



X t r e m e s o f t

AppMetrics Essentials

Version 3

Participant and Instructor Guide

© 2007 Xtremesoft, Inc. All Rights Reserved.

The user is responsible for complying with all applicable copyright laws. Without limiting the rights under copyright, no part of this documentation may be photocopied or reproduced in any form without prior written consent from Xtremesoft, Inc.

Xtremesoft may have patents, patent applications, trademarks, copyrights, or other rights to intellectual property covering items mentioned in this document. Except as expressly provided in any written license agreement from Xtremesoft, the furnishing of this document does not grant to the user any license to these patents, trademarks, copyrights, or other intellectual property.

AppMetrics is a registered trademark of Xtremesoft, Inc.

Microsoft and Windows are registered trademarks of Microsoft Corporation.

The names of actual companies and products mentioned within this documentation may be the registered trademarks or trademarks of their respective owners.

Contents

Section 1: Today's Objectives	1-1
Understand the Problems that AppMetrics Solves	1-1
Installation	1-1
Configuration	1-1
Notifications	1-1
Analyzing reports and other metrics.....	1-2
Strategies for managing your applications	1-2
Section 2: Introduction to AppMetrics	2-1
Concepts of AppMetrics for Transactions	2-1
Monitoring v. Debugging	2-1
A Look back at the origins - What are Transactions?	2-2
Transactions.....	2-2
The A-C-I-D Test.....	2-3
Making Components Transactional	2-3
DTC (Distributed Transaction Coordinator)	2-4
Definitions	2-5
Manager-Agent Model.....	2-7
AppMetrics Agents	2-7
AppMetrics Managers	2-8
AppMetrics Remote Clients	2-8
Section 3: Installing AppMetrics	3-1
Prerequisites for a Manager Machine	3-1
Prerequisites for an Agent Machine.....	3-2
Prerequisites for a Reports and Console Machine.....	3-2
AppMetrics Architecture.....	3-3
N-Tier	3-3
DCOM and Firewalls	3-3
Multiple Windows Domains.....	3-5
Domains without trusts.....	3-5

AppMetrics Security Model.....	3-6
NT v. Win2K.....	3-7
Security in Reports and Remote Console clients	3-9
Security in MTS/COM+ Components	3-10
Security issues in AppMetrics Data Upload & Reporting.....	3-12
Section 4: Lab 1 – Installing AppMetrics	4-14
Objectives	4-14
Preparation	4-14
Step 1: Install & Configure <i>Sample Bank</i>	4-15
Step 2: Install an AppMetrics for Transactions Agent.....	4-19
Step 3: Install an AppMetrics for Transactions Manager	4-20
Step 4: Install an AppMetrics for Transactions Console/Reports User	4-20
Review Questions	4-20
Section 5: Creating Monitors	5-20
Production v. Diagnostic Monitors	5-20
Creating a Production Monitor	5-20
Creating a Diagnostic Monitor	5-20
Adding an Agent Monitor	5-20
Section 6: Lab 2 - Creating Production & Diagnostic Monitors	6-20
Objectives	6-20
Prerequisites	6-20
Preparation	6-20
Step 1: Creating a Production Manager Monitor.....	6-20
Step 2: Creating a Diagnostics Manager Monitor.....	6-20
Review Questions	6-20
Section 7: Configuring Monitors	7-20
Selecting Packages/Applications.....	7-20
Set Intervals	7-20
Set Detail Levels	7-20
Set Default Thresholds	7-20
Set Specific Thresholds.....	7-20

Set All Transactions Thresholds.....	7-20
Set Component Thresholds.....	7-20
Set Transactions Thresholds	7-20
Set Package/Application Process Thresholds	7-20
Naming Transactions	7-20
Section 8: Monitoring Applications.....	8-20
View Package Metrics	8-20
View Component Metrics	8-20
View Transaction Metrics.....	8-20
In-Calls Snapshot	8-20
Using the MTS/COM+ Console.....	8-20
Section 9: Lab 3 - Configuring Monitors & Viewing Metrics	9-20
Objectives	9-20
Prerequisites	9-20
Preparation	9-20
Step 1: Configuring a Production Monitor.....	9-20
Step 2: Configuring a Diagnostics Monitor.....	9-20
Step 3: View Metrics on an AppMetrics Production Monitor.....	9-20
Step 4: View Metrics on an AppMetrics Diagnostics Monitor.....	9-20
Review Questions	9-20
Section 10: Configuring Notifications	10-20
What is a Notification?	10-20
Delivery Mechanisms	10-20
SMTP	10-20
SNMP	10-20
Event Log.....	10-20
Custom Component.....	10-20
Configuring a Delivery Mechanism	10-20
How Tab	10-20
Who Tab.....	10-20
Logging Tab.....	10-20

Section 11: Lab 4 - Creating Event Log Notifications	11-20
Objectives	11-20
Prerequisites	11-20
Preparation	11-20
Step 1: Configuring an Event Log Notification.....	11-20
Step 2: Adjust Benchmarks and Thresholds to Trigger Notification	11-20
Step 3: Verify the Notification.....	11-20
Review Questions	11-20
 Section 12: AppMetrics Reports	 12-20
Rotation.....	12-20
Log vs. Database	12-20
Automatic Rotation	12-20
Manual Rotation.....	12-20
Database Sizing Issues.....	12-20
Viewing the Reports.....	12-20
Selecting your Data	12-20
Navigating the workbook	12-20
Production Reports Definitions	12-20
Diagnostic Reports Definitions	12-20
 Section 13: Lab 5 - Viewing and Analyzing AppMetrics Reports	 13-20
Objectives	13-20
Prerequisites	13-20
Preparation	13-20
Step 1: Configuring Monitor Rotation Schedules.....	13-20
Step 2: View a Production Monitor Report.....	13-20
Step 3: View a Diagnostics Monitor Report.....	13-20
Comparing Component & Method level detail to Component-only level detail in AppMetrics Reports.....	13-20
Review Questions	13-20
 Section 14: Issues in Analysis	 14-20
Library Packages	14-20

Monitoring IIS applications with MTS/COM+	14-20
Section 15: Managing Applications	15-20
Establishing Accurate Thresholds.....	15-20
Diagnostics Monitors in-depth.....	15-20
Analyzing Trends.....	15-20
Section 16: Lab 6 – Analyzing your Applications with AppMetrics Reports	16-20
Objectives	16-20
Prerequisites	16-20
Preparation	16-20
Part 1: Discover applications that may be using excessive resources	16-20
Part 2: Recognize Applications with potential memory leaks.....	16-20
Part 3: Recognize failing packages/applications and components.....	16-20
Part 4: Locating the longest running method in a long duration transaction.....	16-20
Section 17: Lab 7 – Querying AppMetrics Databases	20
Objectives	20
Prerequisites	20
Preparation	20
Part 1: Using SQL Query Analyzer.....	20
Part 2: Querying a Production Monitor	20
Part 3: Querying a Diagnostics Monitor	20
Part 4: Locate Crashing Components and Methods	20

Section 1: Today's Objectives

A dark blue slide with a purple and green starburst graphic on the left. The text 'Xtremesoft' is in purple at the top left, and 'Training Objectives' is in yellow at the top center. A list of five bullet points is in white. At the bottom, there are three small white text items: 'Rev 8/13/07', 'Copyrighted material', and '3'.

Xtremesoft

Training Objectives

- Understand the problems that AppMetrics will solve
- Configuring Production & Diagnostic monitors
- How and when to set Notifications
- Analyzing reports and other metrics
- Strategies for managing your applications

Rev 8/13/07 Copyrighted material 3

Understand the Problems that AppMetrics Solves

You will learn how AppMetrics offers the following solutions:

- Alerting
- Monitoring
- Managing
- Diagnostics
- Track long-term performance trends (capacity planning)
- Demonstrate service levels delivered (production reporting)

Installation

You will learn to install an AppMetrics Agent, Manager, and Console/Reports setup on different machines.

Configuration

After installing AppMetrics on the various types of machines, you will learn how to create and configure Production and Diagnostic monitors, which collect data on packages/applications.

Notifications

Part of the configuration process deals with setting up notifications for specific conditions in your packages/applications. You will learn how to set up the delivery mechanisms for sending these notifications.


Analyzing reports and other metrics

After you configure your monitor and run it for a while, the monitor will begin generating metrics about your packages/applications. You will learn how to view these metrics in the historical reports, which show the performance of your packages/applications over time.

Strategies for managing your applications

Based on the metrics, you can learn important information about your packages/applications. You will learn tips on how to use the metrics to better manage your packages/applications.

