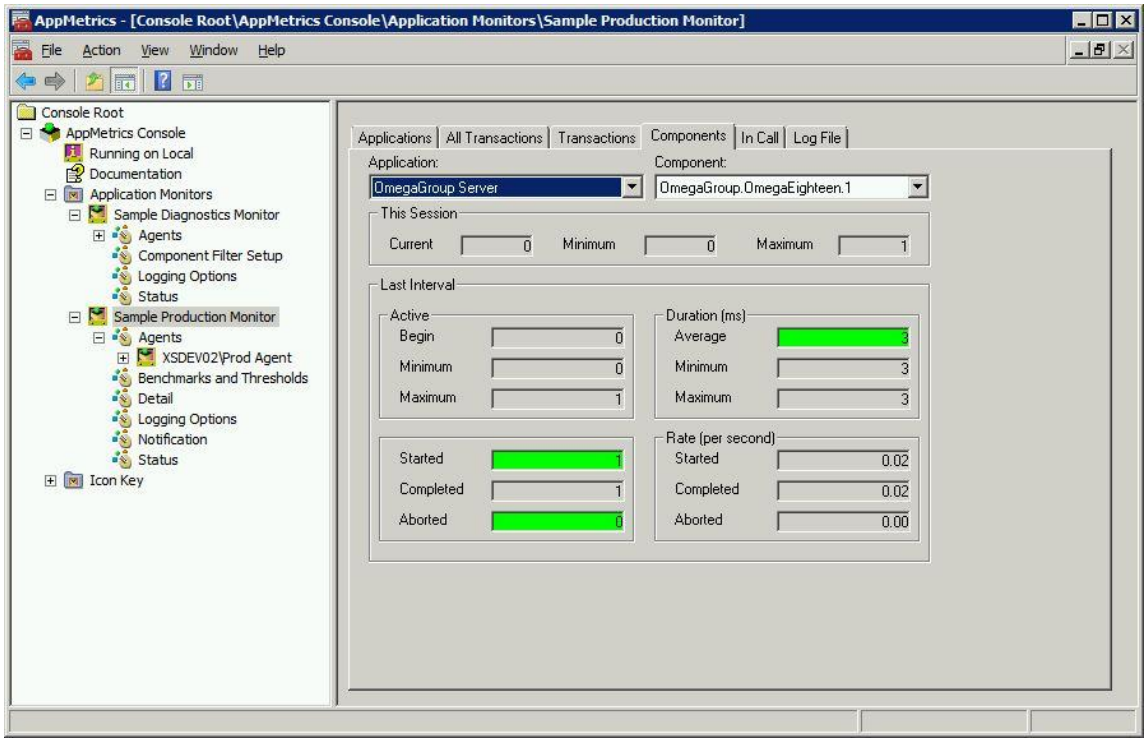


Figure 3-19 Components Runtime View

The ***Components Runtime View*** enables the user to select the metrics to be displayed by component. The **Application** drop-down list contains the names of all the server applications that were selected in the ***COM+ Applications Configuration View***. The **Component** drop-down list contains the names of all the components discovered during the prior AppMetrics monitor session within the selected server application. The **Component** drop-down may also contain names of components within library applications if they were called within the context of the selected server application.

Note: If *Component Filtering* is set to **No Components** in the ***Transactional Detail Level Configuration View*** (Figure 3-32), then AppMetrics will not display component metrics. Additionally, if **Detail Throttling** is triggered then no data will be available for this view. For more information, see **Detail Throttling for COM+ Production Monitors** on Page 3-40.

In Call Runtime View

The ***In Call Runtime View*** shows all active applications, components, and methods in a tree view. *Applications* are listed first, and can be expanded to display the component and method names.

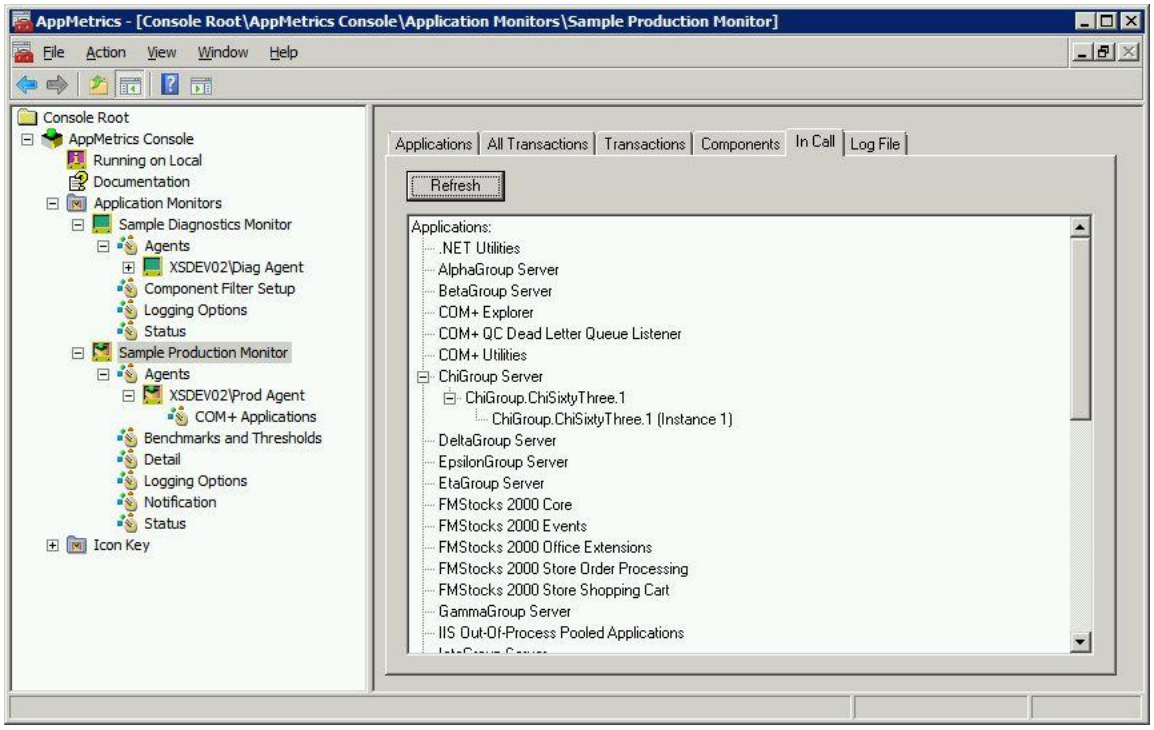


Figure 3-20 In Call Runtime View - Applications

This view is useful when searching for stalled applications or components, as the *Active Transactions* list at the bottom of the view can be expanded to display the method start time (Figure 3-21).

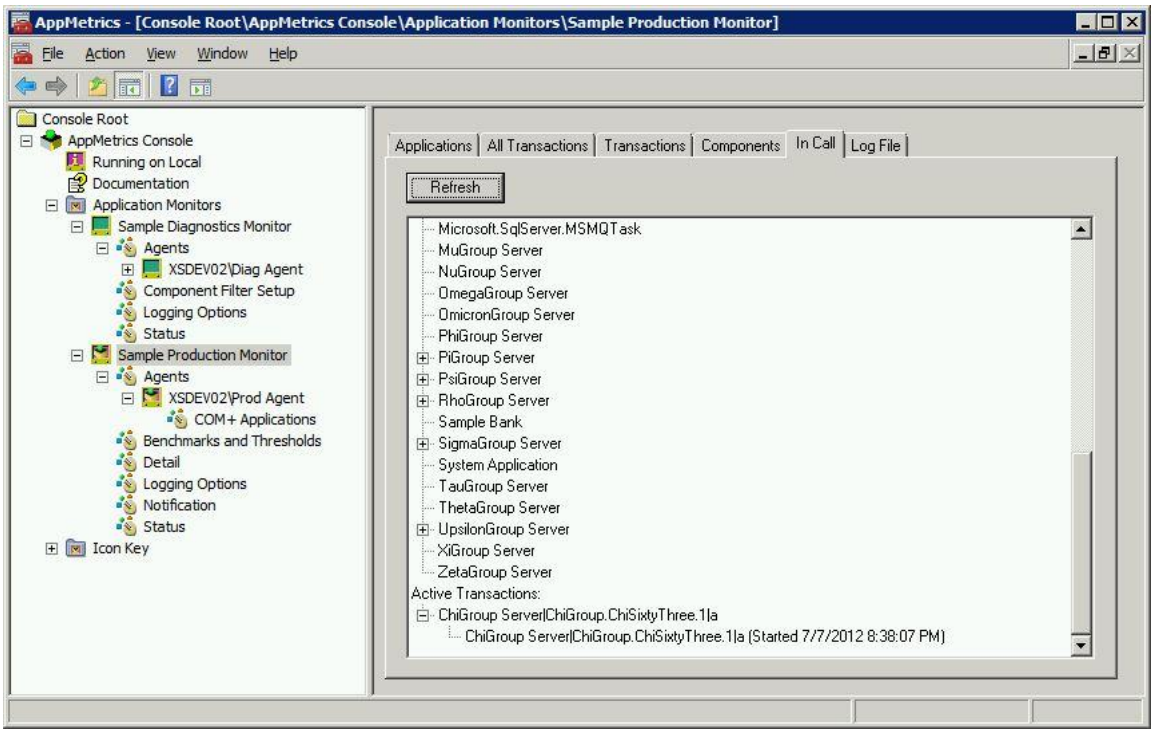


Figure 3-21 In Call Runtime View - Active Transactions

To update the display, click the **Refresh** button.

Note: If detail throttling is in effect, the ***In Call View*** will display no data within *the Active Transactions* list. For more information, see **Detail Throttling for COM+ Production Monitors** on Page 3-40.

Log File Runtime View

The Log File Runtime view displays the directory to which the metrics are currently being logged. Additionally, this view contains the **Upload the AppMetrics Log Files to the Database Now** button which as the name implies, enables users to upload data to the database immediately, and at the same time direct new data to a different Series folder on demand.

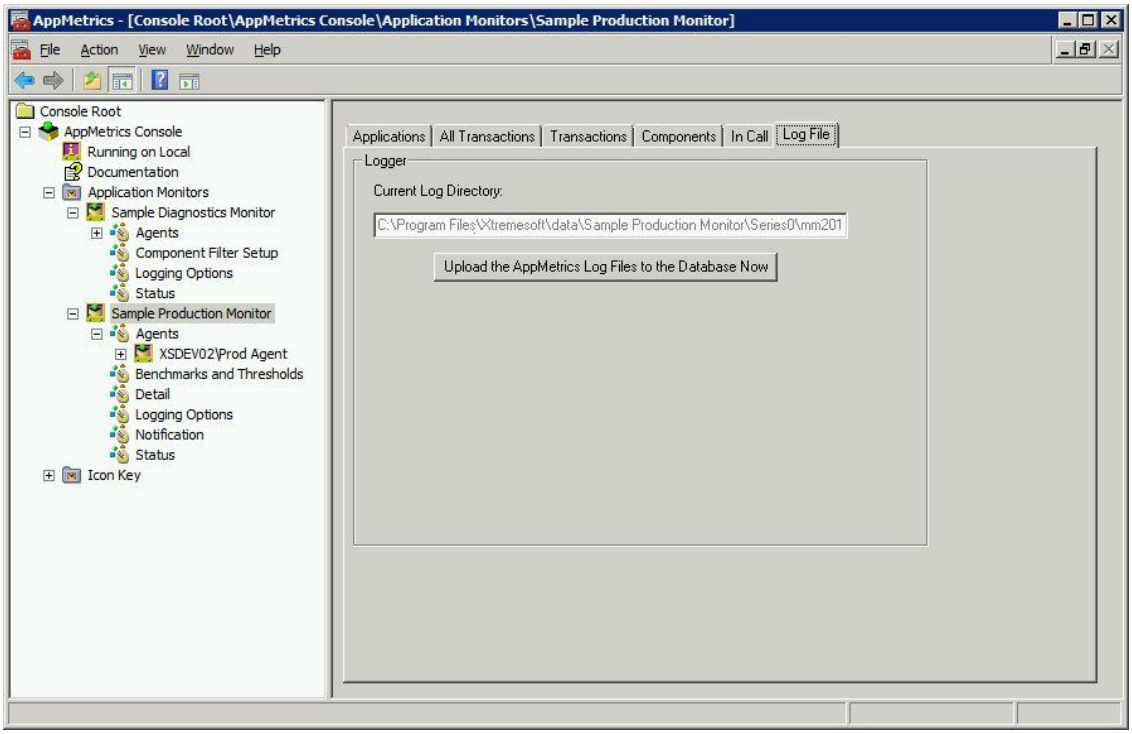


Figure 3-22 Log File Runtime View

The logging feature in production monitors works much the same way as it does in diagnostics monitors. Although the log files for these monitors have different formats and contain different metrics, the operation and user interface in both cases is the same.

The ***Upload the AppMetrics Log File to the Database Now*** button is for situations where the user wishes to diagnose an event that has just occurred in a monitored application. The button immediately uploads the data to the database so that reports can be run to examine the new data.

The log file directory pathname is determined from the following data:

- The data directory specified during the AppMetrics install
- The monitor name
- A series number incremented when the **Upload the AppMetrics Log Files to the Database Now** button is used
- A subdirectory name derived from the date and time

Benchmarks and Thresholds Views

Each of the following configurations view enables users to specify benchmarks and thresholds for a specific set of metrics. These threshold levels are associated with green, yellow or red colors, which highlight the metrics when they reach the given threshold level. A benchmark value can be typed directly into the **Current Benchmark** field for any of the metrics; optionally, AppMetrics will copy the last recorded value (the value of the metric when the monitor was last stopped) into the **Current Benchmark** when the user clicks Copy to Current.

Thresholds are set as a percentage of the benchmark. When the metric reaches the percentage indicated in the **Warning Level** field, the metric's background turns yellow in the associated runtime view. When the metric reaches the specified percentage in the **Notification Level** field, the metric's background color turns red in the associated runtime view. Furthermore, if the monitor is configured for notifications, AppMetrics will send a notification. For more information about configuring notifications for a monitor, see **Notification Configuration View** on Page 3-25.

Note: If the *Current Benchmark* value for a metric is set to zero, then notifications for that metric will not be sent.

As an example of how benchmarks and thresholds work together, refer to the ***All Transactions Configuration View*** in Figure 3-23 and the ***All Transactions Runtime View in*** Figure 3-17. In the configuration view, the **Current Benchmark** value for the *Number Started* metric is 100. When the *Number Started* metric in a given interval exceeds 50 (50% of the 100 Current Benchmark value), the background of the *Started* metric in the runtime view will turn yellow.

When the metric exceeds 75 (75% of 100), the background of the Started metric in the runtime view will turn red (see the *Started* metric in the lower-left corner of the runtime view in Figure 3-17), and a notification will be delivered to all configured recipients.

The rest of this section describes the specifics for the ***All Transactions***, ***Transactions***, and ***Components*** views.

All Transactions Configuration View

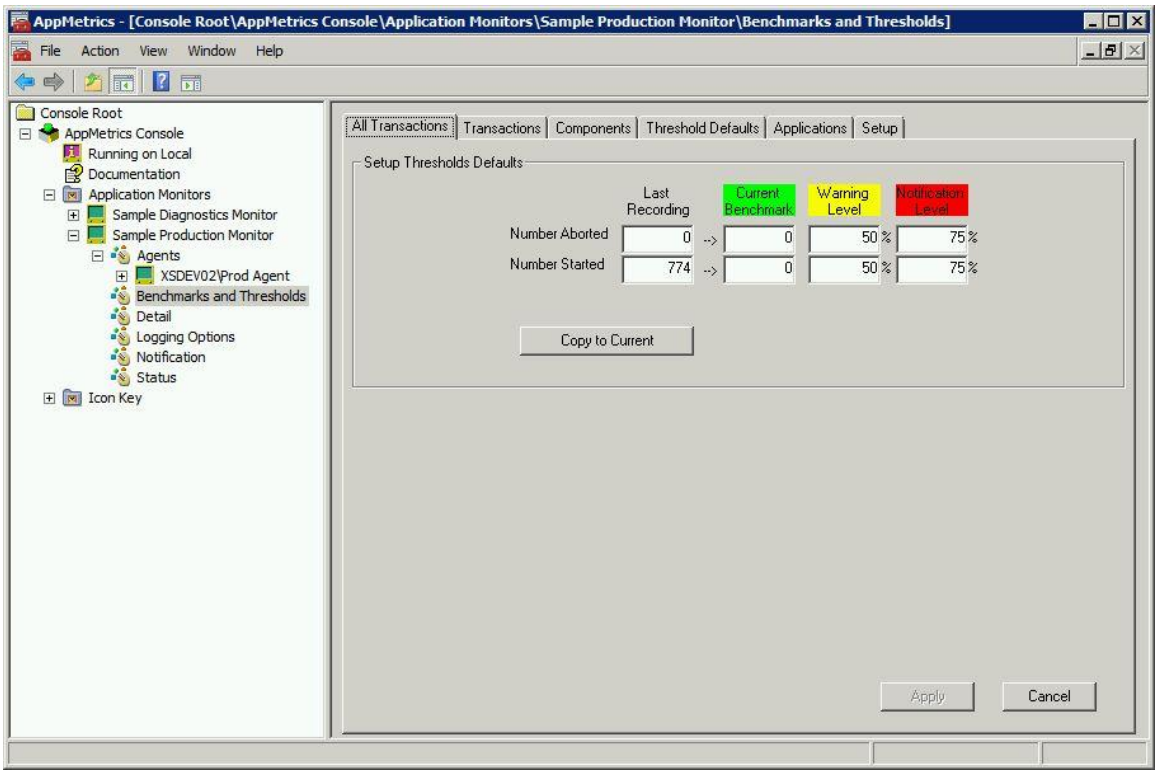


Figure 3-23 All Transactions Configuration View

This view enables the user to set thresholds related to all transactions detected across all applications monitored by the AppMetrics monitor.

Transactions Configuration View

In addition to enabling entry of transaction threshold values, this view enables users to assign business function names to specific calls to components or methods, depending on the transaction detail level selected.

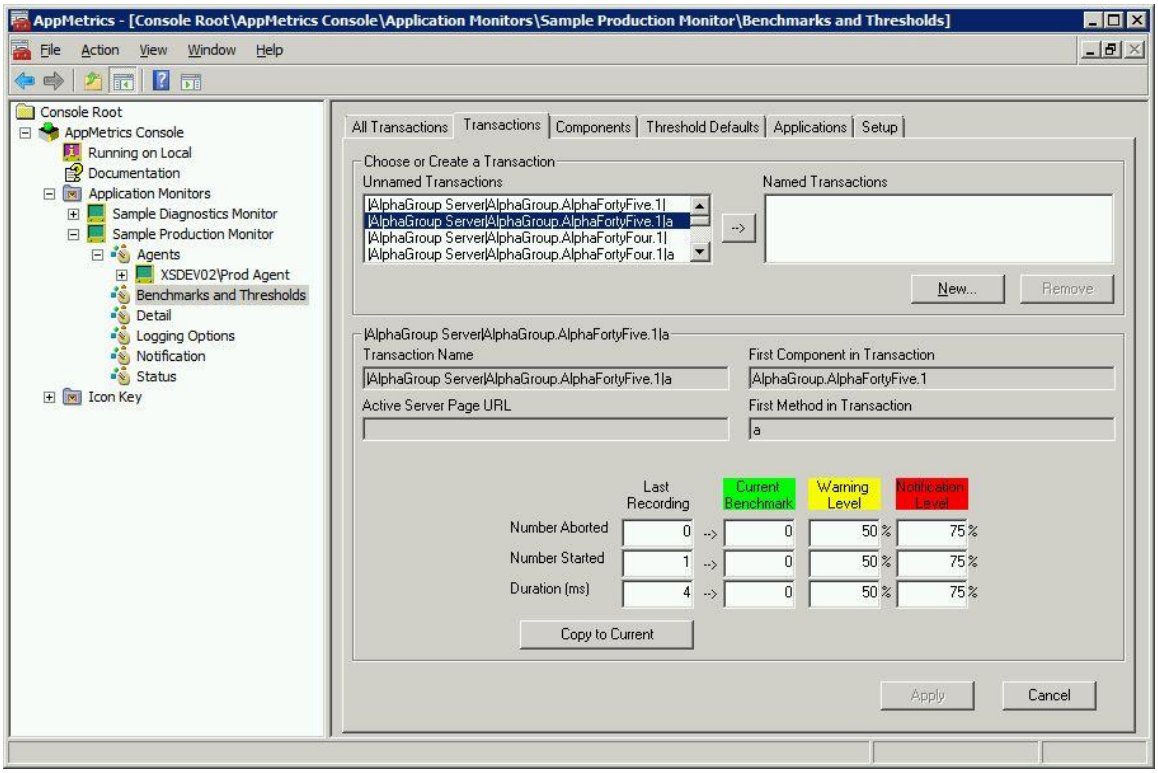


Figure 3-24 Transactions Configuration View

Named and Unnamed Transactions

AppMetrics enables you to assign easily recognizable names to transactions. Transaction names may be assigned using the ***Transactions Configuration View***. Names assigned to transactions appear in the ***Transactions Runtime View*** and AppMetrics reports.

If AppMetrics detects a component that results in a root object, but no transaction name is assigned to it, AppMetrics will list the component as an unnamed transaction.

Based upon the **Transactional Detail Level** setting, discovered transactions will have a default name format as described below.

- **Component-Level Detail**

  For component calls detected when the **Transactional Detail Level** setting is *Identify by Component*, AppMetrics identifies discovered transactions using the following name format:

  |*Application*|*Component*|

- **Method-Level Detail**

  For component calls detected when the Transactional Detail Level setting is *Identify by Component and Method*, AppMetrics identifies discovered transactions using the following name format:

  |*Application*|*Component*|*Method*|

Naming a Discovered Transaction

When a monitor detects new transactions, it will list these transactions in the **Unnamed Transactions** list. Users can assign business function names to these unnamed transactions.

Notes:

- The monitor must be stopped before you can name a transaction.
- This function is only available when the *Transactional Detail Level* Configuration View(Figure 3-32) is set to either *Identify by Component* or *Identify by Component and Method*.

To assign a name to an unnamed transaction:

1. In the **Unnamed Transactions** list, select the transaction to be named.
2. Click the button labeled **->**. The **Name Transaction** dialog opens.

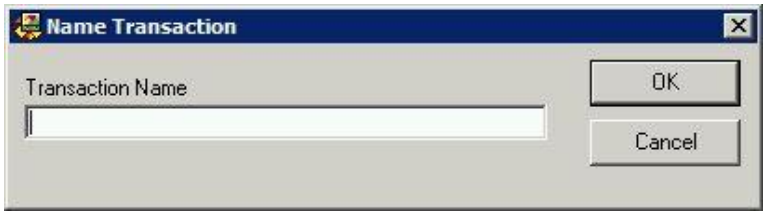


Figure 3-25 Name Transaction Dialog

3. In the **Transaction Name** field, specify the business-function name to describe this transaction.
4. Click **OK**. This adds the transaction to the **Named Transactions** list.
5. If desired, you can assign benchmark, warning, and notification levels to the newly named transaction. Click the transaction in the **Named Transactions** list, specify the values at the bottom of the view, and then click **Apply**.

Adding a New Named Transaction

Some users may need to name a transaction based on a component that the monitor has not yet detected. Others may need to name a transaction before running the monitor. The components for these transactions will not be listed in the *Unnamed Transactions* list. As a result, the previous procedure cannot be used to name these transactions. However, by using the **New Transaction** function, users can access a list of components that the system recognizes in the monitored applications. For any of these listed components, the user can create a named transaction.

Notes:

- The monitor must be stopped before you can name a transaction.
- This function is only available when the *Transactional Detail Level* (in the ***Transactional Detail Level Configuration View***, Figure 3-32) is set to either *Identify by Component* or *Identify by Component and Method*.
- If a transaction occurs in an application that has the same name as another application, AppMetrics does not let you configure monitoring for this application. As a result, you will be unable to configure monitor settings for any transactions in this application.

To add a new named transaction:

1. Beneath the **Named Transactions** list on the ***Transactions Configuration View*** (Figure 3-24), click **New**.

   The **New Transaction** dialog opens.

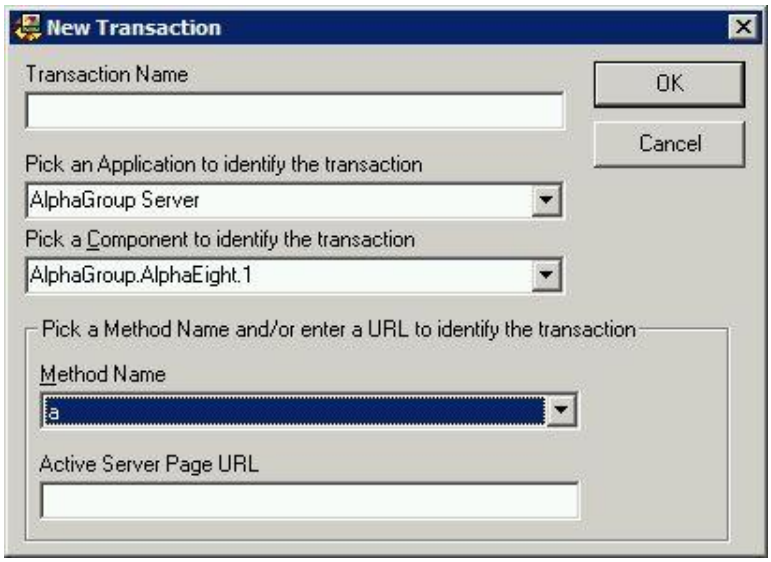


Figure 3-26 New Transaction Dialog

2. In the **Transaction Name** field, specify the business-function name to describe the transaction.
3. In the **Pick an Application to identify the transaction** field, select the application or package that starts the business transaction.
4. In the **Pick a Component to identify the transaction** field, select the component that starts the business transaction.

5. Perform one of the following:
   - If available, select the **Method Name** that starts the business transaction.

     - Or -

   - In the **Active Server Page URL** field, specify the ASP page that calls the method in the component to initiate the transaction.
6. Click **OK**. This adds the transaction to the **Named Transactions** list.
7. You can now assign benchmark, warning, and notification levels to the newly named transaction. Click the transaction in the **Named Activities** list, specify the values at the bottom of the view, and then click **Apply**.

Components Configuration View

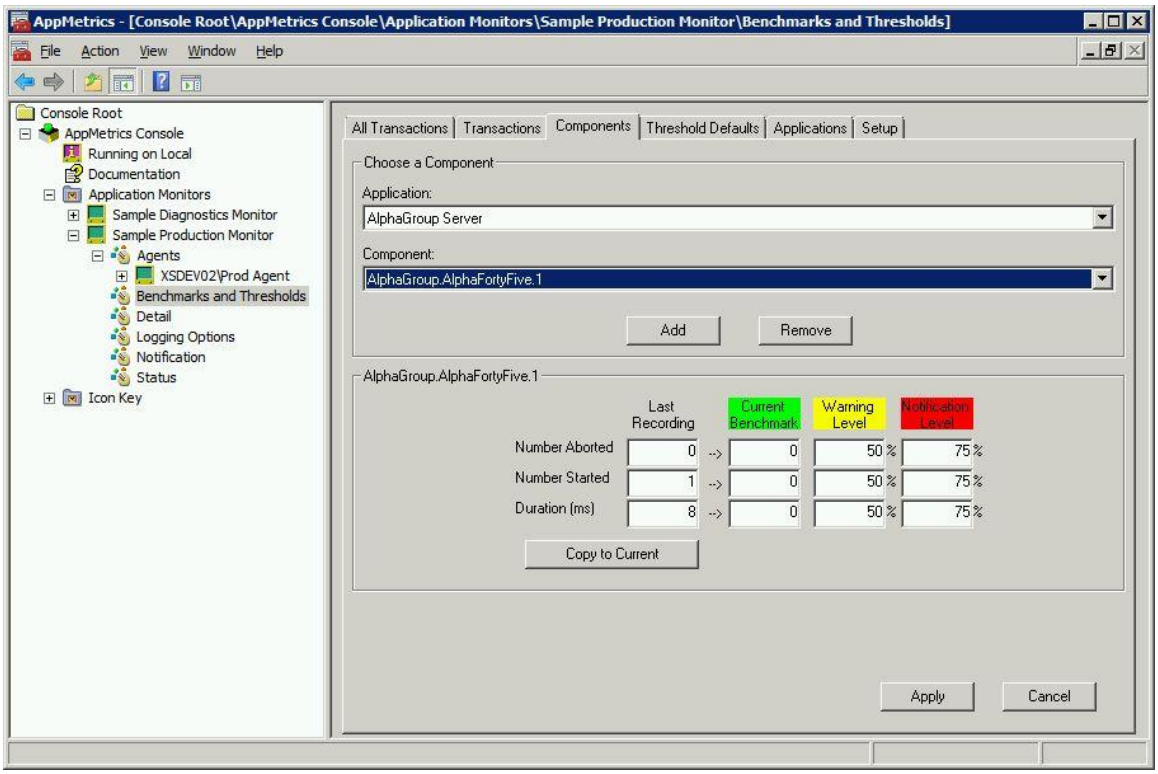


Figure 3-27 Components Configuration View

The ***Components Configuration View*** enables the user to set benchmarks and thresholds for components of interest. Components are selected from the drop-down list boxes at the top of the view. The **Application** drop-down list contains the server applications selected in the ***Configuring an Agent***

Once an agent has been added to the new monitor it will need to be configured by selecting which applications to monitor on that server. The ***COM+ Applications Configuration View*** is accessible by clicking the *COM+ Applications* node located under a monitor's associated agent node (Figure 3-14).

COM+ Applications Configuration View (Figure 3-14).

The **Component** drop-down list contains all the components that were active within the selected server application during the last run of the monitor.

By clicking the **Add** button, the user can select from the components that are included in the selected server application but have not been recently active. By clicking **Remove**, the user can delete the settings for the currently selected component.

Notes:

- If the *Component Filtering* is set to **No Components** in the ***Transactional Detail Level Configuration View*** (Figure 3-32), no component metrics can be configured.

The user can either define the benchmarks for a given component by typing them into the *Current Benchmark* fields, or by using the *Last Recording* values by clicking the **Copy to Current** button.

The *Threshold* levels are percentages of the *Current Benchmark*. The monitor provides default values that the user can override on a per-component basis.

The monitor will change the background color of the metric on the runtime views when the value exceeds the *Warning* level percentage of the current benchmark. It will change the background color again when the value exceeds the *Notification Level* percentage. When this occurs, a notification will be sent if AppMetrics **Notifications** have been configured. See the **Notification Configuration View** on page 3-44.

Notes:

- If the *Current Benchmark* value for a metric is set to zero, then notifications for that metric will not be sent.
- The *Duration (ms)* values apply to all component instances that have been configured and detected by AppMetrics. This applies to both active as well as completed instances. However, the ***Component Runtime View*** and AppMetrics Reports apply only to *completed* (i.e. released/destroyed) instances.

The **Component** drop-down box may also list components of library applications if they were called within the context of the selected server application. For each of these components, AppMetrics lists its name, followed by the name of the library application in parentheses. In the example below (Figure 3-28), for the component *FMStore_Events.ShoppingCart*, the library application is *FMStocks 2000 Events*, thus AppMetrics will list the component as *FMStore_Events.ShoppingCart (FMStocks 2000 Events).*

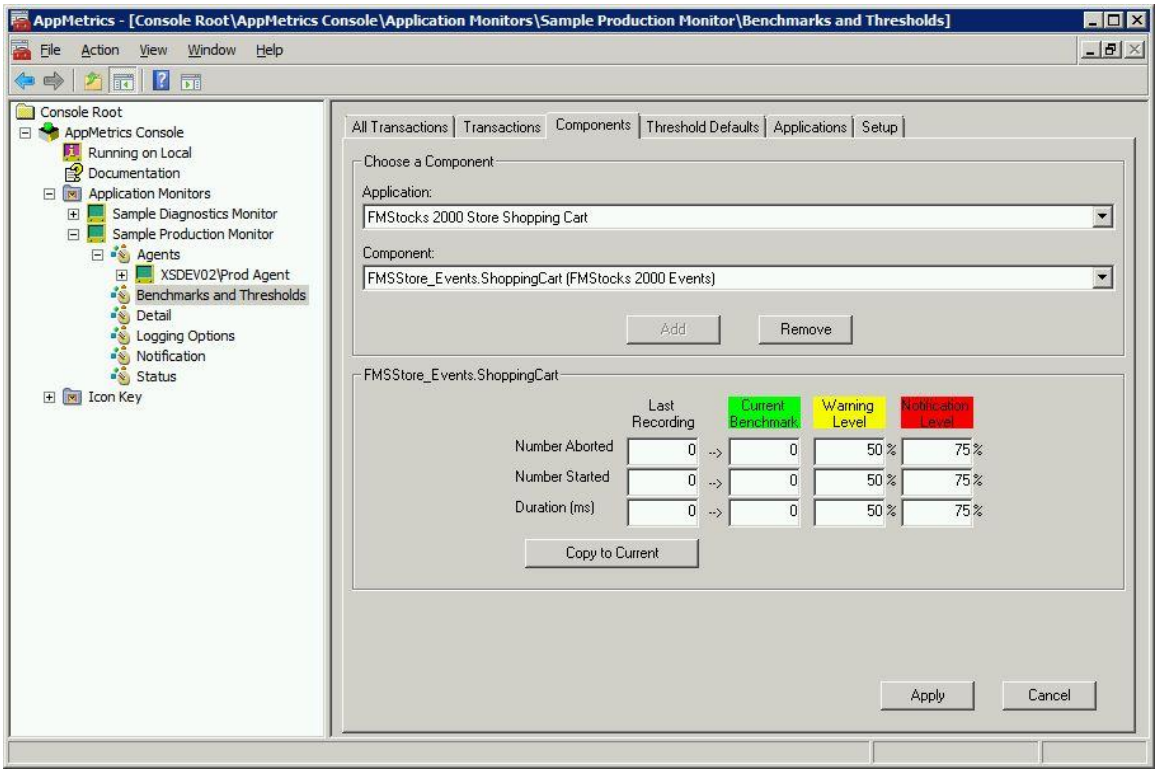


Figure 3-28 Library Application Components

Threshold Defaults Configuration View

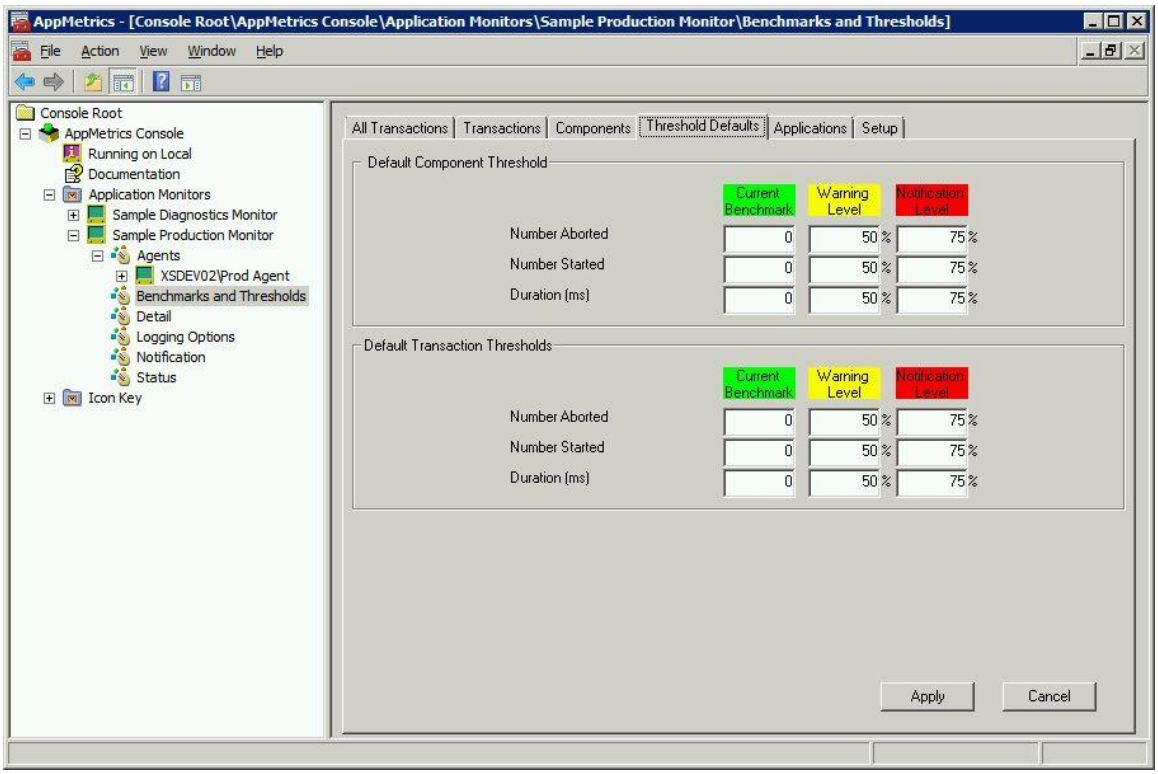


Figure 3-29 Threshold Defaults Configuration View

These values are assigned to any component or transaction, respectively, until the user explicitly overwrites them in the ***Components*** or ***Transactions*** configuration views.

Note: If the *Current Benchmark* value for a metric is set to zero (default setting), then notifications for that metric will not be sent unless overridden on the respective configuration views.