Section 2: Introduction to AppMetrics



Xtremesoft

Introduction to AppMetrics

AppMetrics Concepts

Monitoring and management v. Debugging

- Monitoring and management – Non-intrusive availability, performance and diagnostics
- Debugging – Intrusive

Rev 8/13/07 Copyrighted material 5

Concepts of AppMetrics for Transactions

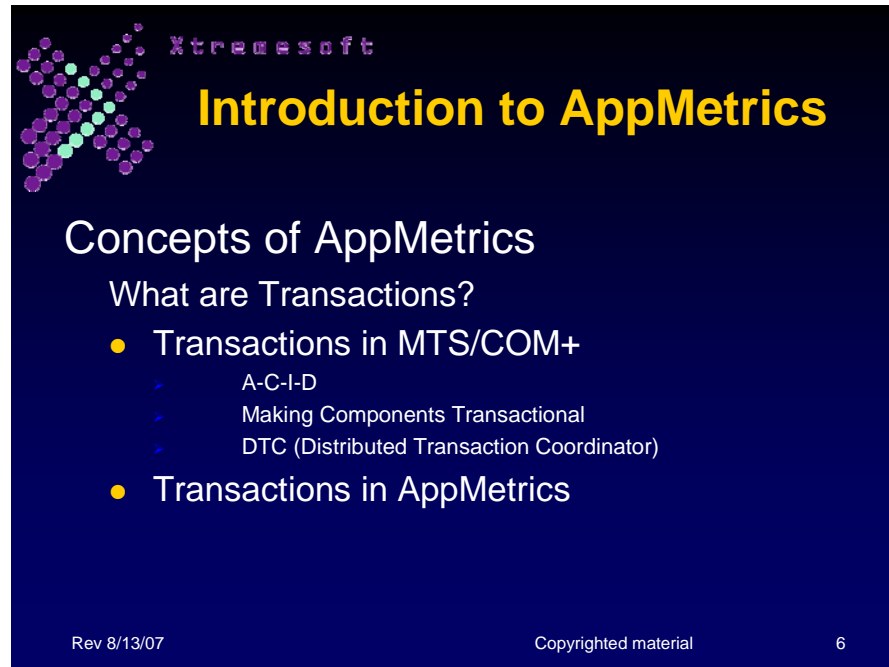
Monitoring v. Debugging

AppMetrics for Transactions is a monitoring system for COM+ and .net Serviced Component applications running in production environments. AppMetrics generates metrics showing the overall health and performance of your packages/applications and the machines on which they run.

AppMetrics is not a debugging system and does not embed itself into your code to see how your packages/applications run.

AppMetrics collects data about your packages/applications unobtrusively, as they run within the COM+ system. AppMetrics does this by utilizing the Microsoft COM+ eventing subsystem.

A Look back at the origins - What are Transactions?



Transactions

It's probably fair to say that "Microsoft Transaction Server" was a bit of a misnomer, since much of what MTS/COM+ (and now .net Servicedcomponents) provides has nothing to do with transactions. Instead, a large part of the functionality focuses on making it easier to build scalable COM servers. Support for transactions is a requirement for many kinds of servers, and it's certainly an important part of what MTS/COM+ and .net servicedcomponents offers.

What exactly does it mean to "use transactions"? In general, a transaction allows you to group together two or more units of work into a single, indivisible unit.

For example, in an order entry scenario, a record in the inventory database must be changed for each item the client requests. For a client ordering several items, several records will be changed. If the order is submitted, all of those changes must be made permanent. If the client cancels the order, all of those changes must be rolled back, leaving things just as if the order had never existed. Allowing some of the order's changes to persist while others do not would leave the data in an inconsistent state. Rather than worrying about this itself, an application can rely on COM+ to ensure that either all the changes occur or none of them do.

So when should an application use transactions? It's easy: if the COM object needs to make two or more changes, such as updates in a database, and the developer wants all of those changes to either succeed or fail—partial success isn't allowed - use transactions.

Without transactions, error recovery is extremely difficult, especially when multiple objects update multiple databases. The possible combinations of failure modes are too great even to consider. Transactions simplify error recovery. Resource managers automatically undo the transaction's work, and the application retries the entire business transaction.

Transactions also provide a simple concurrency model. Because a transaction's isolation prevents one client's work from interfering with other clients, you can develop components as though only a single client executes at a time.

The A-C-I-D Test

MTS/COM+ Transactions adhere to the ACID properties for transactions:

- **Atomicity** ensures that all the updates completed under a specific transaction are committed and made durable, or that they get aborted and rolled back to their previous state.
- **Consistency** means that a transaction is a correct transformation of the system state, preserving the state invariants.
- **Isolation** protects concurrent transactions from seeing each other's partial and uncommitted results, which might create inconsistencies in the application state. Resource managers use transaction-based synchronization protocols to isolate the uncommitted work of active transactions.
- **Durability** means that committed updates to managed resources, such as a database record, survive failures, including communication failures, process failures, and server system failures. Transactional logging even allows you to recover the durable state after disk media failures.

Making Components Transactional

Every COM+ component has a transaction attribute that is recorded in the COM+ catalog. COM+ uses this attribute during object creation to determine whether the object should be created to execute within a transaction, and whether a transaction is required or optional.

When it's installed, a COM+ component can have its transaction attribute set appropriately. An administrator can do this using the Component Services console. The transaction attribute is set using a simple dialog box. Choosing "Requires a transaction" or "Requires a new transaction" will guarantee that all data accesses made by this component will be committed or rolled back as a unit (and in some cases, "Supports transactions" will have the same effect). A developer doesn't have to write any extra code to handle transactions.

DTC (Distributed Transaction Coordinator)

COM+ use the services of the Microsoft Distributed Transaction Coordinator (DTC) for transaction coordination. DTC is a system service that coordinates transactions that span multiple resources and machines. With DTC work can be committed as a single transaction.

DTC was initially released as part of Microsoft SQL Server version 6.5, and is included as part of COM+. DTC implements a *two-phase commit protocol* that ensures that the transaction outcome (either *commit* or *abort*) is consistent across a transaction. It ensures the integrity of the data during the transaction, and it rolls back the data to its original state if the transaction is aborted.


AppMetrics and the DTC.

A transactional component coordinated by DTC may sometimes take longer than anticipated to complete its work - certainly longer than most non-transactional components. To assist in the analysis of long durations, AppMetrics reports the percentage of a transaction & component's duration that is involved in a DTC mediated transaction.

Transactions in AppMetrics

AppMetrics uses the term “transactions” in a completely different context to COM+. Within AppMetrics, a transaction refers to calls from client processes or from non-monitored packages/applications into your monitored components. This is different from the transactions within DTC.

A transaction in AppMetrics usually corresponds to the lifecycle of a root method invoked on a COM+ component. The method may delegate calls to additional methods (sometimes on additional components), but the ‘transaction’ corresponds to the root method invoked. A transaction may also correspond to the lifecycle of a COM+ component instance (from object creation until when it goes out of scope). This will be explained in more detail in Section 7 “Configuring Monitors”.



Xtremesoft

Introduction to AppMetrics

Concepts of AppMetrics - Definitions

- COM+
- .net Servicedcomponents
- MTS/
- Components
- Package/Application
- Thresholds
- Abort
- Crash
- Start
- Stop

Rev 8/13/07 Copyrighted material 7

Definitions

- **COM+:** A set of services for managing the activity and resource usage of packages/applications for Windows 2000 and 2003.
- **.net Servicedcomponents:** A set of services for managing the activity and resource usage of packages/applications for .net based applications.
- **MTS:** A set of services for managing the activity and resource usage of packages/applications. MTS is the version for Windows NT4
- **Components:** Code modules that serve as building blocks for packages/applications. They contain methods that other processes and components can call to perform work.
- **Package/Application:** A group of related components. The components are managed and executed within the context of a single COM+ process on a machine.
- **Thresholds:** Maximum acceptable levels of activity. Any activity levels above the thresholds can be categorized as either warning or notification conditions. AppMetrics can respond when these thresholds are exceeded.
- **Abort:** AppMetrics reports a Transaction "Abort" when the underlying component reports an abort. This can mean

IObjectContext::SetAbort was called or if the Distributed Transaction Coordinator (DTC) is used it may have aborted the transaction for other reasons.

- **Crash:** AppMetrics reports a "Crash" when a COM+ server Package/Application process (MTX.EXE or DLLHOST.EXE) is terminated abnormally.
- **Start:** AppMetrics reports a "Start" when a new process is created for a COM+ Package/Application. This corresponds to the creation of a new MTX.EXE or DLLHOST.EXE instance.
- **Stop:** AppMetrics reports a "Stop" when a COM+ Package/Application is shutdown (manually in the MTS/COM+ explorer UI, because its idle time has been reached, or programmatically).