Applications Thresholds Configuration View

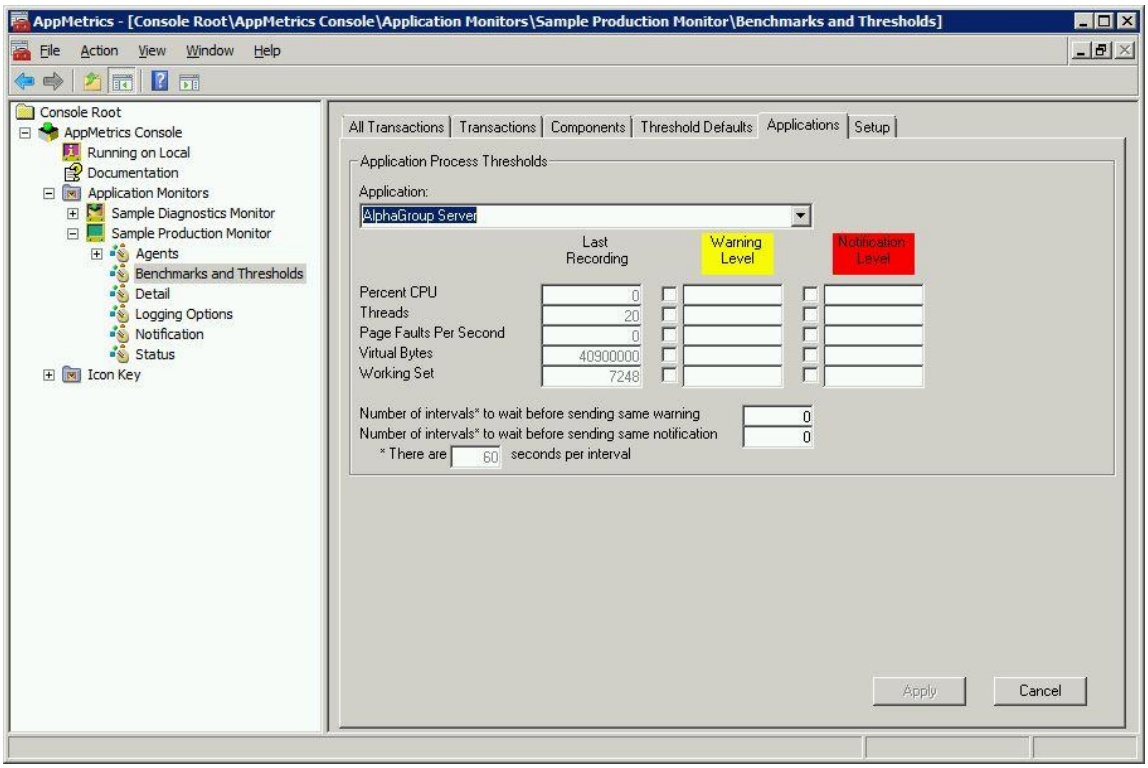


Figure 3-30 Applications Thresholds Configuration View

The ***Applications Thresholds Configuration View*** is used to configure thresholds for a specified application's use of system resources. When a COM+ application meets or exceeds specified levels of resource usage, AppMetrics will deliver a notification of the event if configured to do so.

System Resource	Description
Percent CPU	Percentage of CPU time that was consumed by the threads of the applications during the interval. This value can exceed 100% if more than one processor is used on the computer.
Threads	The concurrent number of active objects that carry out instructions on behalf of the applications.
Page Faults Per Second	The rate in which page faults occur within the threads of the applications. A page fault occurs when a thread tries to access a virtual memory page that does not belong to the process working set in main memory.
Virtual Bytes	The virtual memory used for address space by the process.
Working Set	The current number of bytes in the Working Set for the applications. The Working Set is the group of memory pages that were most recently accessed by the threads in the applications.

Table 3-2 Application Thresholds

Application Thresholds

Application thresholds are similar to those for the *All Transactions*, *Transactions,* and *Components* views. However, they differ in several ways:

- An application process threshold uses an actual value instead of the percentage of a benchmark value. For example, if the warning level for the *Working Set* is specified as 7248, and the application actually uses this resource in excess of 7248, then the warning will be triggered. This differs from the various transaction and component thresholds, which use both a benchmark and a percentage of that benchmark to serve as the threshold value.
- When a warning or threshold level is reached for a system resource, it does not change the color of that system resource in the ***Applications Runtime View*** (Figure 3-16).
- The transactions and components thresholds only send notifications when the *Notification* level is met, whereas the application process thresholds can send notifications when either the *Warning* or *Notification* levels are met.

To configure a warning or notification level for an application's usage of a specific system resource:

1. Click the drop-down arrow in the *Applications* field and then select the desired application in the drop-down list.

   Notes:

   - The applications listed in this field correspond to the ones selected in the **Configuring an Agent**
   - Once an agent has been added to the new monitor it will need to be configured by selecting which applications to monitor on that server. The ***COM+ Applications Configuration View*** is accessible by clicking the *COM+ Applications* node located under a monitor's associated agent node (Figure 3-14).
   - COM+ Applications Configuration View (Figure 3-14) for the monitor.
   - If an applications shares the same name with other applications on the computer, these applications are not available in this drop-down list.

2. For the desired resource, click the check box to the left of the warning/notification level. This copies the *Last Recording* value into the field.

   Note:

   - The last recording is the application's recorded usage of the given resource when the monitor was last stopped.

3. Accept the default value copied from the *Last Recording* field or type the preferred value.
4. In **the Number of intervals to wait before sending same warning** or the **Number of intervals to wait** before sending same notifications field, specify the preferred interval.

   Note:

   - This setting applies to all the threshold levels for the currently selected application.

5. Click **Apply** to save the changes.

Setup Configuration View

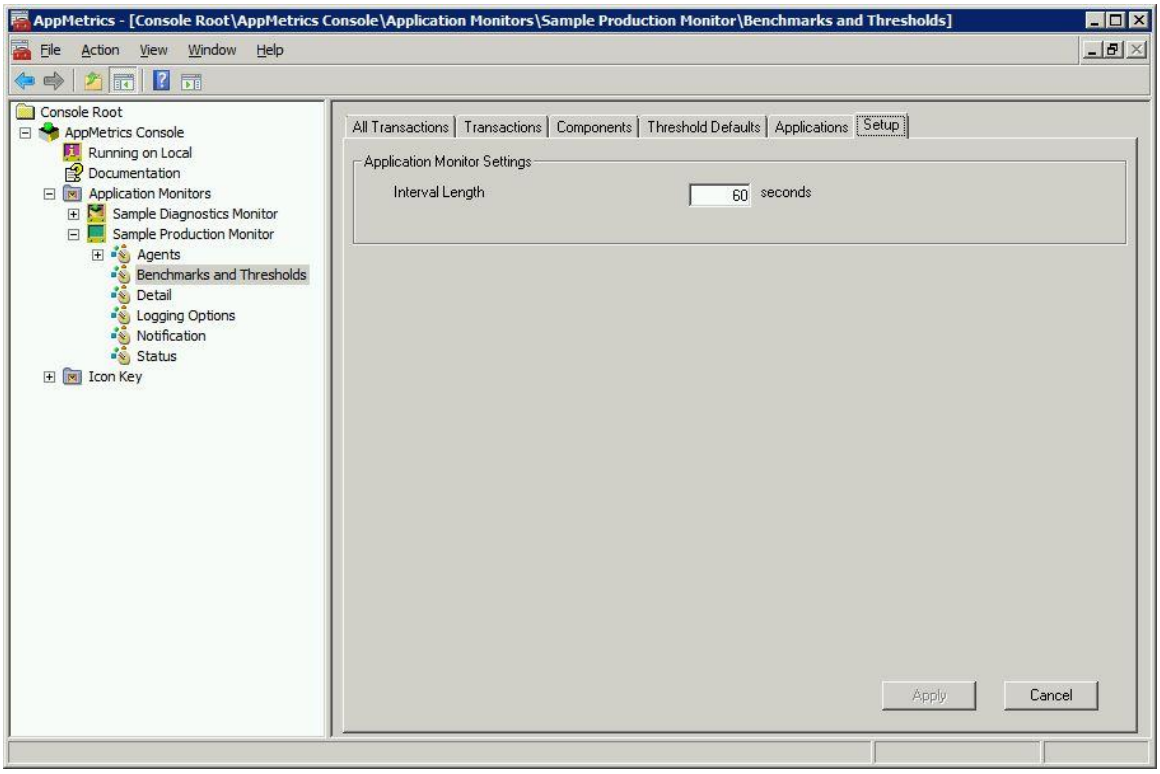


Figure 3-31 Setup Configuration View

The ***Setup Configuration View*** enables users to specify the duration of the interval in which the *All Transactions*, *Transactions*, and *Components* metrics are calculated. The minimum interval is 10 seconds.

This setting differs from the *Check Application Status* setting in the **Configuring an Agent**

Once an agent has been added to the new monitor it will need to be configured by selecting which applications to monitor on that server. The ***COM+ Applications Configuration View*** is accessible by clicking the *COM+ Applications* node located under a monitor's associated agent node (Figure 3-14).

COM+ Applications Configuration View (Figure 3-14).

Transactional Detail Level Configuration View

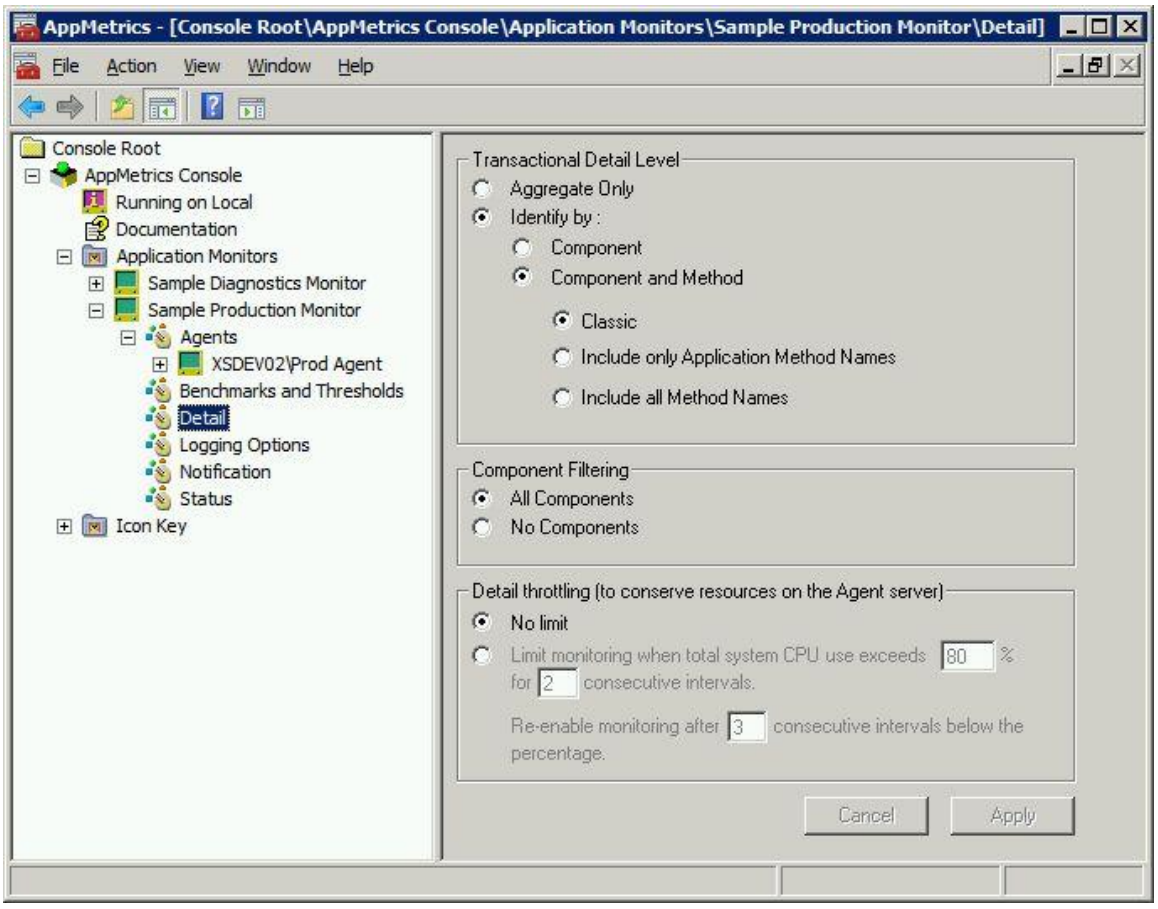


Figure 3-32 Transactional Detail Level Configuration View

The ***Transactional Detail Level Configuration View*** is used to control the transactional detail level that AppMetrics will use to collect data for monitored applications.

Transactional Detail Levels

The following table lists the different transaction detail levels and their associated behavior.

Transactional Detail Level	Detail Setting to Enable this Level of Monitoring	Affects the Following Items:	Description
All Transactions	None. Always monitored, except during a throttle down. See **Detail Throttling for COM+ Production Monitors** on Page 3-40	• All Transactions runtime and configuration views • All Transactions Report	This sums the total number of user transactions that take place during the interval.
Aggregate Only	**Transactional Detail Level** area → **Aggregate only** setting	• Transactions runtime and configuration views • Transactions Report	This stops monitoring for individual user transactions. It prevents AppMetrics from showing data in the Transactions views and report. However, monitoring for All Transactions (previous item) will continue.
Transactions Identified by Component	**Transactional Detail Level** area → **Identify by Component** setting	• Transactions runtime and configuration views • Transactions Report	All user transactions will be monitored, and each individual transaction will be named by its root component.
Transactions Identified by Method	**Transactional Detail Level** area → **Identify by Component and Method** setting	• Transactions runtime and configuration views • Transactions Report	This identifies the root method and component in each user transaction. This selection requires one of three further sub-selections, described below.
Transactions Identified by Component and Method **No Method Names in Late-Bound Calls Available**	**Transactional Detail Level** area → **Identify by Component and Method** setting → **Classic** sub-setting	• Transactions runtime and configuration views • Transactions Report	This identifies the root method and component in each user transaction. However, if an ASP page called the method, AppMetrics will not identify the name of the method, it will identify the URL of the ASP page instead.

Transactional Detail Level	Detail Setting to Enable this Level of Monitoring	Affects the Following Items:	Description
Transactions Identified by Component and Method **Include all Method Names**	**Transactional Detail Level** area → **Identify by Component and Method** setting → **Include all Method Names** sub-setting	• Transactions runtime and configuration views • Transactions Report	This identifies the root method and component in each user transaction. AppMetrics will report on any user transaction started through *Invoke*, using the actual method name instead of Invoke. However, it will also report other *IDispatch* methods such as *GetIDsOfNames*, etc. Use this option if you want to see symmetrical counts between transactions and their related components.
Transactions Identified by Component and Method **Include only Application Method Names**	**Transactional Detail Level** area → **Identify by Component and Method** setting → **Include only Application Method Names** sub-setting	• Transactions runtime and configuration views • Transactions Report	This identifies the root method and component in each user transaction, but with the limitations described below. AppMetrics will report on any user transaction started by *Invoke*, but it will name the transaction after the method that *Invoke* actually calls. This lets you see the actual method name, not the Invoke name. AppMetrics will exclude other *IDispatch* methods, such as *GetIDsOfNames*.
All Components	**Component Filtering** area → **All Components** setting	• Components runtime and configuration views • Components Report	This generates metrics of all the components that were called in the monitored packages/ applications.
No Components	**Component Filtering** area → **No Components** setting	• Components runtime and configuration views • Components Report	This stops monitoring of called components. It prevents AppMetrics from showing data in the Components views and report.

Table 3-3 Transactional Detail Levels

Switching Between Transactional Detail Levels

When **Identify by Component** is selected, AppMetrics will identify transactions using component names discovered by the monitor. Those names will appear in the following lists:

- The drop-down list in the ***Transactions Runtime View*** (Figure 3-19).
- The *Unnamed Transactions* list in the **Transactions Configuration View** (Figure 3-24). If *Identify by Component and Method* is selected, the component-method names are listed in these same transactions lists.

If you use *Identify by Component* at first, and then subsequently switch to *Identify by Component and Method*, the component-only names will continue to be listed in the transactions lists along with the newly detected component-method names. However, no metrics will be generated and displayed in the ***Transactions Runtime View*** for the component-only transaction names since AppMetrics is generating metrics for the component-method transaction names.

This also applies to switching from *Identify by Component and Method* to *Identify by Component*. In this case, metrics will only be generated and displayed for the component-only names. Although the component-method names will continue to be listed, AppMetrics will not generate and display metrics for them as long as the detail level remains set to *Identify by Component*.

AppMetrics retains transaction information associated with the non-active detail level so that configurations for them are preserved. As a result, you can quickly return to using those transactions again with their previous configurations intact.

Detail Throttling for COM+ Production Monitors

Using the detail throttling feature, a *throttling threshold* may be specified to set the maximum CPU% usage allowed across all processors on a monitored computer. If the CPU% usage exceeds the throttling threshold, AppMetrics reduces the monitor's impact on system resources. This releases system resources to the COM+ applications and other processes on the computer.

Effects of Detail Throttling

When AppMetrics throttles-down a monitor the following actions occur:

- The monitor continues collecting resource usage data for monitored applications, but if the monitor was set to collect data about components and transactions, then AppMetrics disables monitoring for these items.
- The runtime views do not update the component and transaction data. Additionally, the ***In Call Runtime View*** (Figure 3-20) provides no entries in the *Active Transactions* list.
- AppMetrics logs the following message to the Windows Application Event Log: "'Monitor name' limited monitoring due to low resources on 'Computer name'."
- AppMetrics stops all data logging (including resource usage data). Hence, any report that includes the time when detail throttling was in effect will contain no data for that segment of time.

Restoring Full Monitoring

After AppMetrics throttles down a monitor, it waits for the CPU usage to return to lower levels. When the CPU usage remains below the *Throttling Threshold* after a certain number of consecutive intervals, AppMetrics restores full monitoring. In particular, the following items occur:

If the monitor was set to collect data for components and transactions, then AppMetrics resumes monitoring for these items.

The runtime views begin updating component and transaction data. Likewise, the ***In Call Runtime View*** provides entries in the *Active Transactions* list.

Once full monitoring has been restored, AppMetrics will log the following message to the Windows Application Event Log:

"'*Monitor name*' resumed normal monitoring."