Introduction to AppMetrics

Concepts of AppMetrics Manager – Agent model

- AppMetrics Agents
 - Collects package/application data - no analysis
 - Communicates with COM+/MTS Eventing subsystem
 - No caching of data (only small buffer), depends on reliable connections
- AppMetrics Managers
 - Average/aggregate data
 - Write to logs, rotate to database
- AppMetrics Remote Clients
 - Just a remote console to AppMetrics Manager - No processing or caching on remote client machine
 - No security

Rev 8/13/07 Copyrighted material 9

Manager-Agent Model

The Manager-Agent model enables AppMetrics to monitor packages/applications while using a minimum amount of system resources. It achieves this by performing only the minimum number of functions required on the package/application machine while performing the more system-intensive tasks on a separate Manager machine.

The following illustration depicts the Manager-Agent model:

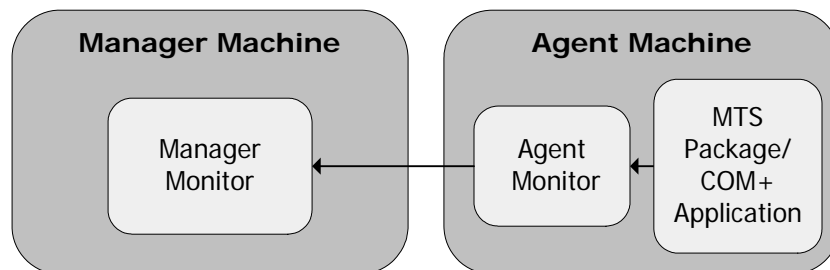


Figure 1

AppMetrics Agents

In the Manager-Agent model, an agent monitor runs on the package/application machine (Agent machine). The agent monitor performs the following tasks:

- It collects data about the package(s)/application(s)

- It reports this data to the manager monitor on the Manager machine over a reliable network connection using DCOM/RPC protocols.
- It caches virtually no data on the Agent machine, flushing its data to the manager approx. every five seconds.

As a result, the agent's work on the Agent machine is almost negligible, freeing up system resources on the machine for running and managing the packages/applications.

AppMetrics Managers

The Manager monitor collects, analyzes, and processes the metric data from the Agent—all performed directly on the a Manager machine. This frees up resources on the Agent machine for running and managing the packages/applications.

The Manager monitor stores this data in a set of logs on the Manager machine. Periodically, it uploads these logs into the MSDE/SQL Server database that corresponds to that monitor.

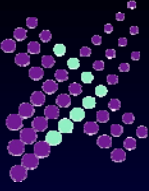
AppMetrics Remote Clients

An AppMetrics Remote Client, also referred to as a 'remote console', does *not* perform any analysis or storage of Agent or Manager data. It is just a remote console that permits the user to directly view and manage the data and configuration of a Manager monitor. It also includes the Excel worksheets necessary to report on AppMetrics Production and Diagnostic monitors.

Because it is only a remote console, the AppMetrics Remote Client does not need to run any AppMetrics services. The Remote client communicates directly with any AppMetrics Manager via DCOM/RPC using the security account of the currently logged on Interactive User.

Note: Consider installing only the AppMetrics Reports worksheets on the machines of AppMetrics users who have no reason to be directly accessing an AppMetrics Manager

Section 3: Installing AppMetrics



Xtremesoft

Installing AppMetrics

- Prerequisites
 - Hardware
 - Software
- AppMetrics Architecture
 - N - tier
 - DCOM over TCP/IP as primary protocol
 - DCOM across a firewall
 - AppMetrics in a multiple domain environment
 - One and two-way trusts
 - Domains without trusts

Rev 8/13/07 Copyrighted material 10

Prerequisites for a Manager Machine

Hardware

- Intel® Pentium® and Celeron™ processors only
- Disk space:
 - Install: 50 MB
 - Program Files: 35 MB
 - Database Manager: 15 MB
 - Archival Data: 10 GB, varies per deployment
- 20 MB RAM, plus 15 MB per monitor

Software

- Microsoft Windows® 2000 SP3, 2003 Server, and XP
- Microsoft SQL Server™ (2000 or 2005)
- For AppMetrics Reports: Microsoft Excel® 2000, 2003, or XP
- For AppMetrics Notifications via SNMP on Windows NT 4.0: The SNMP Service must be installed prior to installing AppMetrics
- For Help Documentation: Adobe® Acrobat® Reader™

Prerequisites for an Agent Machine

Hardware

- Intel® Pentium® and Celeron™ processors only
- Disk space:
 - Install: 50 MB
 - Program Files: 35 MB
- 20 MB RAM, plus 15 MB per monitor

Software

- Either of the following Operating Systems:
 - Microsoft Windows NT 4.0, SP 6a
 - Microsoft Windows® 2000 SP3 or 2003 or XP

Prerequisites for a Reports and Console Machine

Hardware

- Intel® Pentium® and Celeron™ processors only
- Disk space:
 - Install: 50 MB
 - Program Files: 35 MB

Software

- Any of the following operating systems:
 - Microsoft Windows® 2000 SP3 or 2003 or XP
- For AppMetrics Reports: Microsoft Excel® 2000, 2003, or XP
- For Help Documentation: Adobe® Acrobat® Reader™

AppMetrics Architecture

N-Tier

As already mentioned earlier, AppMetrics uses a Manager-Agent model to monitor packages/applications and their host machines. An additional, optional tier is the Reports and Console machine.

The following illustration shows the relationship between the different AppMetrics machines:



Figure 2

The Reports & Console machine contains the following items:

- The AppMetrics Reports module, which lets you view historical data about your packages/applications.
- The AppMetrics console, which lets you create, configure, and manage your monitor. It also displays the latest set of updated metrics about your packages/applications.

The Reports and Console machine is optional because its functions are also available on the Manager machine.

DCOM and Firewalls

AppMetrics uses the DCOM (distributed COM) protocol to communicate between its different parts running on separate machines.

DCOM over TCP/IP. By default, DCOM is configured to use UDP as the primary protocol. However, the UDP protocol is not the most reliable transport, and will contribute to many errors in any DCOM implementation, including AppMetrics.

Xtremesoft strongly recommends that you configure *all* AppMetrics Managers, Agents, and Remote Clients to utilize ‘Connection-oriented TCP/IP’ as the primary protocol for DCOM.

To make this change, or to confirm your existing settings, open the DCOM configuration application by typing **dcomcnfg** at a command-line prompt. When the Distributed COM Configuration Properties dialog box opens, click on the Default Protocols tab, and note the current primary DCOM protocol (i.e. the protocol at the top of the listing). If TCP/IP is not the primary protocol, promote it to the top of the list selecting it, and clicking on the Move Up button until it is at the top. (See Figure 3 below).

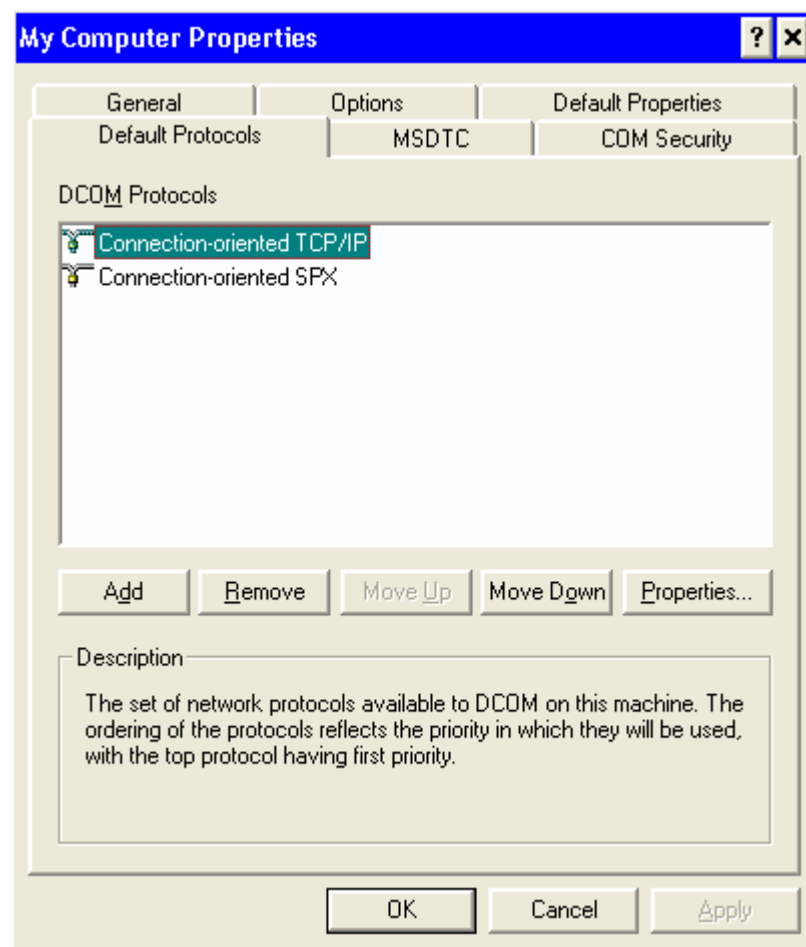


Figure 3

DCOM across Firewalls. To test and configure DCOM to work across a firewall, you will need the following:

- a) Basic IP connectivity. Test this by ensuring that you can PING the Manager machine from the Agent machine, and vice-versa.
- b) DNS/Netbios connectivity (needed when entering a 'pretty name' in the Manager configuration). Test this by PINGing the Manager with its computer name from the Agent, and PINGing the Agent with its computer name from the Manager.
- c) DCOM port configuration. DCOM uses SCM to dynamically assign ports. SCM uses UDP/TCP, specifically UDP port 135, and TCP port 135. Additional DCOM requirements, where necessary, are spelled out in Microsoft's White Paper "Using Distributed COM with Firewalls" found at

<http://support.microsoft.com/KB/248809>

Multiple Windows Domains

AppMetrics is able to work across domains as long as a trust relationship exists between these domains. At a minimum, AppMetrics can work in a one-way trust relationship. This requires the Agent machine to be in a domain that trusts the Manager machine's domain. Figure 4 below illustrates the principles of domain trusts in AppMetrics.

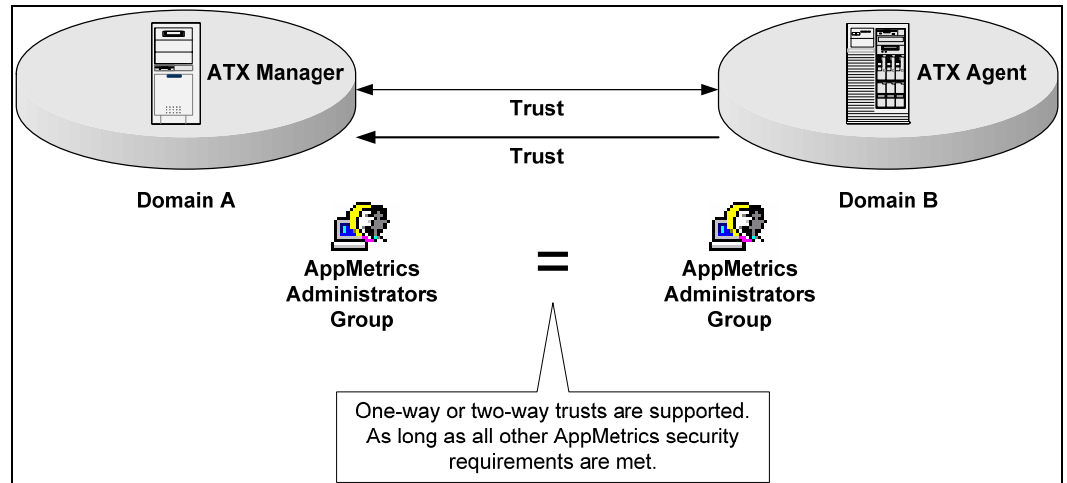


Figure 4


Domains without trusts

AppMetrics is able to work across two domains without a trust relationship as long as each Manager and Agent utilizes the same local accounts in each machine's AppMetrics Administrators Group.

For example, if the AppMetrics Manager machine uses a local account named 'PSmith' with password 'appmtrc02\$', and this account is included in its local AppMetrics Administrators Group, then the AppMetrics Agent machine must also include a local account named 'PSmith' with password 'appmtrc02\$' in its local AppMetrics Administrators Group. (Don't forget that Windows passwords are case sensitive).

This duplication of local account names and passwords must hold for **every** account in the local AppMetrics Administrators Group in both Manager and Agent machines. If the Manager machine is monitoring more than one Agent, then the Manager must include a duplicate of every local account in that other Agent's AppMetrics Administrator Group.

Obviously, this configuration can be difficult to set up and administer, and is not a recommended configuration.



Installing AppMetrics

AppMetrics Security Model

- Security Configuration
- Security issues in data uploads and reporting
- Security issues

Rev 8/13/07
Copyrighted material
12

AppMetrics for Transactions: Manager - Agent System

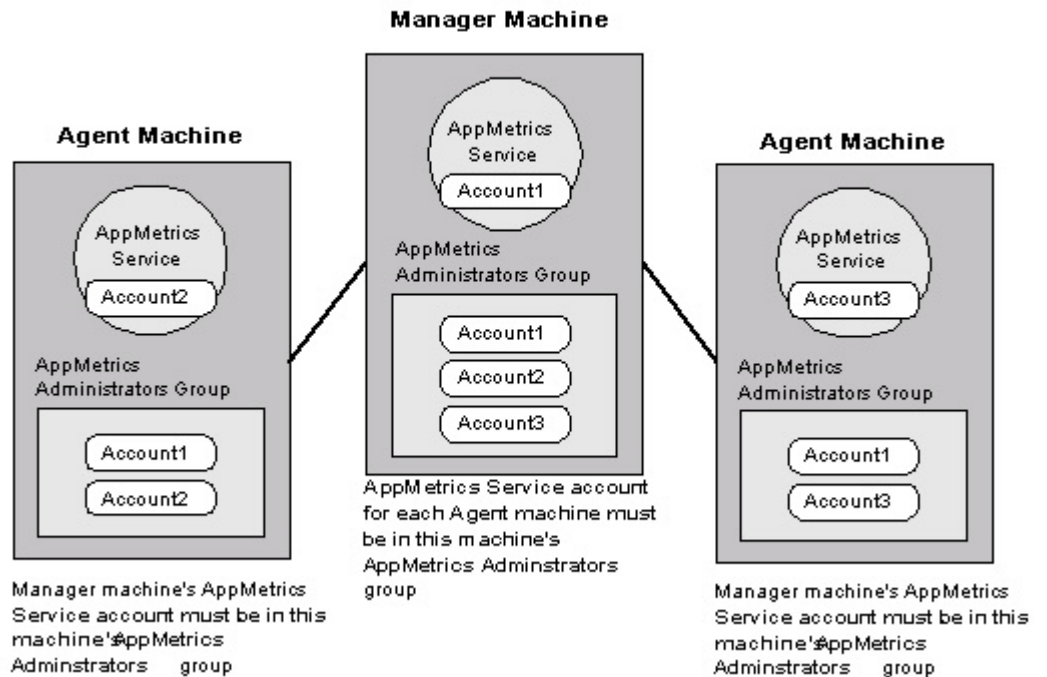


Figure 5

AppMetrics Security Model

The AppMetrics security model relies largely on the AppMetrics Administrators privilege. If you want to access AppMetrics on a machine,

your account must have the AppMetrics Administrators privilege on the machine. (See Figure 5).

The same applies to the AppMetrics services running on Manager and Agent machines. The service on a Manager machine must access the service on an Agent machine, and vice-versa. As a result, the service on one machine must have the AppMetrics Administrators privilege on the other machine.

If you plan to install AppMetrics on multiple machines within your network, you should consider creating and using the **AppMetrics Admins global group** in your domain. This group can simplify management for the following items:

- The AppMetrics Console users who must access different machines
 - The AppMetrics service accounts that must access different machines
- As Figures 6 & 7 illustrate, users and accounts added into the AppMetrics Admins global group will be inherited automatically into the AppMetrics Administrators local group on any AppMetrics Agent or Manager machine within that domain or another trusted domain.

NT v. Win2K

The AppMetrics security model differs slightly between Windows NT 4 and Windows 2000. The difference occurs in the accounts used by the AppMetrics service on either operating system. On Windows 2000, the AppMetrics service uses its run-as account for the following purposes:

- To log on to the machine and start in an unattended mode.
- To access other AppMetrics machines.

On Windows NT 4, the AppMetrics service uses two accounts: Its run-as account and the remote-access account.

The service run-as account on Windows NT 4 lets the service log on to the machine. It also provides access to the local MTS packages running under different identities.

By default, the service run-as account on Windows NT 4 must use the local System Account. You cannot assign any other account to the run-as account. (See Figures 6 & 7).