Detail Throttling Settings for COM+ Production Monitors

The table below describes the settings for **Detail Throttling for COM+ Production Monitors** (see Figure 3-32) and their effect:

Item	Description
No limit	This disables detail throttling.
Limit monitoring when total system CPU use exceeds *x* % for *y* consecutive intervals.	This enables detail throttling to begin based on the combination of the following settings: ***x*** = The Throttling Threshold, which is the maximum CPU% usage allowed across all processors on the computer before a throttle down is triggered. Valid values for this setting range from **40** through **90**. AppMetrics will not accept values outside of this range. **Note:** The CPU% usage is the average level of usage during an interval. ***y*** = The number of consecutive intervals that must occur before a throttle down is triggered. The interval is based on the specified **Interval length** in the monitor's Setup configuration view (Figure 3-31).
Re-enable monitoring after *z* consecutive intervals below the percentage.	When the previous item is selected, this item controls when normal levels of monitoring may begin again. If the CPU% usage for each monitored application remains below the Throttling Threshold during ***z*** consecutive intervals, then the monitor resumes the tasks that were stopped

Table 3-4 Detail Throttling Settings

Logging Options Configuration View

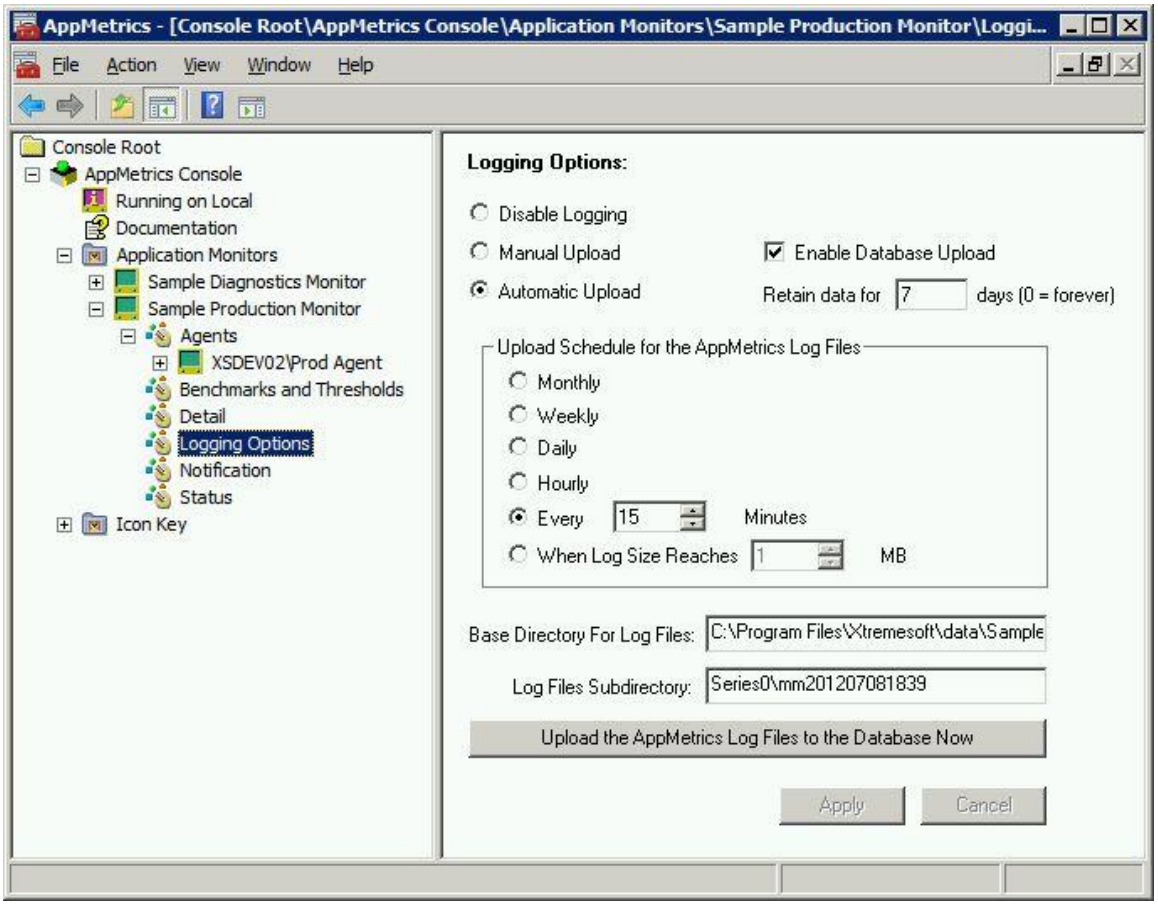


Figure 3-33 Logging Options Configuration View

AppMetrics incorporates automatic uploads of collected data to a SQL Server database. The upload can be scheduled to occur periodically, or be triggered when any individual log file exceeds a specified size.

Metrics are directed into separate files for easy upload into databases. If SQL Server is available on the system, then the **Enable Database Upload** check box will be available. The ***Retain data for x days*** field controls how long the data will be kept in the database.

The monitor's base log file directory can be changed, if desired, by changing the path in the **Base Directory for Log Files** field.

The **Upload the AppMetrics Log Files to the Database Now** button will upload the log files to SQL Server, and create a new *Series* folder for the next set of log files. It is identical to the button of the same name mentioned in the ***Log File Runtime View***.

The ideal upload schedule depends upon the volume of data generated by the selected applications, how much detail is chosen in the ***Transactional Detail Configuration View***, SQL Server database size limitations, and available disk space.

Logging can be completely disabled so that no files are written to disk.

Notification Configuration Views

When a metric's value exceeds the *Notification Level* threshold, a production monitor can be configured to deliver alerts using the following notification methods;

- **Windows Event Log**
  - AppMetrics notifications sent to the event log contain detailed information on the object whose threshold metric was exceeded. This allows management tools such as ***Microsoft System Center Operations Manager (SCOM)*** to detect problems with COM+ applications as they occur.
  - Besides AppMetrics notifications configurable via Benchmarks and Thresholds, the Windows Application Event Log will also contain other AppMetrics status events which the AppMetrics ***SCOM*** management pack will detect and report.
- **SNMP**
  - Email messages may be sent to operations staff to alert them of problem conditions, allowing for the use of electronic pagers to draw immediate attention to hung applications and other critical issues.
- **SMTP**
  - SNMP traps can be delivered to SNMP management systems in environments using such software to monitor application health.
- **Windows Component Script**
  - AppMetrics includes scripts which may be configured to run when hung applications are detected. These scripts can either shut down or recycle the hung application instance, please see ***Appendix E - Using AppMetrics to Handle Hung Components and Applications***) for more details

    Additionally, custom scripts may be written in order to trigger other desired actions. A sample script is supplied with AppMetrics.

Note: An incorrectly configured monitor can generate a very large number of notifications. The user should use due caution when configuring benchmarks and thresholds before enabling notifications.

The ***Notification Configuration View*** is divided into three pages, ***How***, ***Who***, and ***Logging***.

- **How Page**
  - Allows the user to configure the desired notification delivery method(s)
- **Who Page**
  - Used to set the recipients of SNMP email notifications
- **Logging Page**
  - Allows for notifications to be logged in a text file in order to ensure that the configured delivery mechanisms are functioning correctly.

Note: Notifications for individual monitors may be disabled by checking the **Disable** checkbox at the top of the ***Notification Configuration View***.

Notification Configuration View - How Page

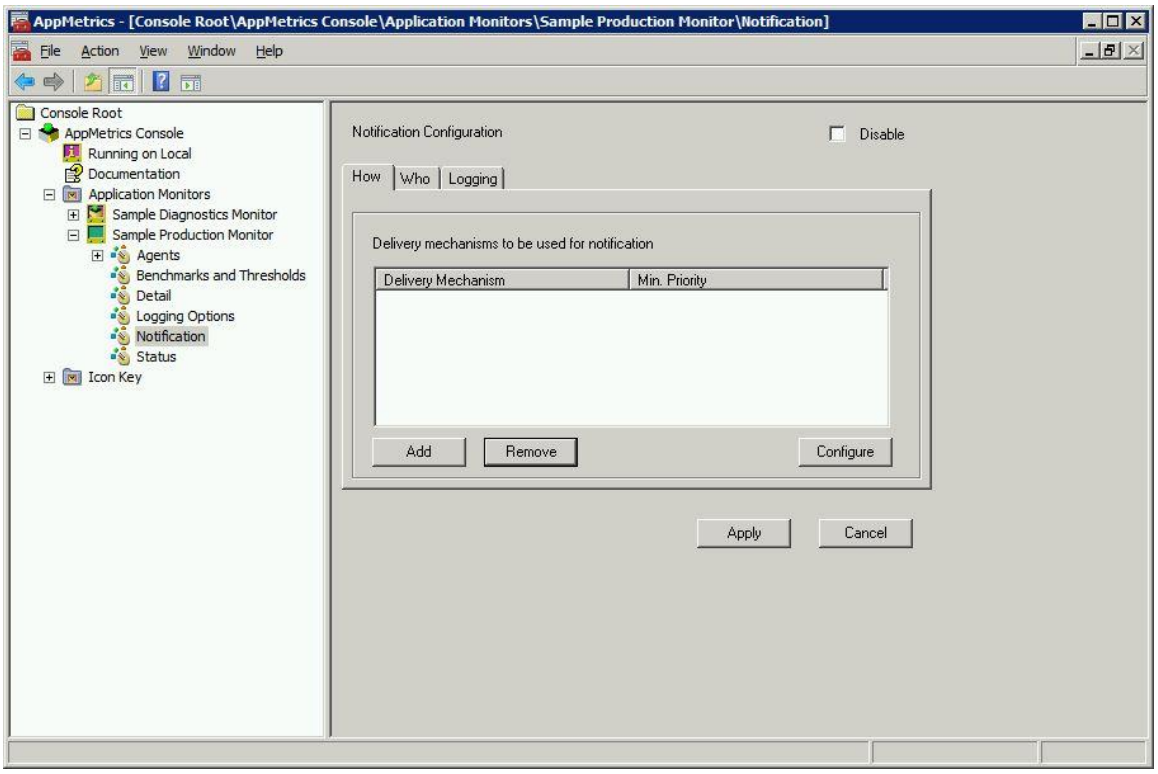


Figure 3-34 Notification Configuration View - How Page

The ***How Page*** is used to select one or more notification delivery mechanisms. The **Add** button will activate the ***Add Delivery Mechanism Dialog*** (Figure 3-35).

The **Remove** button will delete the selected delivery mechanism from the list.

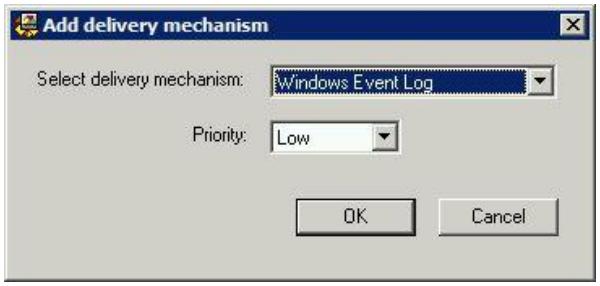


Figure 3-35 Add Delivery Mechanism Dialog

Component, *SNMP Manager*, *SMTP Mail*, and *Windows Event Log* notifications are selectable by choosing the desired mechanism from the **Select delivery mechanism** drop-down box. A priority of *High*, *Medium*, or *Low* can be specified for the messages produced by the delivery mechanism by selecting from the **Priority** drop-down box.

Windows Event Log Delivery Mechanism

Notifications to the Windows Application Event Log give detailed information on which object exceeded its threshold, and depending on the type of object, will identify the object name, the time and date the event occurred, and the name of the computer and application which experienced the event.

The event log may be monitored with management tools such as ***Microsoft System Center Operations Manager***, eliminating the need for manual parsing of the event log by operations staff.

Below is an example of an AppMetrics alert delivered to the Windows Event Log, and detected by the AppMetrics SCOM management pack. Please refer to the *AppMetrics for Transactions SCOM MP Users Guide* for further details.

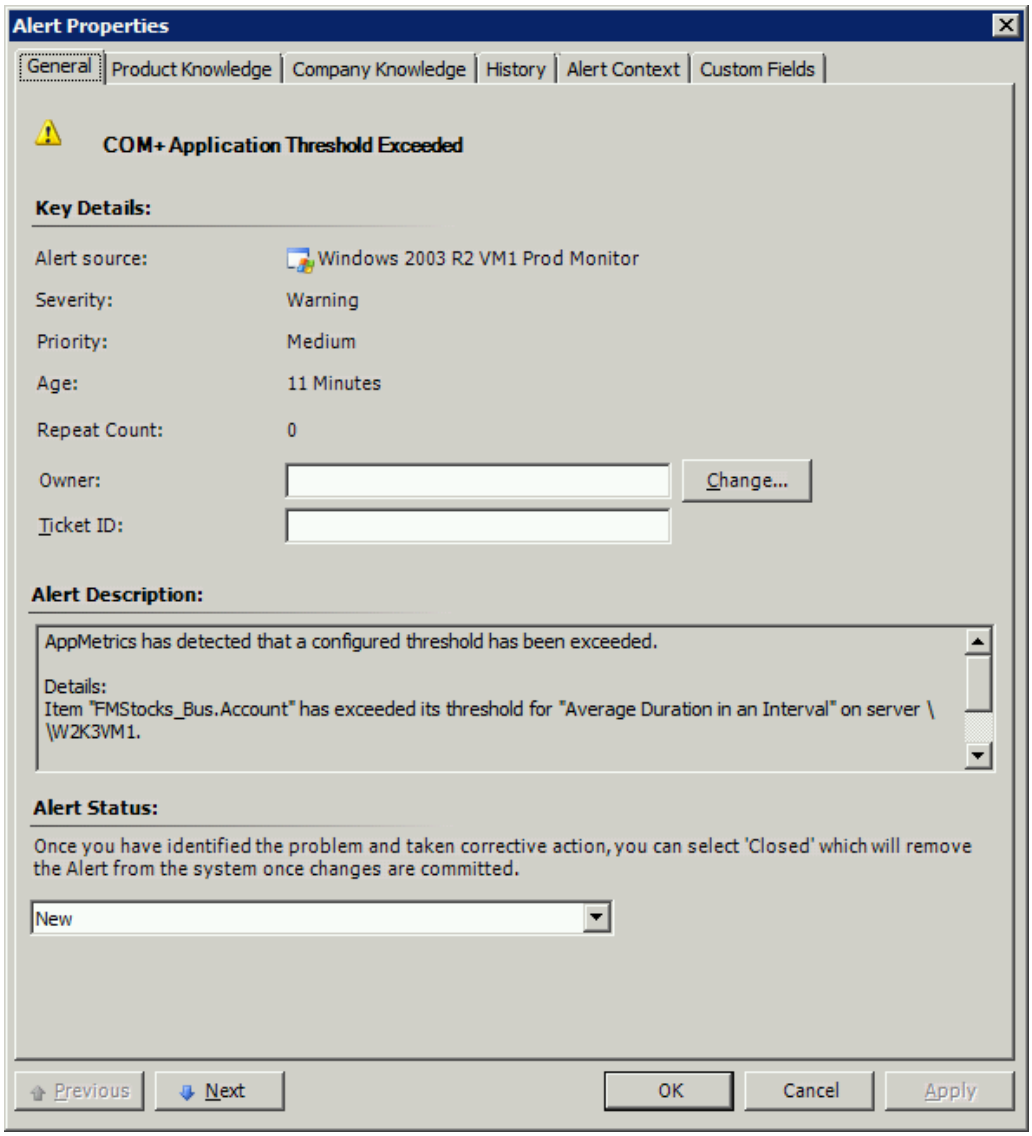


Figure 3-36 SCOM COM+ Alert

Component Delivery Mechanism

Component delivery allows for either Windows Script Components or COM component applications to be called when a threshold is exceeded.

The user can choose between Windows scripts supplied by Xtremesoft, or one of a custom design.

Depending on the version of Windows Scripting installed, you can register WSC files using regsvr32 or by right-clicking the file in Windows Explorer and choosing **Register**. Consult your Microsoft documentation for specific details and any required arguments for regsvr32.exe. Registering the WSC file requires that Windows Scripting is present on the system.

Although the supplied scripts are implemented using Windows Scripting Host, any COM component implementation should work as well, and may be preferable for those who do not enable the Windows Scripting Host feature of Windows.

AppMetrics Windows Script Components for Recycling or Shutting Down Hung Applications

There are three Windows Script Components supplied with AppMetrics, located in the \Program Files\Xtremesoft\AppMetrics for Transactions folder.

- XSHungComponent.wsc
  - This script can either recycle or shutdown COM+ application instances in which a hung component has been detected. The script requires that the **UseRecycle** variable be set within the script by use of a text editor (such as Notepad). The variable can be set to *True* to configure the script to recycle applications, or *False* to cause the hung application to be shutdown.
  - The following code resides in the XSHungComponent.wsc script, and illustrates the default setting for the **UseRecycle** variable, which is *True*.

```
' The following variable will cause either Recycle or
' Shutdown to be called for the specified application
' instance

' If set to True, Recycle will be used, if False,
' Shutdown will be used

Dim UseRecycle

UseRecycle = True
```

- XSRecycleApplication.wsc
  - Same as above, but only recycles applications. This can be used instead of XSHungApplication.wsc when shutting down an application is not a desirable option, and eliminates the need to manually edit the script.

Custom Windows Script Components

You can optionally implement a custom Windows Script Component that takes corrective action whenever a notification is sent.

A sample file named XSNotification.wsc is installed on your system with AppMetrics in the \Program Files\Xtremesoft\AppMetrics for Transactions folder (default location).

The sample Windows Scripting Host script component is offered as an illustrative example only. Modification of the component to fit a particular purpose is left to the user. Xtremesoft Professional Services can also provide customized components to your specification. Contact Xtremesoft Support for more information.

Configuring the Component Delivery Mechanism

After adding the *Component Delivery* option, you must configure the following items:

ProgID: This should correspond to a registered COM Component that includes a XSNotificationComponent_GotAlert function. A default ProgID is provided (an example of which is provided in the sample file, XSNotification.wsc).

Timeout: The value represents the number of seconds to wait before terminating the call to the XSNotificationComponent_GotAlert function in case the component does not respond.

For further information on configuring AppMetrics for the supplied hung application handler scripts, please refer to ***Appendix E - Using AppMetrics to Handle Hung Components and Applications***)

To configure a custom component of your own design, please follow the steps below.

1. Register the COM Component with a **XSNotificationComponent_GotAlert** function as specified in the sample file XSNotification.wsc.
2. In the **Delivery Mechanism** list on the ***Add Delivery Mechanism Dialog***, select *Component***.**
3. On the ***How View***, click the **Configure** button.

   The **Configure Component *Dialog*** appears.

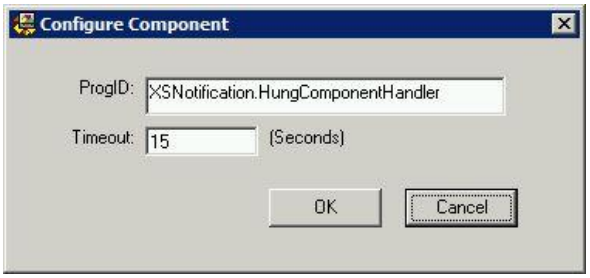


Figure 3-37 Configure Component

4. Enter a **ProgID** corresponding to the desired COM Component**.** As an example, the supplied Windows component script, XSNotification.wsc, defines the *ProdID* as XSNotification.CustomAlert.
5. Change the **Timeout** value if desired.
6. When done, click **OK**.
7. Click the **Apply** button at the bottom of the ***How View***.

SNMP Manager Delivery Mechanism

For AppMetrics to be able to deliver Notifications via SNMP, the SNMP Service must be installed prior to installing AppMetrics. If AppMetrics has been installed prior to adding the SNMP service, add the SNMP service from Windows Add/Remove components, then run AppMetrics install in **Repair** mode in order to add the necessary registry keys.

The SNMP service will need to be configured in order for AppMetrics to utilize SNMP trap notifications.

Please refer to the following Microsoft support page for details.

http://support.microsoft.com/kb/315154

Additional information can be found at the Microsoft TechNet site.

http://www.microsoft.com/technet/archive/winntas/maintain/getting.mspx?mfr=true

After adding the SNMP Manager option for notifications from the How tab, you must configure the Community setting, which is the case-sensitive SNMP community name to which this computer will send notifications. This must match the community name that was entered in the SNMP Service trap and security configuration.

Using the **Advanced** dialog, the following settings may also be specified:

- **Timeout**: The time (in milliseconds) when AppMetrics will abort an attempt to communicate with the SNMP agent.
- **Retries**: The number of times AppMetrics will attempt to communicate with the SNMP agent.

Check with your SNMP administrator to identify these settings.