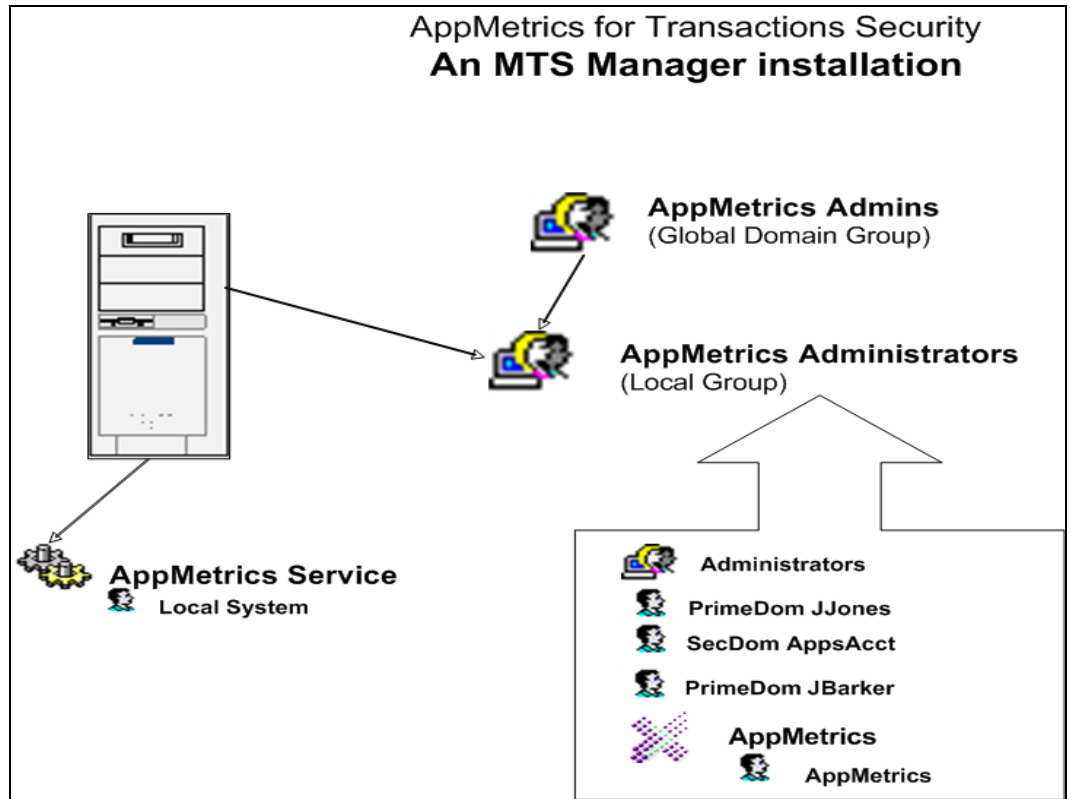


Figure 6

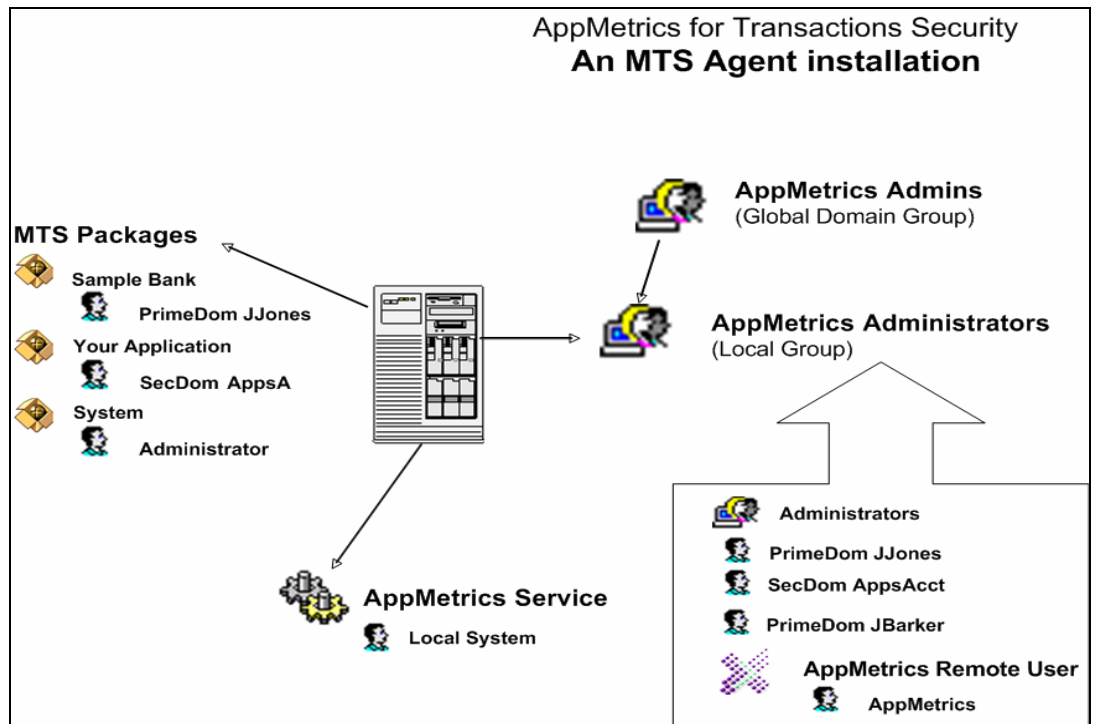


Figure 7

The other account used by the AppMetrics service on Windows NT4 is the remote-access account. This account enables the service to access other AppMetrics machines.

Note in the COM+ Agent Security diagram (Figure 8) that the Remote Access account from NT 4 is not present. This is because COM+ can use the AppMetrics account (or any other Windows account) as the 'Service Run As' security account under which the AppMetrics Service will run.

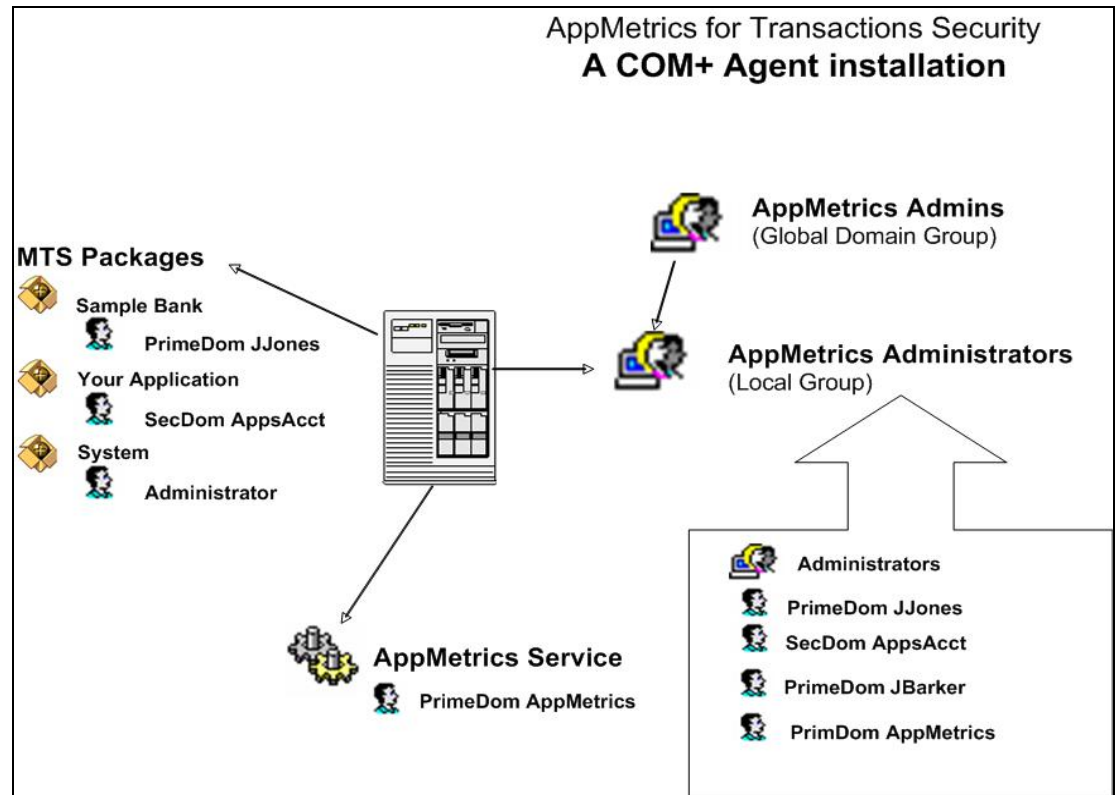


Figure 8

Security in Reports and Remote Console clients

Security in an AppMetrics 'Reports and Console' setup is handled entirely via the context of the currently logged-in Interactive User account.

These clients **do not** run any applications to be monitored by AppMetrics. So there is no need for an AppMetrics Service on these clients and no need to configure AppMetrics security groups on the client machine.

The N-tier security diagram below (see Figure 9), demonstrates the simple rule that the interactive user logged into a reports or remote console client must have their domain account included in the AppMetrics Administrators local group on both the AppMetrics Manager and Agent machines.

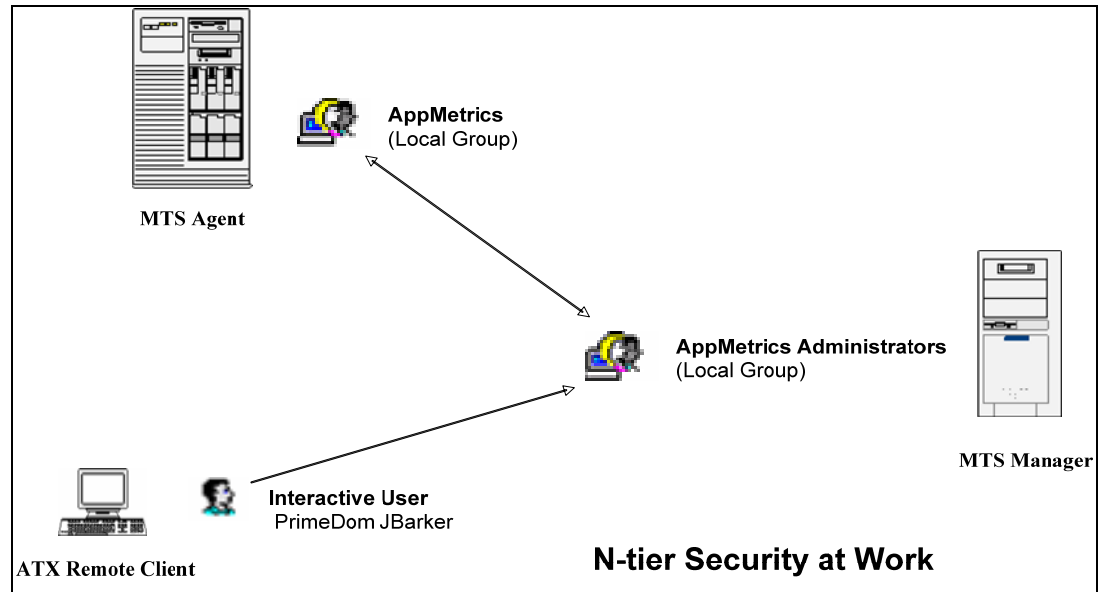


Figure 9

Security in MTS/COM+ Components

AppMetrics is capable of monitoring an MTS/COM+ package running on Windows NT 4/Windows 2000. To accomplish this, the package's identity, which is a Windows user account, must be a member of the AppMetrics Administrators local group.

When you install an AppMetrics Agent on your application server machine, the package identities will automatically become members of the local AppMetrics Administrators group.

The AppMetrics Agent security configuration illustration above (See **Figures 6 & 7**), indicates the security identities assigned to the packages 'Sample Bank', 'Your Application', and 'System'. When configuring this sample Agent, the System Administrator must ensure that the security identities on these packages are included in the local AppMetrics Administrators group on both the AppMetrics Agent and AppMetrics Manager machine.

A package identity is configured via the Identity tab of the package properties window. (See **Figure 10** below).

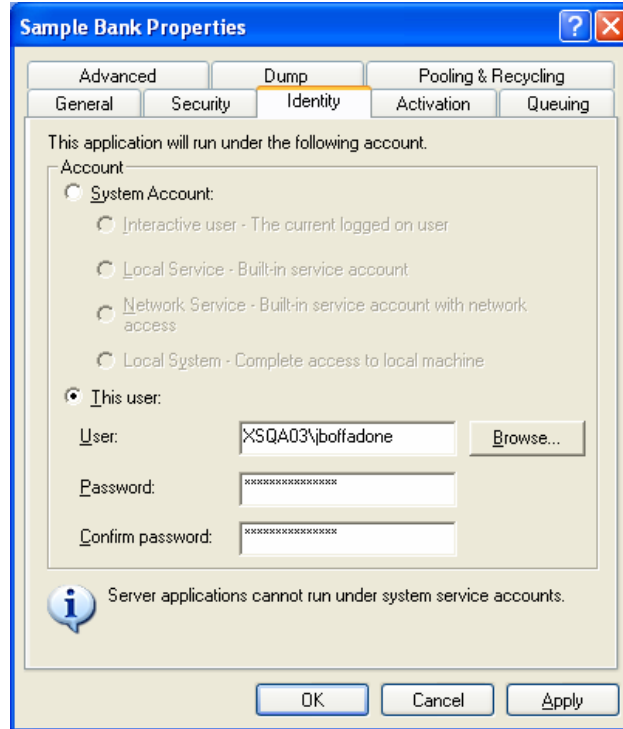


Figure 10

However, if you later add any package to the machine after the install, and you want to monitor it, be sure to set the package identity with an account that belongs to the local AppMetrics Administrators group.

While not recommended as a permanent configuration, it is possible for AppMetrics to monitor a package whose identity is set to the Interactive user. When the identity is set this way, AppMetrics requires a user to log into the machine with an account that belongs to the local AppMetrics Administrators group. The user can then start the package, and AppMetrics will be able to monitor it. But if the user logs in with an account that does not belong to the group, then AppMetrics cannot monitor the package.

The MTS/COM+ Catalog – MTS/COM+ store configuration information about MTS/COM+ components in the MTS/COM+ catalog. This catalog is accessed by AppMetrics via the System package. If the identity for the System package is set to the **Interactive user** on an NT 4 agent machine, a manager monitor may fail to obtain catalog information from the machine when no one is logged into it. To avoid this problem, the AppMetrics install will change the identity of the System package to the same account as the AppMetrics service run-as account. However, if you prefer to use another account for the identity, make sure the account has both Administrators and AppMetrics Administrators privileges on the local machine. The Administrators privilege is an MTS requirement.

Note: On Windows 2000 Agents the System package uses the local system identity and cannot be changed, therefore the COM+ catalog will always be available.

Security issues in AppMetrics Data Upload & Reporting

AppMetrics utilizes SQL Server security accounts when performing uploads of data logs from hard disk to SQL Server databases, and to subsequently report against those databases.

The information below details the specific SQL Server accounts that AppMetrics will create and/or use as it performs these functions.

SQL Administrator Login - during installation AppMetrics will prompt for a SQL Server account with System Administrators privs:

- **On SQL 2000, SQL 2005, and SQL Express**

Database Creator

Security Administrators

This SQL Admin Login need not be the 'sa' account, but it must exist and have these necessary privs. This account is used to create two other SQL Logins ('AppMetrics' and 'appmetrics_reader').

AppMetrics continues using this SQL Admin Login for creating and deleting databases (after install). **Note:** After the install completes, to use a different SQL Admin Login or to update the password for the AppMetrics SQL Admin Login you use the Advanced Properties in AppMetrics. (Open the AppMetrics Snap-In UI, right click "AppMetrics Console", choose Properties, change the view level to "Advanced", click "Apply". Then right click "AppMetrics Console" and choose Properties (again).

SQL User Login - during installation AppMetrics will prompt for the name and password for a new SQL User Login account it creates for use when writing to AppMetrics databases. The default name for this SQL User Login is 'AppMetrics' (but you can change this at install). Minimum required privs are:

- **On SQL 2000, SQL 2005, and SQL Express**

Bulk Insert Administrators

Note: After the install completes, to use a different AppMetrics SQL User Login or to update the password for the AppMetrics SQL User Login you use the Advanced Properties in AppMetrics. (Open the AppMetrics Snap-In UI, right click "AppMetrics Console", choose

Properties, change the view level to "Advanced", click "Apply". Then right click "AppMetrics Console" and choose Properties (again) - you should now have the option to change the SQL User Login.

SQL appmetrics_reader account - during installation AppMetrics will create a new SQL account called 'appmetrics_reader' for use in creation of the AppMetrics Reports. Required privs automatically assigned to the 'appmetrics_reader' account are:

- **On SQL 2000, SQL 2005, and SQL Express**

db_datareader privs

No system-wide privs are required.

When a monitor and its database are created the appmetrics_reader account is given db_datareader privs in the new database. By default the password is blank. If the password is changed in SQL Server it also must be changed in the hidden 'State' worksheet in the AppMetricsForTransactions.xls file.