To configure the SNMP Manager

1. In the **Delivery Mechanism** list, select the *SNMP Manager* item.
2. Click **Configure**. The **Configure SNMP Manager** dialog opens.

Figure 3-38 Configure SNMP Manager Dialog

3. Enter the **Community** name information for your SNMP system in the dialog, and optionally, click on the **Advanced** button to enter the **Timeout** and **Retries** settings.
4. When done, click **OK** in the ***Configure SNMP Manager*** dialog.
5. Click the **Apply** button at the bottom of the view.

Third-Party SNMP Applications

If you intend to use a third-party SNMP application to receive and monitor SNMP traps generated as AppMetrics notifications, you should compile the following MIBs into your SNMP management application. The MIB files are located in the MIBs subdirectory under the AppMetrics program files folder, which by default is *Program Files\Xtremesoft\AppMetrics for Transactions*.

- Xtremesoft.mib
- AXSAppMetrics.mib
- ProductionMonitor.mib

The procedure for compiling MIBs varies from application to application. Please consult your SNMP Management application's documentation for more information.

SMTP Mail Delivery Mechanism

After adding the SMTP Mail option for notifications, you must configure the following items:

- The originator of the e-mail (if the server has an e-mail account, the use of that account may be more convenient in situations with multiple servers)
- An SMTP host mail server
- An SMTP port (often 25)

Check with your SMTP server administrator to identify these settings.

Setting Up SMTP Virtual Server Domains for AppMetrics

If you want AppMetrics to send alerts via SMTP, first determine if any of the following conditions are true:

- The AppMetrics Manager computer has no access to an e-mail server.
- The AppMetrics Manager computer has access to an e-mail server, but this e-mail server does not allow relaying.

If either of the above conditions are true, then on the AppMetrics Manager computer a SMTP virtual server domain should be used within Internet Information Services (IIS).

A SMTP virtual server solves the first case by sending the messages to the recipients on behalf of AppMetrics. It solves the second case by using a Windows user account to authenticate to the e-mail server, bypassing the relay restrictions.

The following procedure explains how to configure a SMTP virtual server domain.

Note: This procedure requires the SMTP service subcomponent of IIS. You can install this subcomponent through IIS install or the IIS configuration console.

1. Start the Internet Services Manager.
2. In left pane, under the **Tree** tab, expand the computer name, and then expand **Default SMTP Virtual Server**.
3. Right-click **Default SMTP Virtual Server** and then select **Pause**.
4. Under **Default SMTP Virtual Server**, select and right-click **Domains**, and then in the popup menu, select **New → Domain**.
5. For the domain type, select **Remote**, and then click **Next**.
6. In the **Name** field, type the address space for the e-mail messages to be delivered by the domain (e.g., ***organizationname*.com**), and then click **Finish**.
7. In the right pane of the **Internet Information Services** snap-in window, right-click your newly created domain, and select **Properties**.
8. Select **Allow incoming mail to be relayed to this domain**.
9. If you will route mail to an existing e-mail server in your organization, then in the **Route domain** area, select **Forward all mail to smart host**. In the box below it, type the name or IP address of the computer for the e-mail system.
10. If you are routing to an Exchange Server host, you should set **Outbound Security** for the domain. To perform this, you can use the following sub-steps:
    a. Click the **Outbound Security** button.
    b. Select ***Windows security package***.

c. In the **Account** box, type the name of the domain and user name for the account using the following format: ***domain\username***.

   - Or -

   Click **Modify**, locate and select the preferred account, and then click **OK**.

d. Type the **Password** for the account.

e. Click **OK**.

11. Click **OK**.

12. In the left pane of the **Internet Information Services** snap-in window, right-click **Default SMTP Virtual Server**, and then select **Start**.

Configuring SMTP Mail Delivery

1. In the **Delivery Mechanism** list on the ***Add Delivery Mechanism Dialog***, select *SMTP Mail*.

2. Click the **Configure** button on the ***How View***.

   The ***Configure SMTP Mail Dialog*** appears.

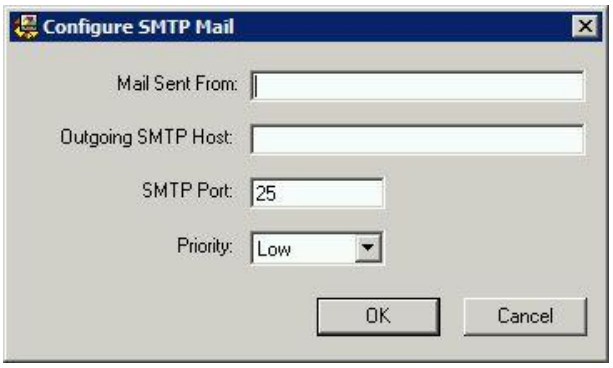


Figure 3-39 Configure SMTP Mail Dialog

3. Complete the information pertinent to your SMTP server.

   Note: If you configured an SMTP virtual server domain in IIS using the procedure in *Setting Up SMTP Virtual Server Domains* on Page 3-51, then in the **Outgoing SMTP Host** field, type the name of the local computer.

   Otherwise, if you are not using an SMTP virtual server domain, type the name of your organization's e-mail server.

4. When done, click **OK**.

5. Click the **Apply** button at the bottom of the view.

Notification Configuration View - Who Page

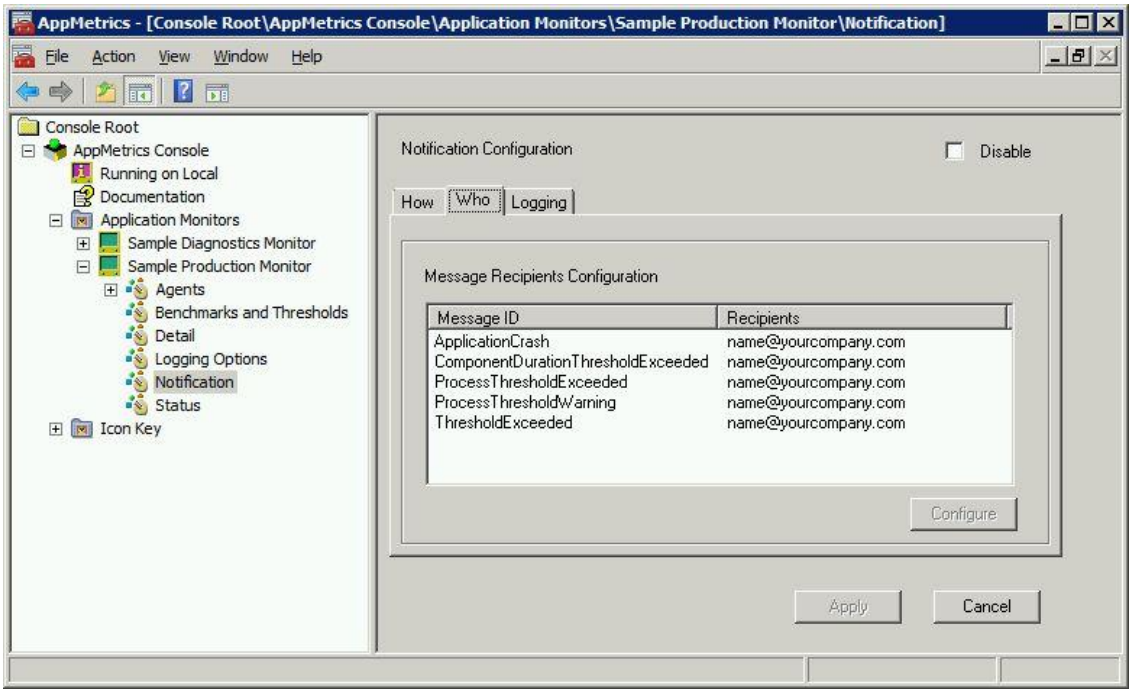


Figure 3-40 Notification Configuration View - Who Page

The ***Who*** page is used to specify one or more SMTP email notification recipients by selecting the message ID which correlate to the type of alert, and then clicking the **Configure** button. This opens the ***Configure Recipients*** dialog.

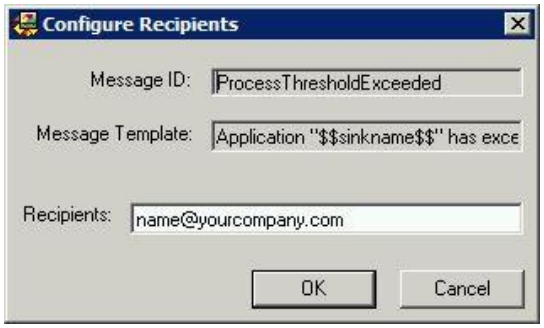


Figure 3-41 Configuration Recipients Dialog

In the **Recipients** field, each address should be in the "Internet email" format, such as username@domain.com. A comma should separate each address when entering multiple recipients.

Notification Configuration View - Logging Page

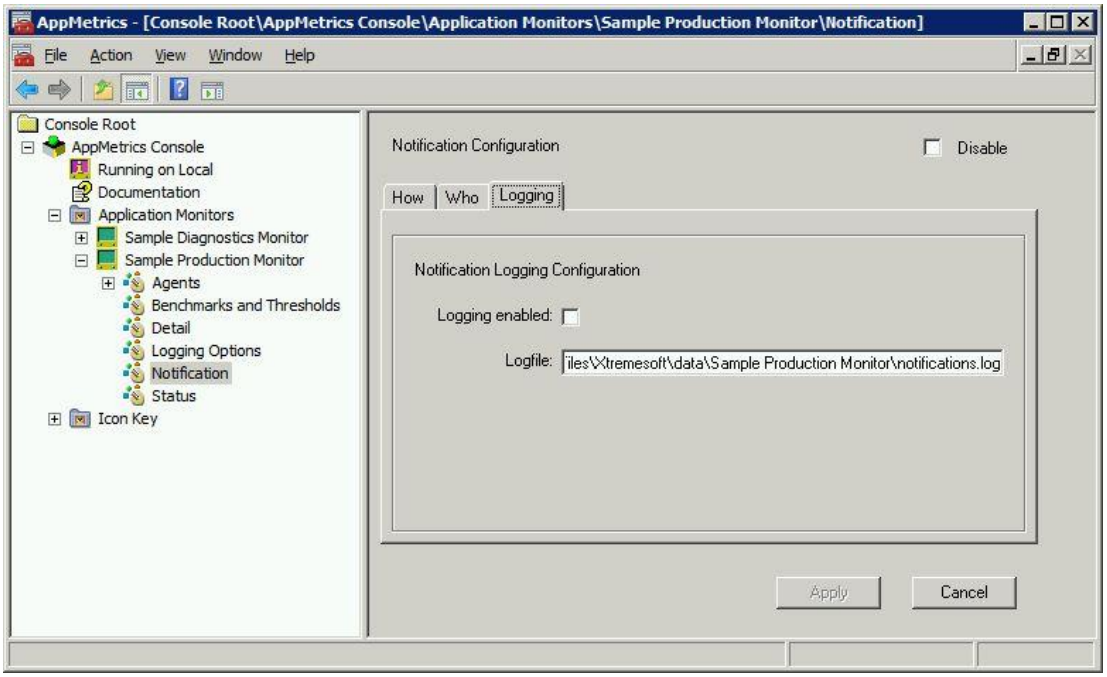


Figure 3-42 Notification Configuration View - Logging Page

The ***Logging*** page of the ***Notifications Configuration View*** is used to enable notification logging in order to determine if notifications are being delivered correctly when thresholds are exceeded or when an application crashes. Checking the **Logging enabled** checkbox enables logging to the file specified in the **Logfile** field.

This feature should be used when notifications are first configured to ensure that they are being delivered as expected. It can be disabled once proper operation is verified.

Status View

Both production and diagnostic monitors display current and historical status information on the ***Status View***. The ***Current Status*** page will differ however depending on the type of monitor.

Current Status Page

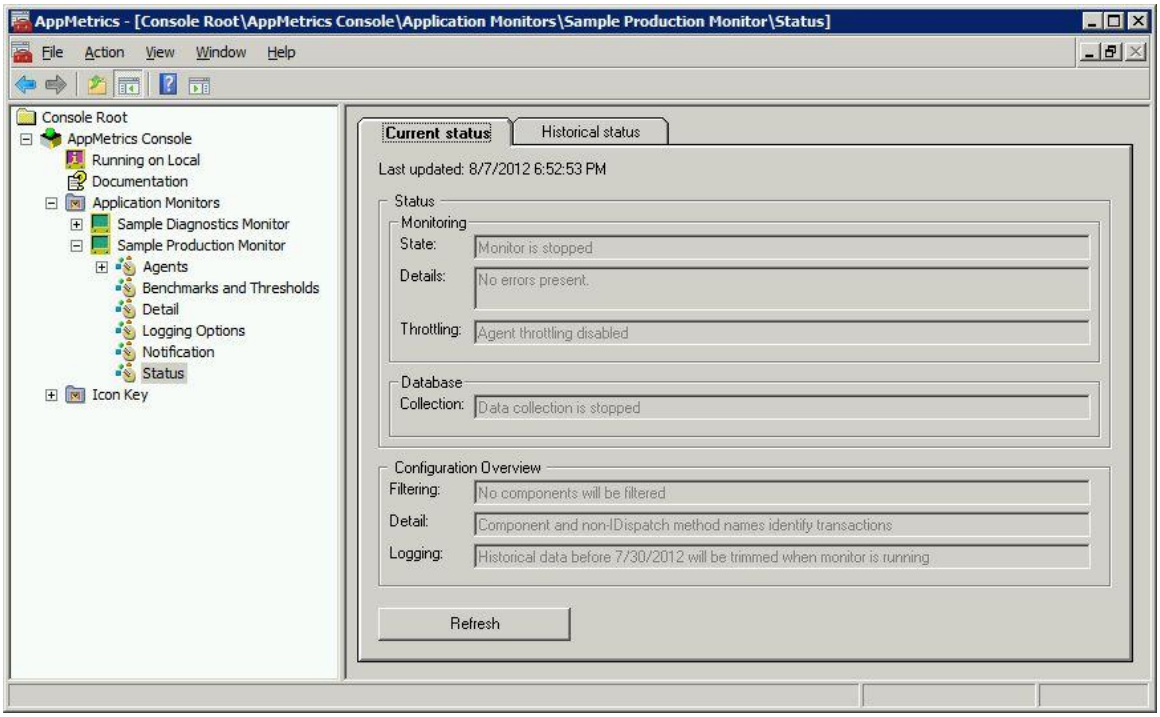


Figure 3-43 Current Status Page

The **Current Status** page displays the current configuration and operational state of the monitor:

- The time the status was last refreshed.
- A description of the current operational state of the monitor.
- A description of any error state present.
- An indication of whether or not throttling is enabled for the associated agent, or a description of the conditions under which agent throttling will occur. For more information, see **Detail Throttling for COM+ Production Monitors** on page 3-41.
- A description of the current state of data collection.
- The *Component Filtering* setting selected on the monitor's **Transactional Detail Level Configuration View**.
- The *Transactional Detail Level* setting selected on the monitor's **Transactional Detail Level Configuration View**.
- An overview statement for the current configuration of the data logging settings from the ***Logging Options View***.

The **Refresh** button is used to query the current state of the monitor.

Historical Status Page

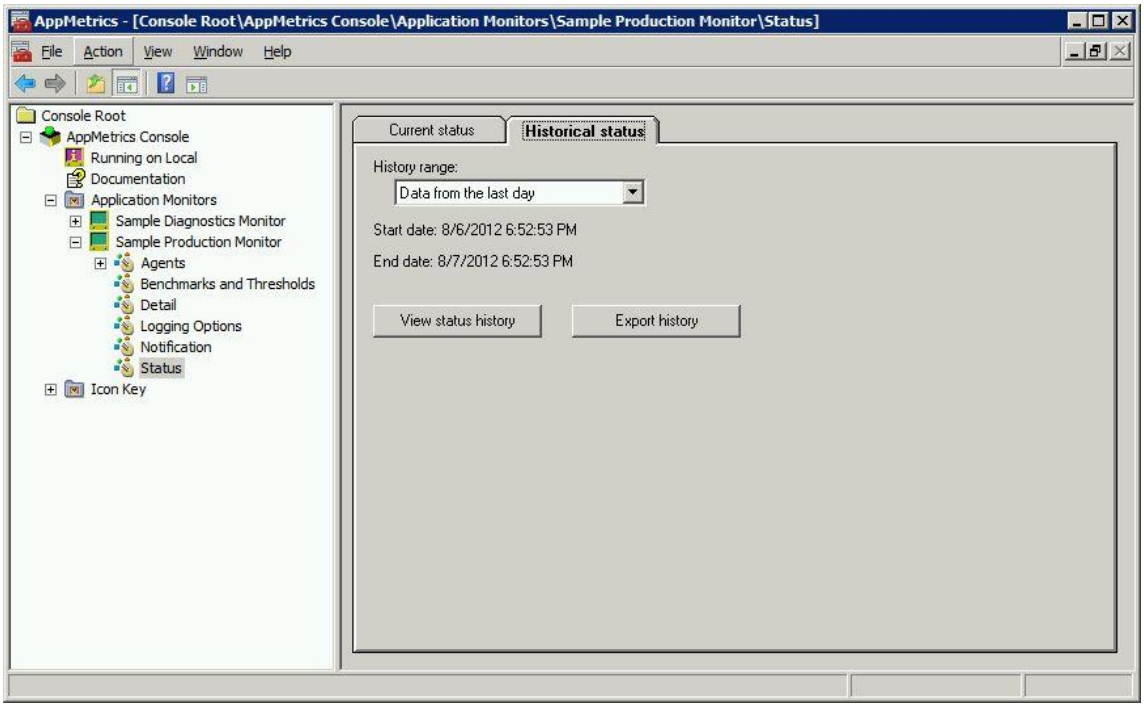


Figure 3-44 Historical Status View

The ***Historical Status*** page lets you view or export a historical summary of the system's operational state. You start by selecting one of the predefined time periods from the **History range** drop-down list. Note that the specified time period starts with the current time and ends with the specified interval. As an example, selecting *Data from the last day* returns data from the last 24 hours to the current time.

To save the historical status data to an XML file, click the **Export history** button and then specify a file name and location.

To view the historical status, click the **View status history** button. The ***Monitor Status Window*** contains information about the monitor and any associated agent, displayed in a tabular form as a time-ordered list with the most recent events listed first.

Both the exported XML file and the ***Monitor Status Window*** display the following information about each event:

- **Event time:** The timestamp for the event.
- **Event name:** Provides a categorization of the events and contains items such as startup and shutdown messages.
- **Event reported by:** The entity reporting the event. This is either the current monitor or a sub-system within the monitor.
- **Status:** A description of the cause of the event. This may be blank.
- **Status code:** A code that corresponds to the description in the *Status* field. If *Status* contains a description, then the status code field always contains a corresponding code.

Production Monitor Shutdown Option

If a monitored application has a large number of distinct components and transactions, a significant delay may be experienced in shutting down the associated production monitor.

In such a case, the shutdown delay can be eliminated by unchecking the **Save Application Metrics** checkbox on the production monitor ***Properties*** dialog.