Section 4: Lab 1 – Installing AppMetrics

Objectives

- Install **AppMetrics Agent, Manager, and Console/Reports**
- Install *Sample Bank* as a load-generating MTS application

Preparation

The labs in this course are best performed in groups of 4 students, however they can accommodate teams of between 3 and 5 students.

You must have at least three computers available to perform this Lab. One computer will be designated the application server, and it will have the *Sample Bank* application installed and initiated to generate load, as well as have the **AppMetrics Agent** installed to monitor that application. Another computer will install the **AppMetrics Manager**. At least one more computer should have an **AppMetrics Console/Reports** client installed.

Each team should receive from the instructor one CD-ROM that includes the AppMetrics for Transactions 3.X installation and the *Sample Bank* application.

Ask your Instructor which domain user account has been set up for your use with these Labs (generally Train1 – Train20). Your Instructor should provide you with a designated user account and password.

Students performing the Agent and Load Generation installation should follow Steps 1 & 2 below.

Students performing the Manager installation should skip to Step 3.

Students performing the Console/Reports installation should skip to Step 4.

Note: In this training lab environment, all the machines have been pre-screened for compliance with AppMetrics' hardware, software, and security requirements. In an actual installation you would have to ensure compliance of all machines before attempting to install.

Step 1: Install & Configure *Sample Bank*

1. Copy the **Bank** directory from the CD-ROM to the root directory on your machine.
2. Open a Command Prompt window, and register the DLLs **Bank.dll** and **vcacct.dll** from the command line (see Figure 2).

From the command line, change directory to **%root%\Bank**, and invoke **regsvr32** for both **bank.dll** and **vcacct.dll**, type:

```
Regsvr32 bank.dll <enter>
```

```
Regsvr32 vcacct.dll <enter>
```

Note that each registration will provide a confirmation dialog box.

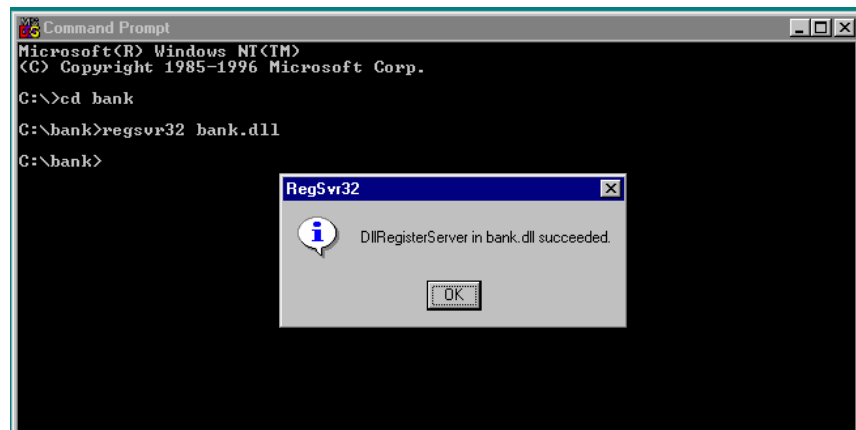


Figure 2

3. Open the Visual Basic script **instdll.vbs**, found in the **%root%\Bank** directory with Windows Notepad so that you can edit it.
Locate line 17 of the VB script, **ComitorRoot = "d:\"**, and modify it to point to your actual root drive letter (e.g. **"c:\"**).
4. Open a Command Prompt window and run **instdll.vbs**.

```
instdll.vbs <enter>
```

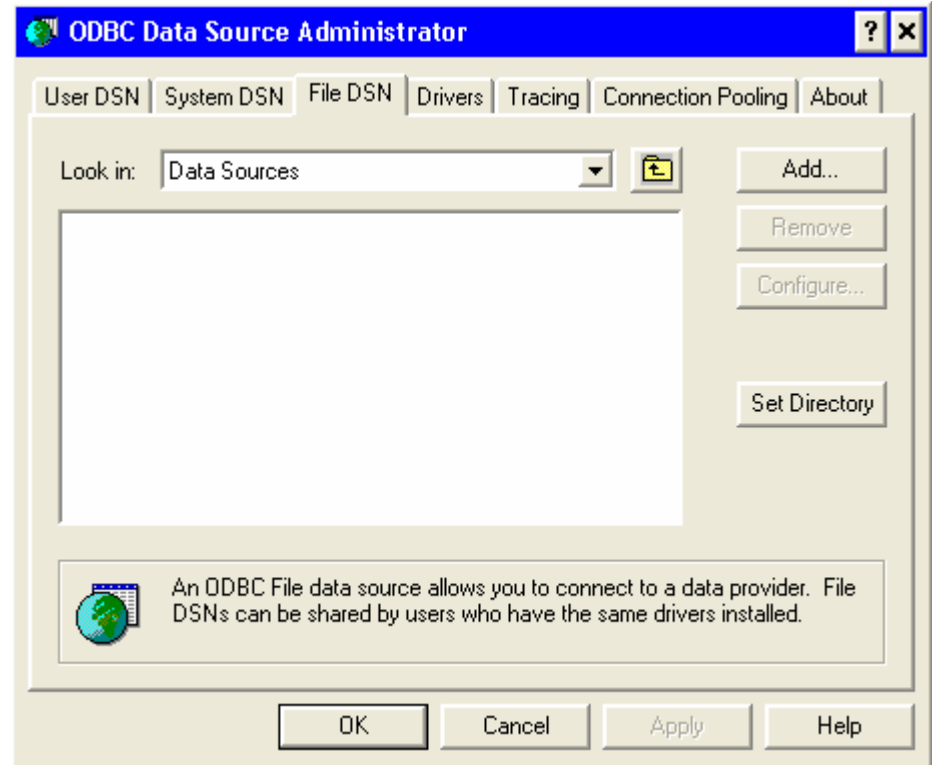


Figure 3

5. Open up your **Data Sources (ODBC)** control panel.

Click on the **File DSN** tab, and then click the **Add** button (see Figure 3).

On the **Create New Data Source** dialog box, choose **SQL Server** as the driver for this data source. Next enter **MTSSamples** as the name of the data source, and click the **Finish** button.

In the **Create a New Data Source to SQL Server** dialog box, enter a description for the data source (optional), and then select **Local** from the **Server** drop-down list.

In the next screen, choose the **With SQL Server authentication...** radio button, and enter **sa** as the **Login ID** and leave the **Password** field blank (see Figure 4).

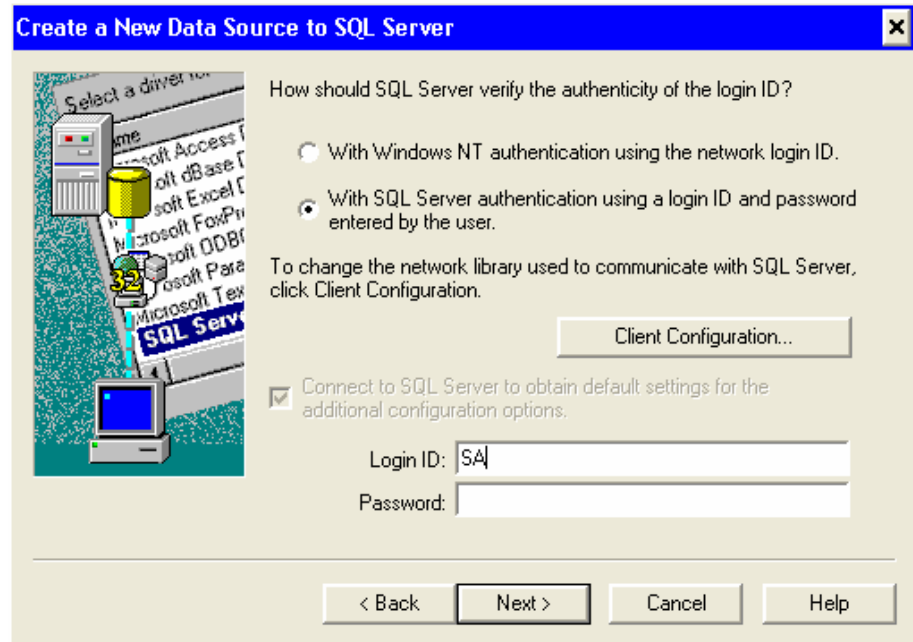


Figure 4

At the next screen, select **pubs** from the **Change the default database** drop-down list. Leave all other options at the defaults (see Figure 5). Click **Next**.