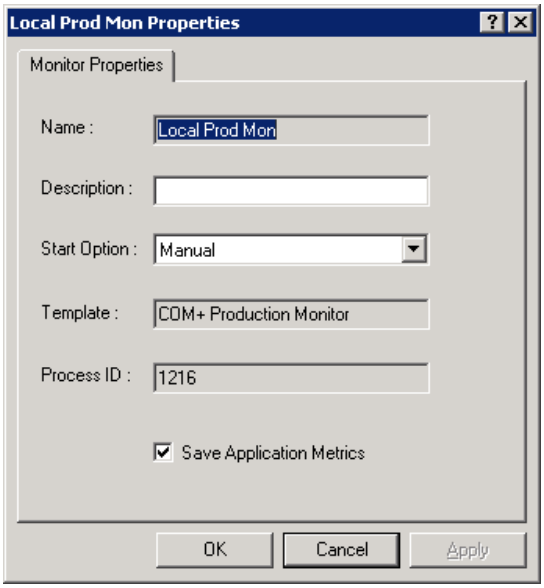


Figure 3-45 Production Monitor Properties Dialog

This has the effect of no longer saving freshly detected components and transactions, thus should only be used *after* running the production monitor at least once under expected loads for a long enough period of time to capture all desired components and transactions, then shutting down the monitor in order to save the initial snapshot of those components and transactions.

The existing components and transactions will still be stored for use by AppMetrics, and this option will not affect the functioning of the *Benchmarks and Thresholds* screens or other related functions.

The *Save Application Metrics* checkbox can be reset as needed in order to delete obsolete information from the stored components and transactions, or to add new or updated component and transaction names to the stored list.

To modify the **Save Application Metrics** setting;

1. Right click the desired production monitor node.
2. Select **Properties** from the popup menu.
3. Modify the **Save Application Metrics** checkbox as desired, and then click **OK**.

COM+ Diagnostics Monitors

The purpose of a diagnostic monitor is to capture as much information about the application as possible in order to diagnose problem areas. The tradeoff is that diagnostic monitoring introduces a higher load on the application server, thus should only be used when necessary to obtain detailed method call and component information which can later be viewed in the *AppMetrics Diagnostic Drilldown Reports*.

Diagnostic monitors collect a larger set of instrumentation events produced by COM+ with regards to the application(s) under test than production monitors. The resulting log files can grow to large sizes, so log file management is more important in diagnostic monitoring and should be utilized as necessary in order to maintain sufficient disk space and control the size of the SQL Server database files.

The descriptions of the data elements that are written to the log files and uploaded to the SQL Server database are described in Chapter 4 - *Metrics* chapter. The formats of the actual log files are described in the *Appendix*.

A new subdirectory is created in the AppMetrics data folder each time a monitor is created. The new subdirectory shares the name the user gives the monitor. By default all logging data generated by that monitor is placed into subdirectories beneath the monitor folder.

Diagnostics Monitor Runtime Views

Applications Runtime View

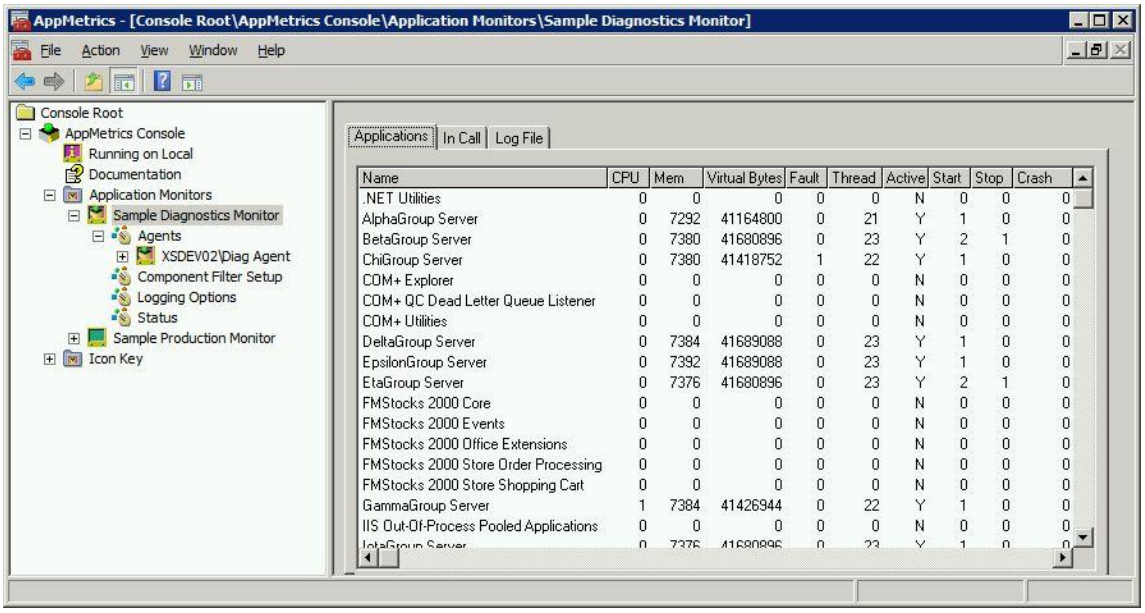


Figure 3-46 Applications Runtime View

The ***Applications*** runtime view is identical between both the diagnostic monitor and the production monitor.

When a diagnostic monitor node is selected, the **Applications** runtime view displays a list of COM+ applications currently being monitored along with process metrics and operational status for each. The metrics are updated every 5 seconds by default and can be sorted by clicking a column heading in the list.

In Call Runtime View

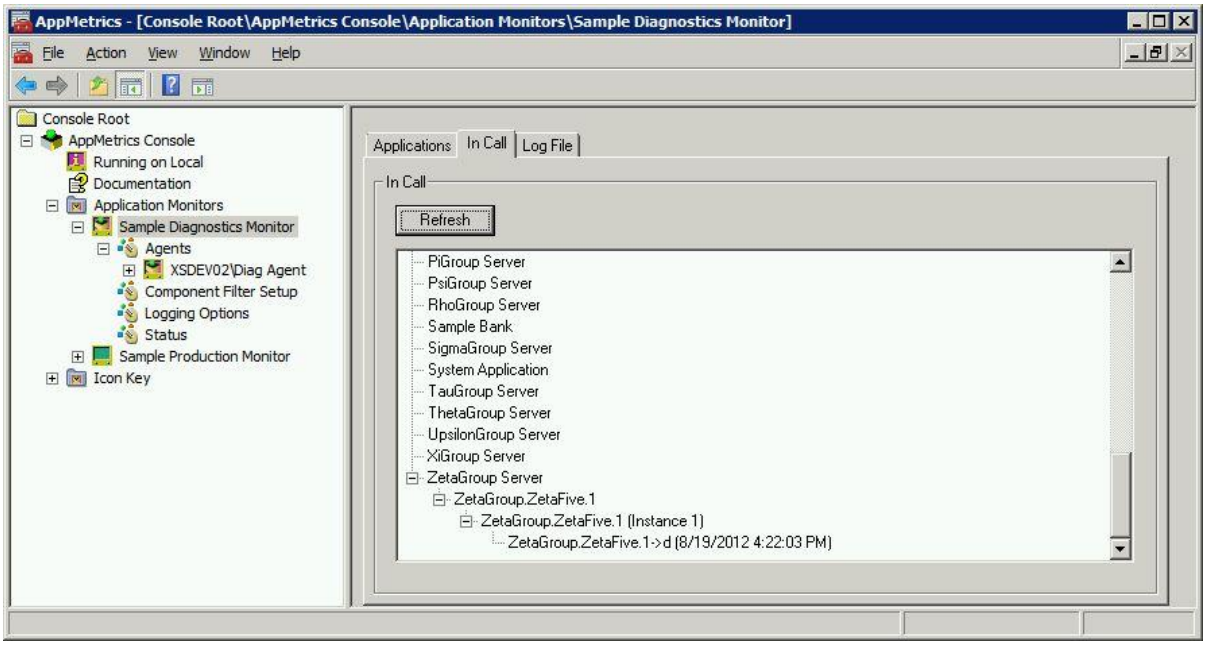


Figure 3-47 In Call Runtime View

The **In Call Runtime View** will show all active applications, components and methods in a tree view. This display is useful when looking for stuck applications, components, or transactions and the method start time is displayed for this purpose. The display is updated when the **Refresh** button is clicked.

This view differs from the production monitor ***In Call Runtime View*** in that it only displays the *Applications* list, not the *Transactions* list.

Log File Runtime View

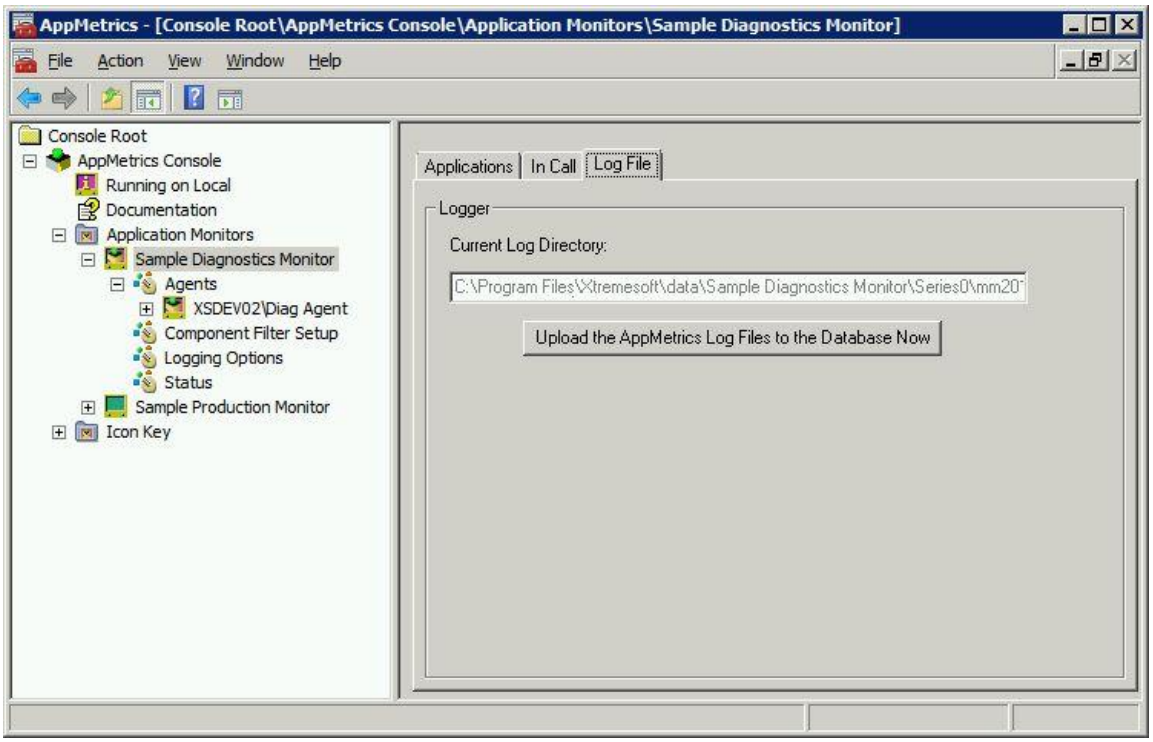


Figure 3-48 Log File Runtime View

The logging feature in diagnostics monitors works much the same way as it does in production monitors. The log files have different formats and contain different metrics, but the operation is the same in both cases.

The ***Upload the AppMetrics Log File to the Database Now*** button is for situations where the user wishes to diagnose an event that has just occurred in a monitored application. The button immediately uploads the data to the database so that reports can be run to examine the new data.

The log file directory pathname is determined from the following data:

- The data directory specified during the AppMetrics install
- The monitor name
- A series number incremented when the **Upload the AppMetrics Log Files to the Database Now** button is used
- A subdirectory name derived from the date and time

COM+ Applications View – Diagnostics Monitor

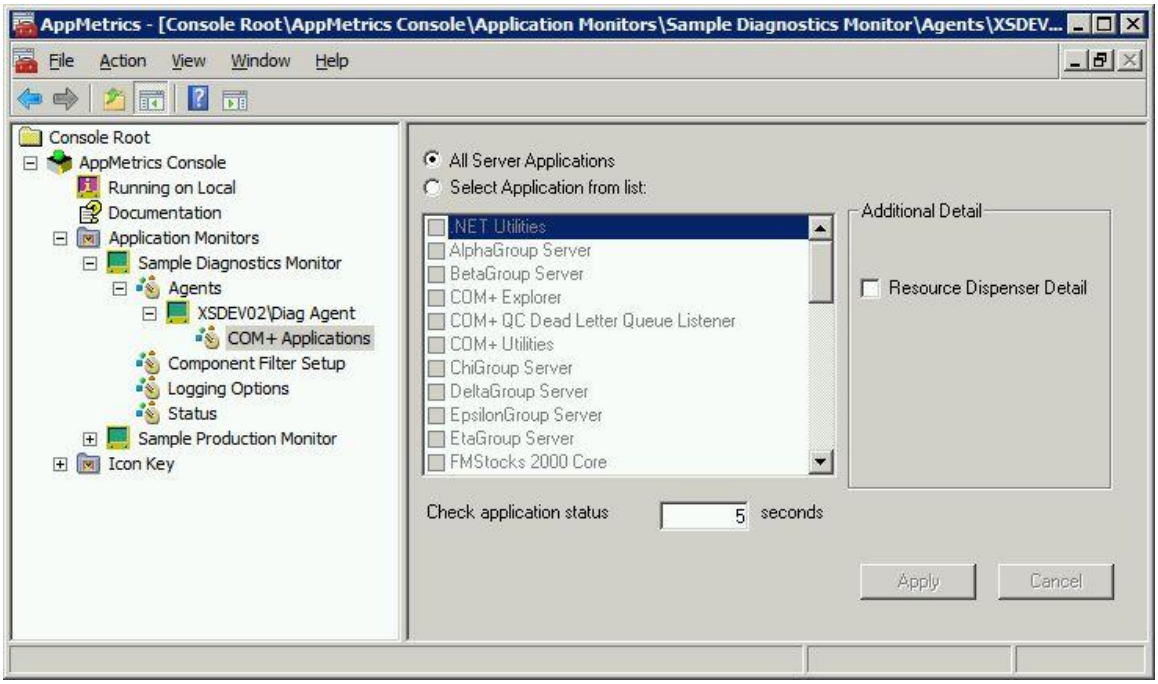


Figure 3-49 COM+ Applications View

The ***COM+ Applications View*** is used to configure the agent associated to the diagnostics monitor. It is accessible by expanding the navigation tree under the monitor to the agent node, and then selecting *COM+ Applications*.

A diagnostics agent's ***COM+ Applications View*** differs from that of a production agent in that it contains an *Additional Detail* section. By default, **Resource Dispenser Detail** events are not sent to the log files because these events are not used in AppMetrics reports, although if it is desired to collect such data for further analysis, it is possible to do so by checking the checkbox.

Note: The monitor must be stopped prior to making changes on this screen.

Component Filter Setup View

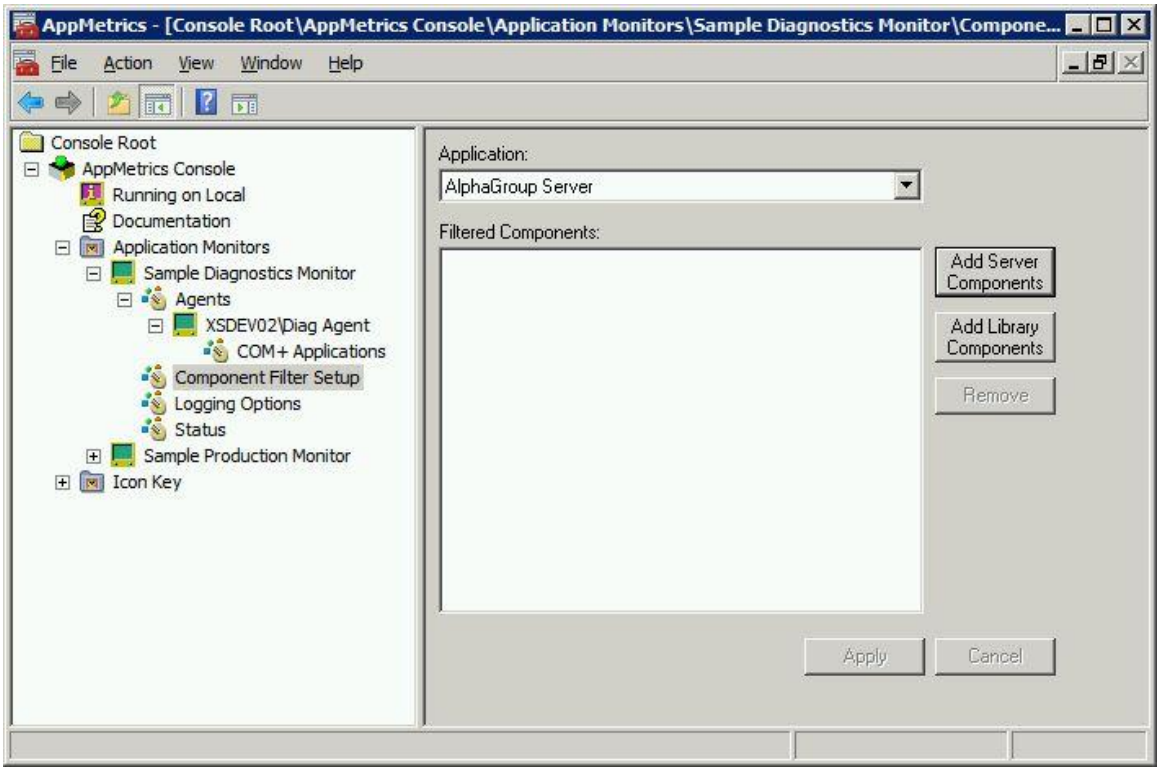


Figure 3-50 Component Filter Setup View

As diagnostics monitors collect and log a large number of COM+ instrumentation events for each monitored application, diagnostic monitoring may place a measurable load on the monitored server. Since detailed information might only be required for a single component or set of components, unneeded information related to other components can be filtered using the ***Component Filter Setup View*.**

If the user wishes to filter specific components, the ***Component Filter Setup View*** can be used to list all of the detected components of the selected type within the application specified by the *Applications* drop-down box. The user may then select which components to filter. Those components will then be ignored by AppMetrics monitoring.

An example is provided below of configuring a diagnostics monitor to filter a specific COM+ server component.

1. Select the desired application from the **Application** combo box.
2. Click the **Add Server Components** button.
3. The ***Add Server Components Dialog*** (Figure 3-51) will appear.

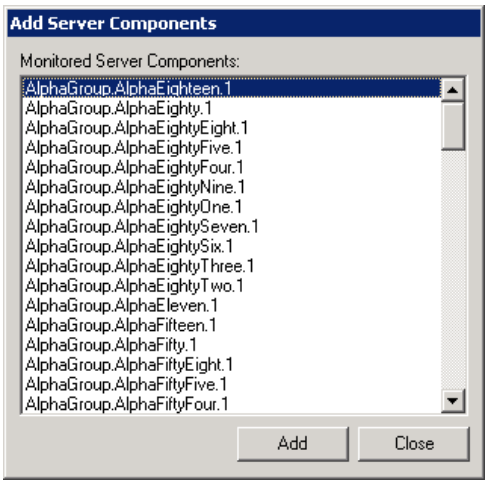


Figure 3-51 Add Server Components Dialog

4. Select the desired component(s) from the list box.
5. Click the **Add** button.
6. Additional selections may be made, and the **Add** button clicked for each individually or for multiple selections.
7. When done, click the **Close** button.
8. The selected components will then appear in the **Filtered Components** list.
9. Select the **Apply** button to save the changes.

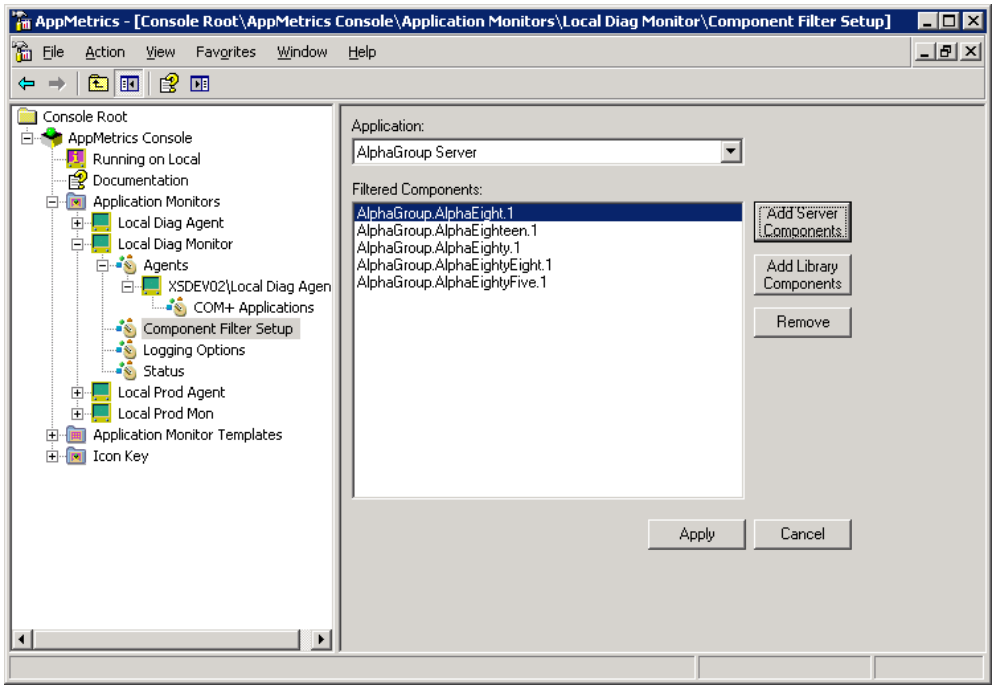


Figure 3-52 Filtered Components

Logging Options View

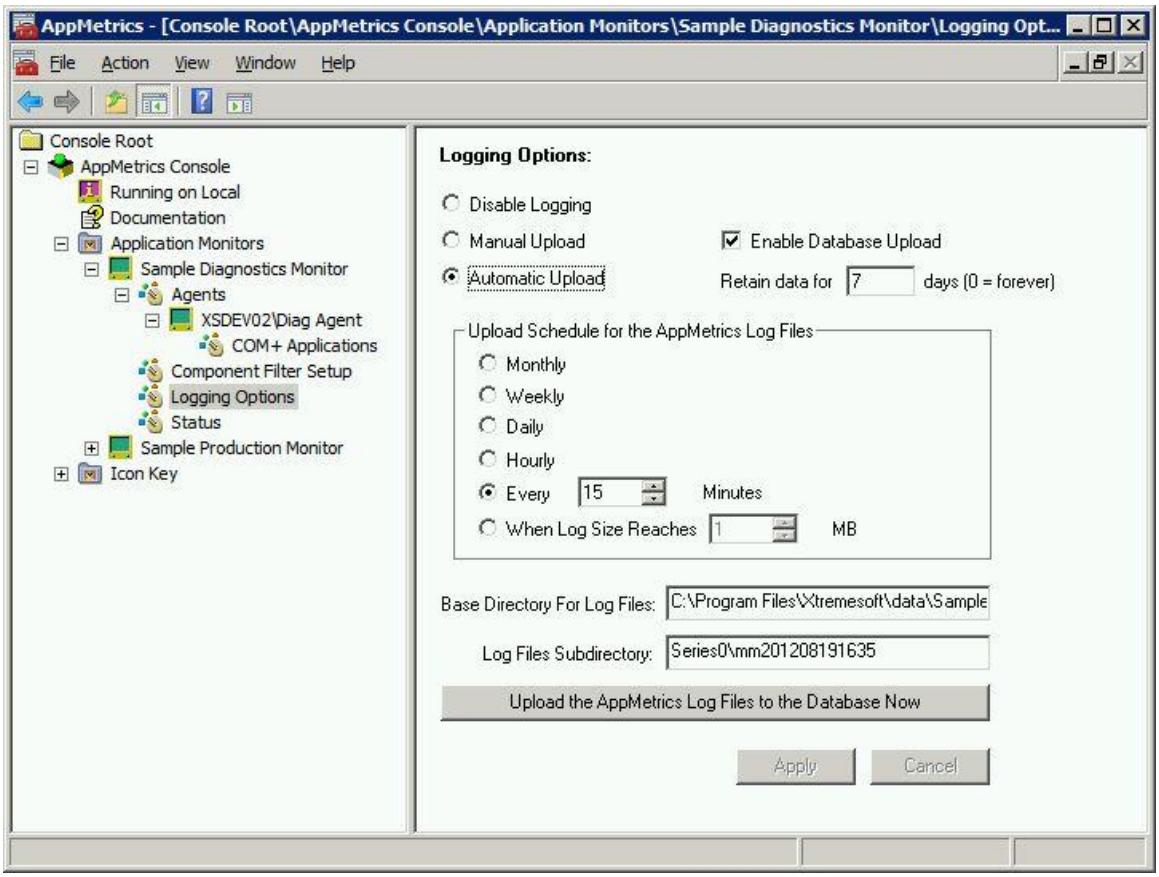


Figure 3-53 Diagnostics Logging Options View

The diagnostics monitor ***Logging Options View*** is identical to that of the production monitor.

AppMetrics incorporates automatic uploads of collected data to a SQL Server database. The upload can be scheduled to occur periodically, or be triggered when any individual log file exceeds a specified size.

Metrics are directed into separate files for easy upload into databases. If SQL Server is available on the system, then the **Enable Database Upload** check box will be available. The ***Retain data for x days*** field controls how long the data will be kept in the database.

The monitor's base log file directory can be changed, if desired, by changing the path in the **Base Directory for Log Files** field.

The **Upload the AppMetrics Log Files to the Database Now** button will upload the log files to SQL Server, and create a new *Series* folder for the next set of log files. It is identical to the button of the same name mentioned in the ***Log File Runtime View***.

The ideal upload schedule depends upon the volume of data generated by the selected applications, component filtering, SQL Server database size limitations, and available disk space.

Logging can be completely disabled so that no files are written to disk.

Status View

Both production and diagnostic monitors display current and historical status information on the ***Status View***. The ***Current Status*** page will differ however depending on the type of monitor.

Current Status Page

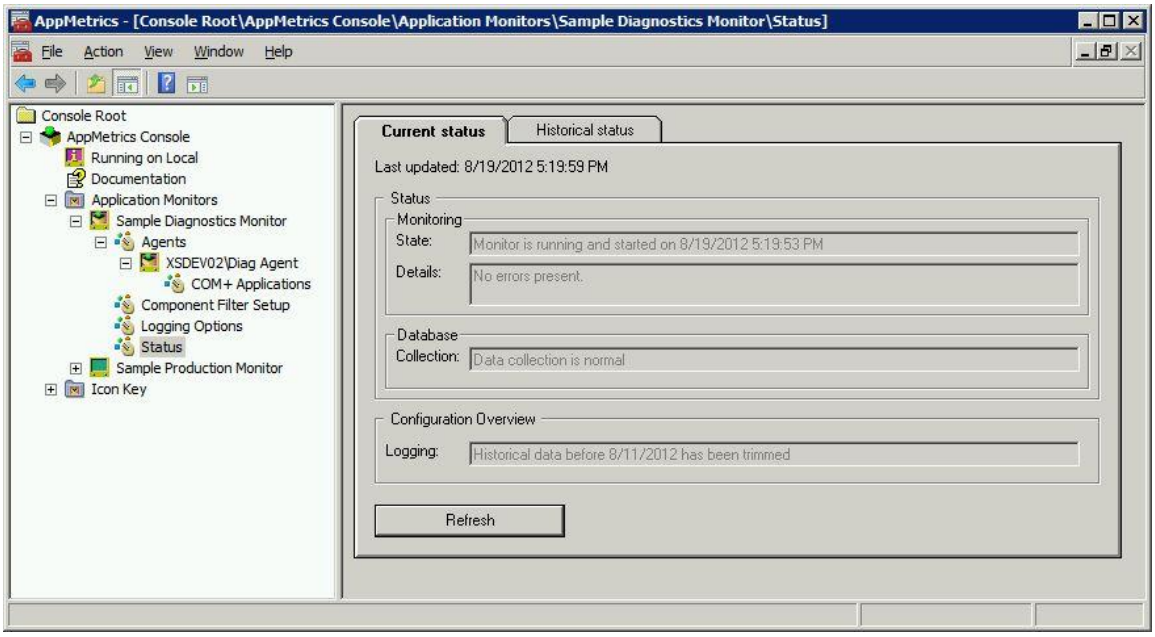


Figure 3-54 Current Status Page

The **Current Status** page displays the current configuration and operational state of the monitor:

- The time the status was last refreshed.
- A description of the current operational state of the monitor.
- A description of an error state present.
- A description of the current state of data collection.
- An overview statement for the current configuration of the data logging settings from the ***Logging Options*** view.

The **Refresh** button is used to query the current state of the monitor.

Historical Status Page

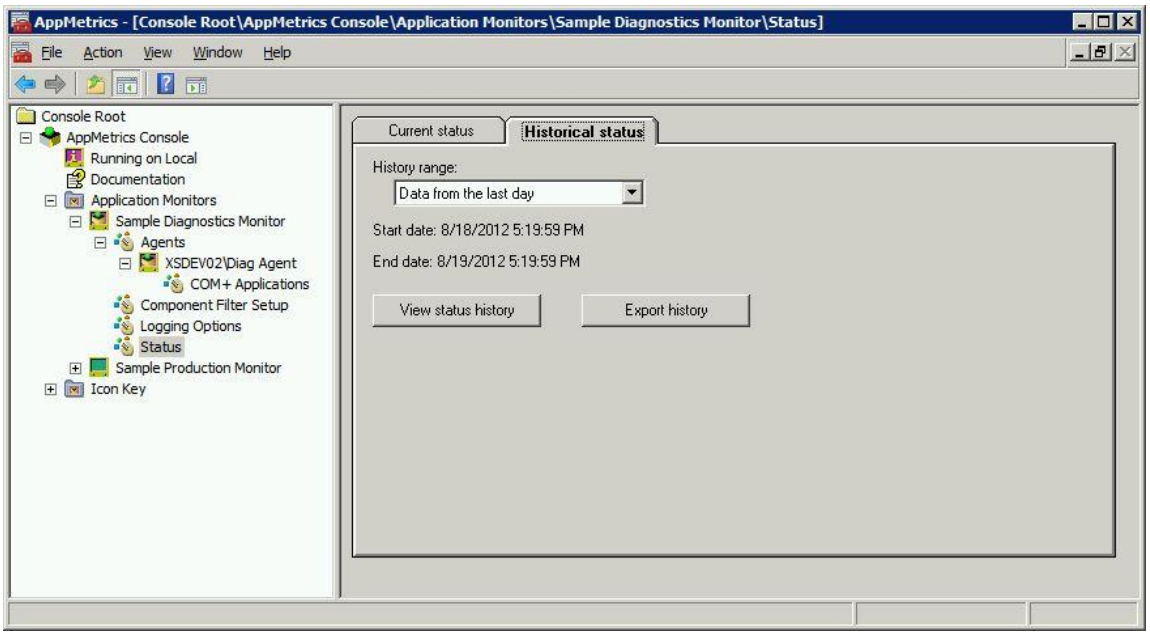


Figure 3-55 Historical Status View

The ***Historical Status*** page lets you view or export a historical summary of the system's operational state. You start by selecting one of the predefined time periods from the **History range** drop-down list. Note that the specified time period starts with the current time and ends with the specified interval. As an example, selecting *Data from the last day* returns data from the last 24 hours to the current time.

To save the historical status data to an XML file, click the **Export history** button and then specify a file name and location.

To view the historical status, click the **View status history** button. The ***Monitor Status Window*** contains information about the monitor and any associated agent, displayed in a tabular form as a time-ordered list with the most recent events listed first.

Both the exported XML file and the ***Monitor Status Window*** display the following information about each event:

- **Event time:** The timestamp for the event.
- **Event name:** Provides a categorization of the events and contains items such as startup and shutdown messages.
- **Event reported by:** The entity reporting the event. This is either the current monitor or a sub-system within the monitor.
- **Status:** A description of the cause of the event. This may be blank.
- **Status code:** A code that corresponds to the description in the *Status* field. If *Status* contains a description, then the status code field always contains a corresponding code.

Multiple AppMetrics Console Instances

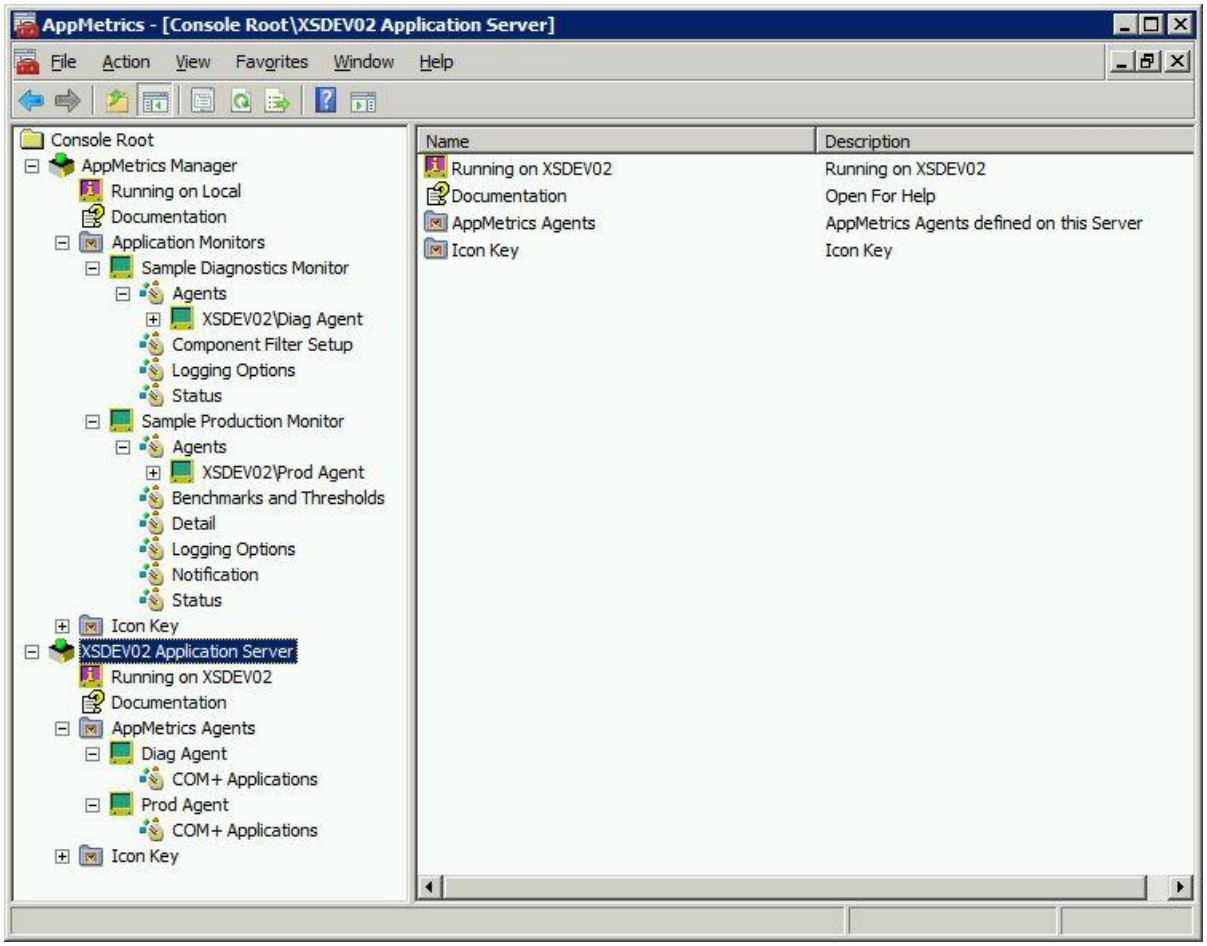


Figure 3-56 Multiple AppMetrics Console Instances

The *AppMetrics Console* is implemented as a Microsoft Management Console (MMC) snap-in. Each instance of *AppMetrics Console* can manage application monitors and/or agents defined on a specific manager or agent computer.

To manage multiple AppMetrics manager or agent computers from a single MMC instance, you can create an *AppMetrics Console* instance for each computer to be managed from within the single MMC instance.

Notes:

- The following procedures require that the MMC be run in *Author* mode.
- Any changes made to the MMC in authoring mode must be saved when closing AppMetrics; otherwise any changes made will be lost. The user will be prompted to save the console settings upon exit. These changes only relate to the snap-in modifications and do not affect any changes made to the AppMetrics monitor or agent settings for the connected servers.

Entering Authoring Mode for the MMC

1. Using Windows Explorer, navigate to the AppMetrics Program Files folder. The default location is **\Program Files\Xtremesoft\AppMetrics for Transactions**.

2. Right click the **AppMetrics.msc** file and select the *Author* menu item.

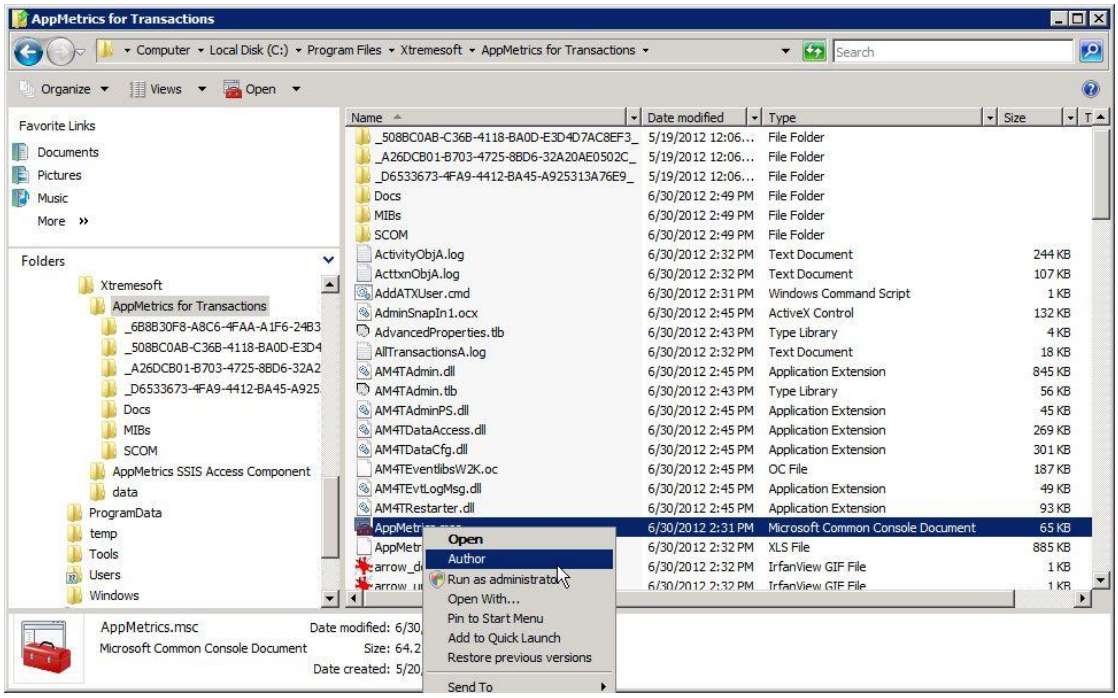


Figure 3-57 Entering Authoring Mode for the MMC

Adding an AppMetrics Console Instance

1. From the AppMetrics Console *File* menu, select *Add/Remove Snap-in*.

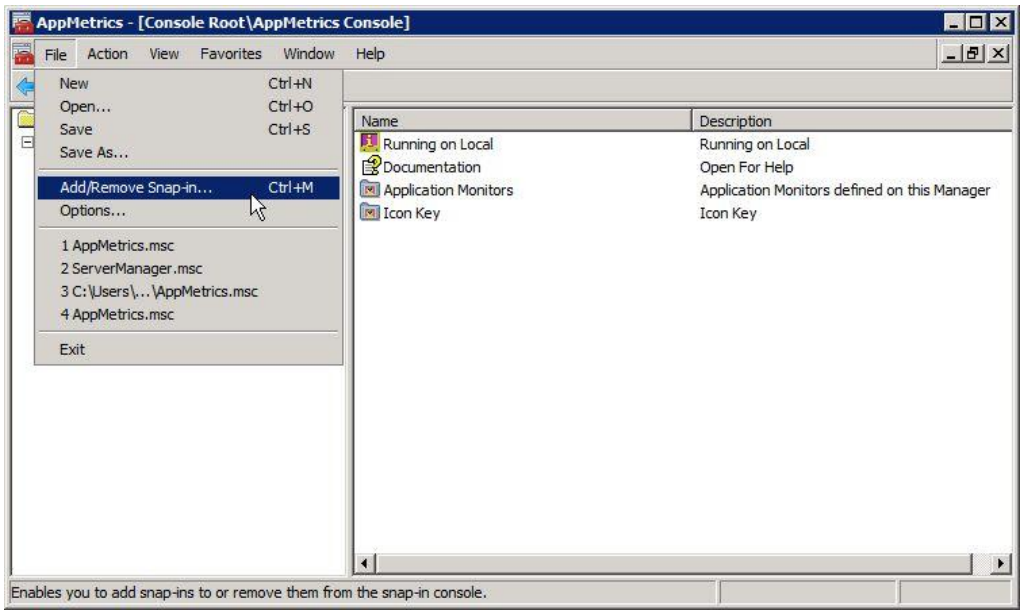


Figure 3-58 AppMetrics Console - Add/Remove Snap-in

2. This opens the ***Add/Remove Snap-ins*** dialog.

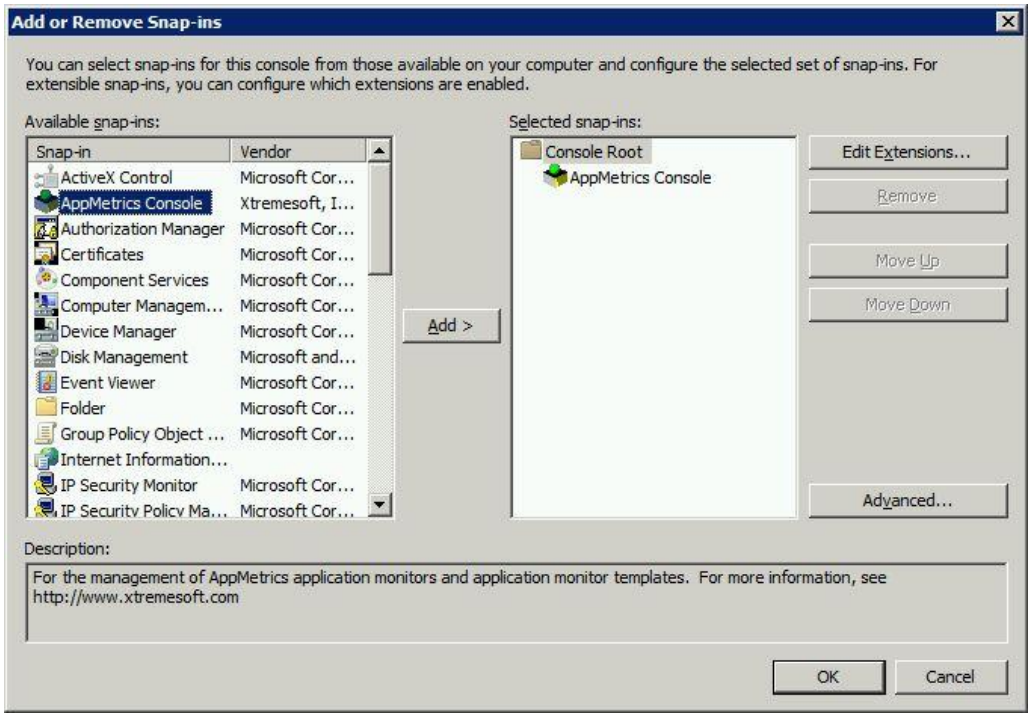
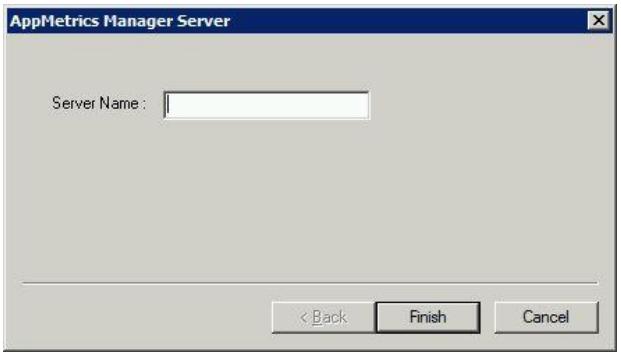


Figure 3-59 Add/Remove Snap-ins Dialog

3. Select the *AppMetrics Console* item from the *Available snap-ins* list.
4. Click the **Add** button to place an *AppMetrics Console* snap-in instance to the *Selected snap-ins* list.
5. The *AppMetrics Manager Server* dialog will open.



6. In the **Server Name** field, enter the name of the AppMetrics computer to attach to.

   Note:

   Although the dialog is named AppMetrics Manager Server, AppMetrics Agent computers may also be accessed in this manner.

7. Click **Finish**.
8. Additional *AppMetrics Console* instances may be added by clicking the **Add** button on the ***Add/Remove Snap-Ins*** dialog and entering the additional server names as required.
9. Once done adding console instances, click **OK**.

   The new *AppMetrics Console* instances are added to the *Console Root* node.

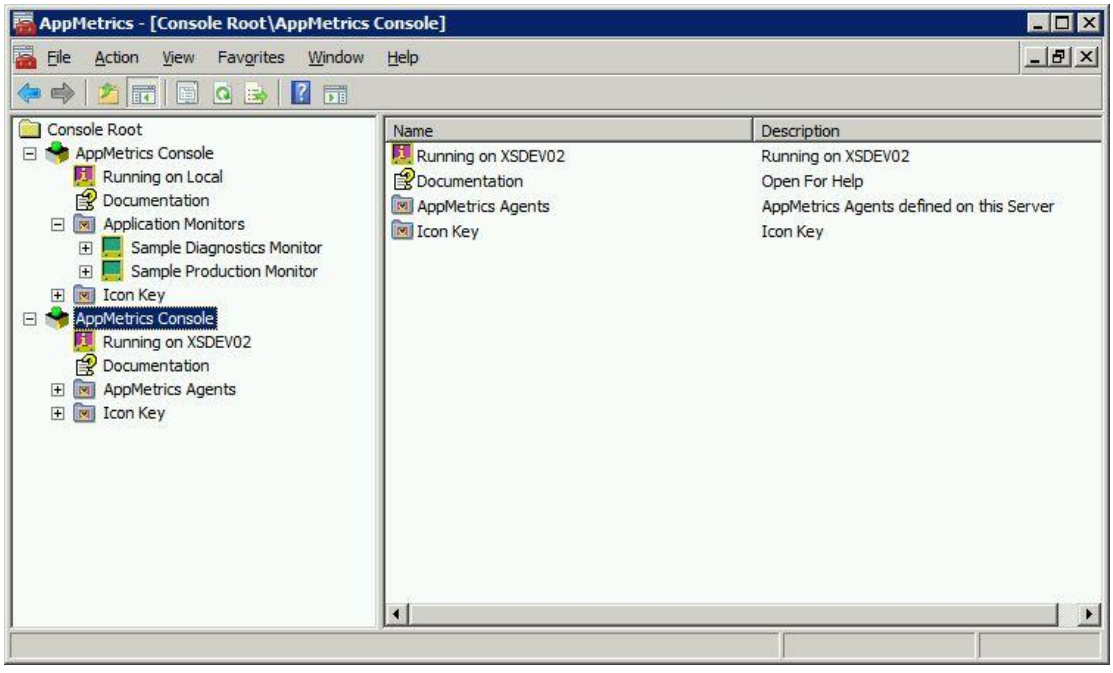


Figure 3-60 New AppMetrics Console Instance

Renaming an AppMetrics Console Instance

By default, an AppMetrics Console instance added to the MMC is given the name "AppMetrics Console". In order to organize the MMC logically, it is recommended that the new instance (and original instance if desired) be renamed to indicate the name of the connected server.

To rename an AppMetrics Console instance:

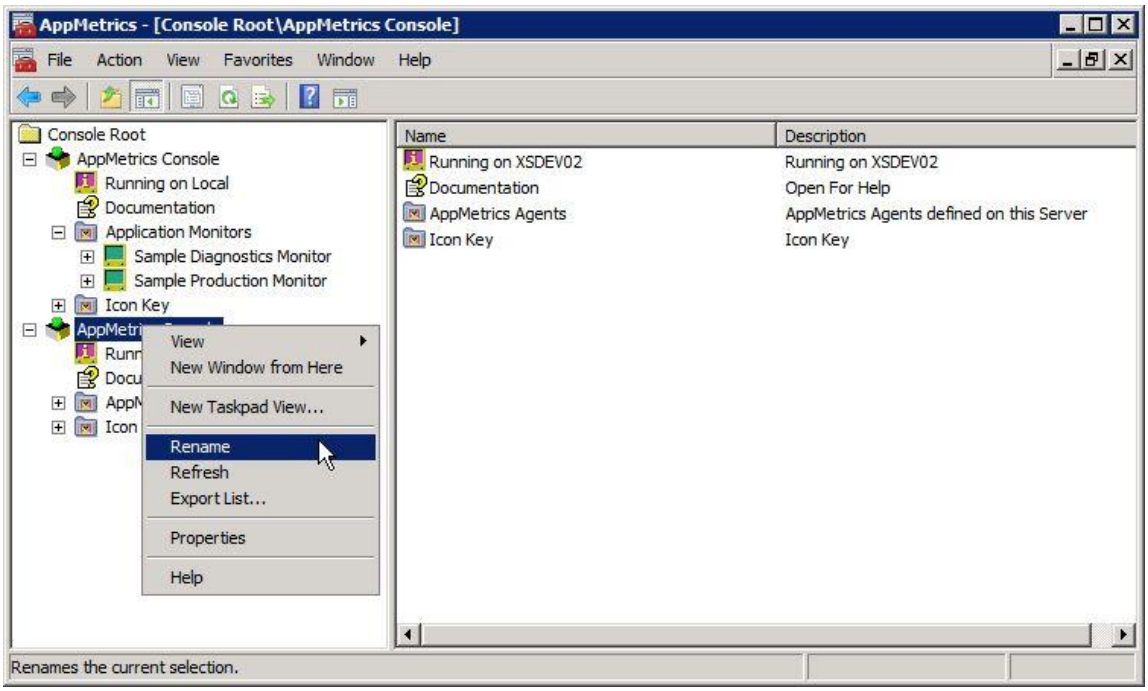


Figure 3-61 Renaming an AppMetrics Console Instance

1. Right-click the instance and then select **Rename**.
2. Enter the new name for the instance.

Changing the order of the AppMetrics Console Instances

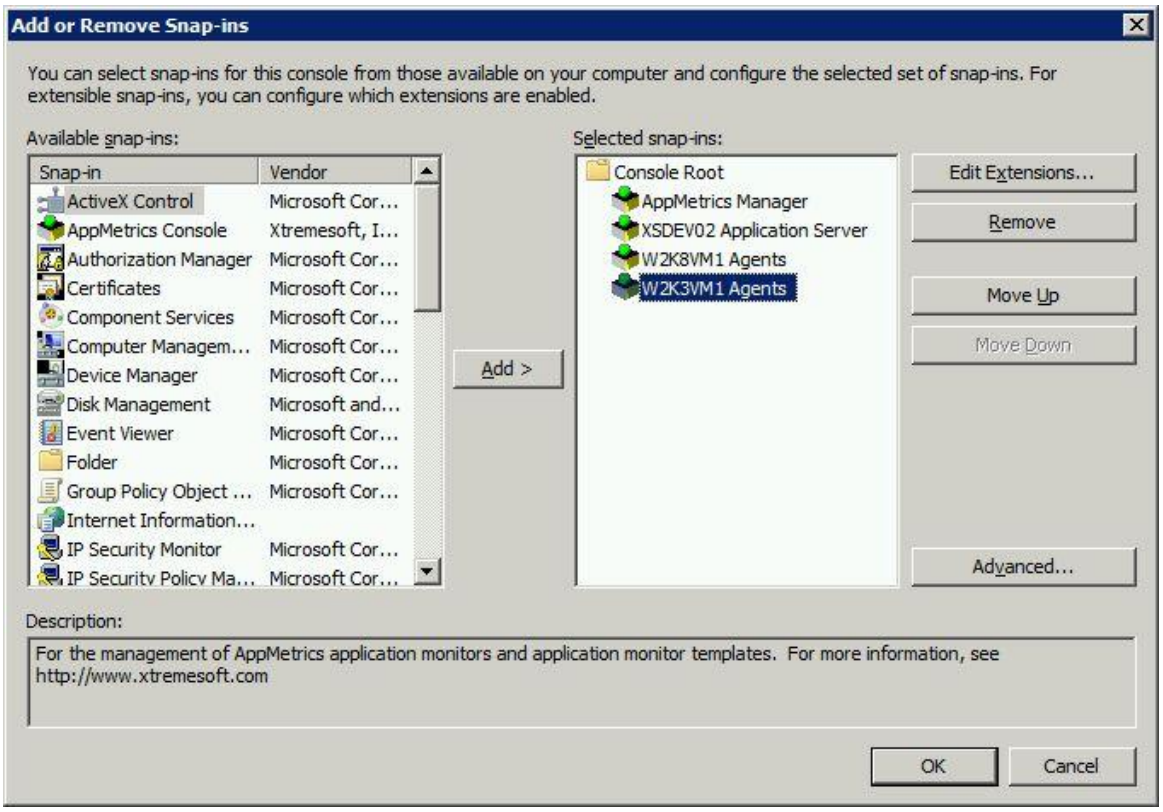


Figure 3-62 Changing the order of the AppMetrics Console Instances

The order in which the *AppMetrics Console* instances are displayed in the MMC can be modified by using the **Move Up** and **Move Down** buttons on the ***Add or Remove Snap-Ins*** dialog.

Removing an AppMetrics Console Instance

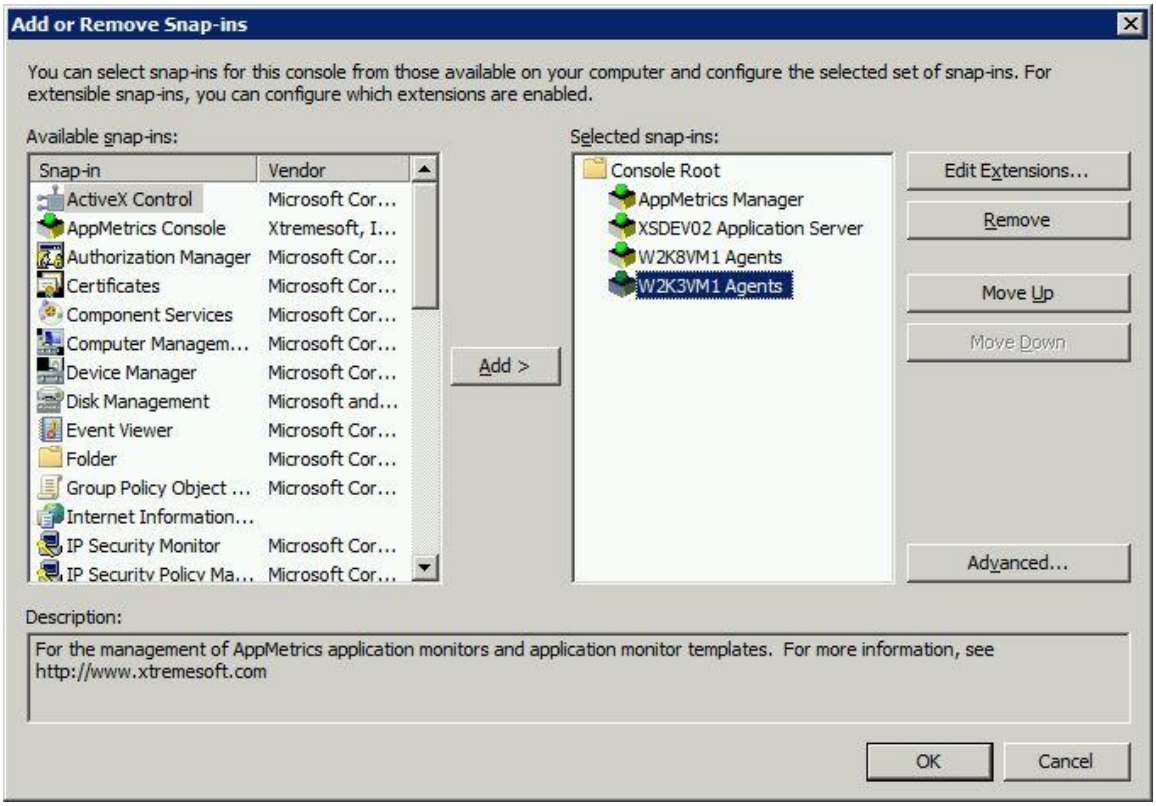


Figure 3-63 Removing an AppMetrics Console Instance

AppMetrics Console instances can be removed from the MMC by using the **Remove** button on the ***Add or Remove Snap-ins*** dialog.