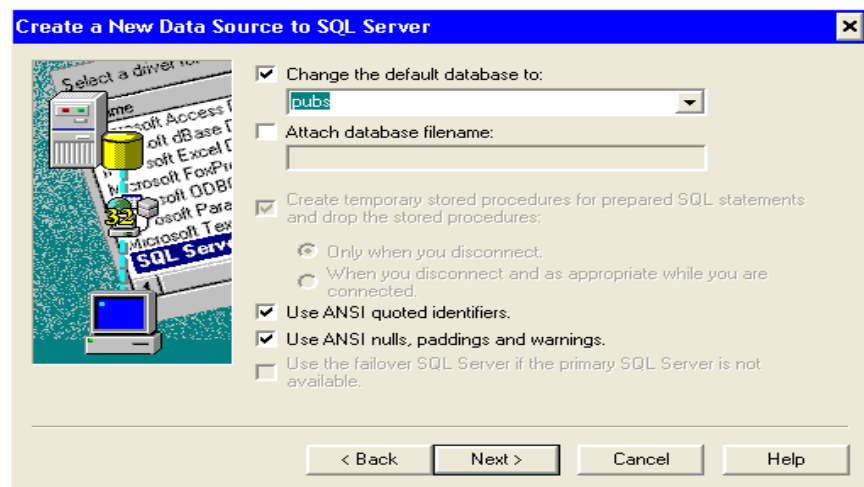


Figure 5

Click the **Finish** button at the next screen.

At the **ODBC Microsoft SQL Server Setup** screen click on the **Test Data Source** button. You should receive the message **'TESTS COMPLETED SUCCESSFULLY'** in the **Test Results** window. If you receive some other message, see your instructor for assistance. Click the **OK** button to exit the ODBC control panel.

6. You can now start the *Sample Bank* application by invoking **bank.exe** from either the Command Prompt or by locating the executable in Windows Explorer and double-clicking on it.

When you do so, you will see the **C Bank** application GUI as it appears below (see Figure 6).

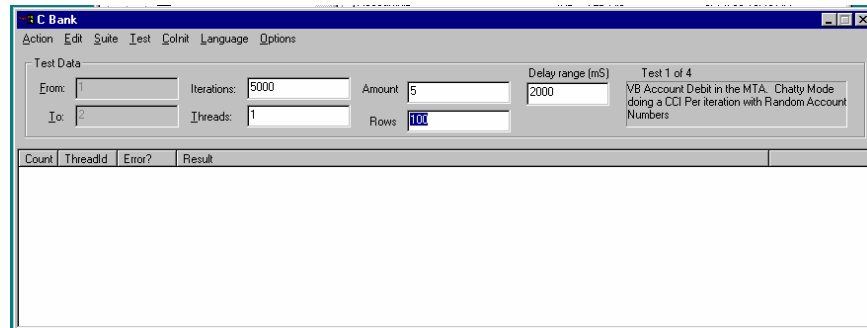


Figure 6

7. Make the following configuration changes:

Ensure that **Options** → **Retry on Errors** is set.

Next create a schema by choosing **Test** → **Create Schema**, and then **Action** → **Go**. The results panel below will state 'Done Creating Schema' when complete.

Change the language to C++ by choosing **Language** → **C ++**.

Type **3** into the **Threads** field.

Now set row count values by choosing **Action** → **Set Row Count**.

To see sufficient transactions we will configure a new action per iteration by selection **Options** → **Create Per Iteration**.

Lastly, change the **Iterations** field to 100,000 (to provide a long period of load).

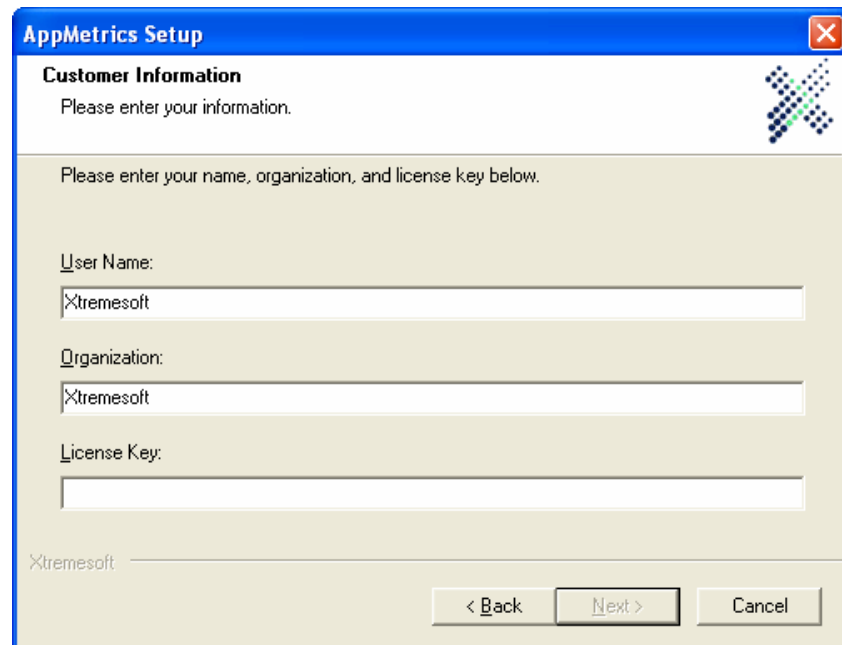
8. It's time to begin generating load.

Select **Test** → **Move Money** → **Credit**, then **Action** → **Go**.

Note: If you see any messages in the results panel that end with an **'HRESULT'** hexadecimal value, your setup of *Sample Bank* has failed for some reason. Ask your instructor for assistance.

Step 2: Install an AppMetrics for Transactions Agent

1. Login to the destination Agent machine with an account that belongs to either the local machine's Administrators group or the domain's Administrators group. Close any other programs currently running on the machine.
2. Place the AppMetrics CD-ROM in the CD-ROM drive, and using Windows Explorer, locate the folder corresponding to your particular setup type (NT 4.0 or Win2K).
3. Double-click on **setup.exe**. Click **OK**. After the initial Xtremesoft splash screen, the **Welcome** screen appears. Click **Next**. The **License Agreement** screen appears.
4. After reading the license agreement, click **Yes**. The **Customer Information** wizard screen appears.
5. Enter your name, company name, and for serial number enter the word **EVAL** (see Figure 7). Click **Next**.



The screenshot shows a Windows-style dialog box titled "AppMetrics Setup". The main heading is "Customer Information" with a sub-instruction "Please enter your information." Below this, a larger instruction reads "Please enter your name, organization, and license key below." There are three text input fields: "User Name:" containing "Xtremesoft", "Organization:" containing "Xtremesoft", and "License Key:" which is empty. At the bottom left is the "Xtremesoft" logo. At the bottom right are three buttons: "< Back", "Next >", and "Cancel".

Figure 7

6. Browse to select the folders in which to install the software and data files, or click **Next** to accept the default locations.
Note: You may receive prompts if the specified folders do not exist on the machine. Click **Yes** to these prompts.
7. Choose the setup type. In this case click on the **Agent** radio button (see Figure 8). Click **Next**.

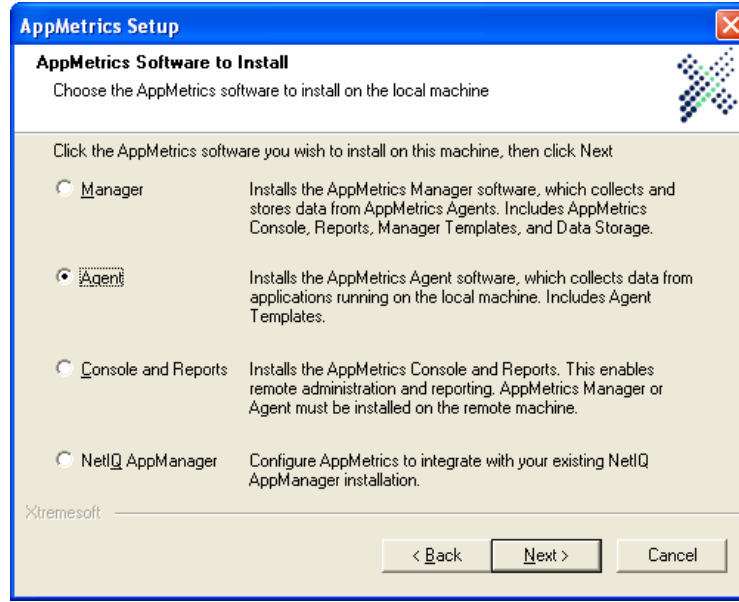


Figure 8

8. Choose the install type **Typical** (recommended for most users). Click **Next**.
9. Specify the Windows user account for AppMetrics (account and password information should be provided by your Instructor - see Figure 9). In Windows NT the AppMetrics Service will run under the context of the Local System and the DCOM protocol will run under the context of the provided user account. In Windows 2000 both the AppMetrics Service as well as DCOM will run under the context of the provided user account.

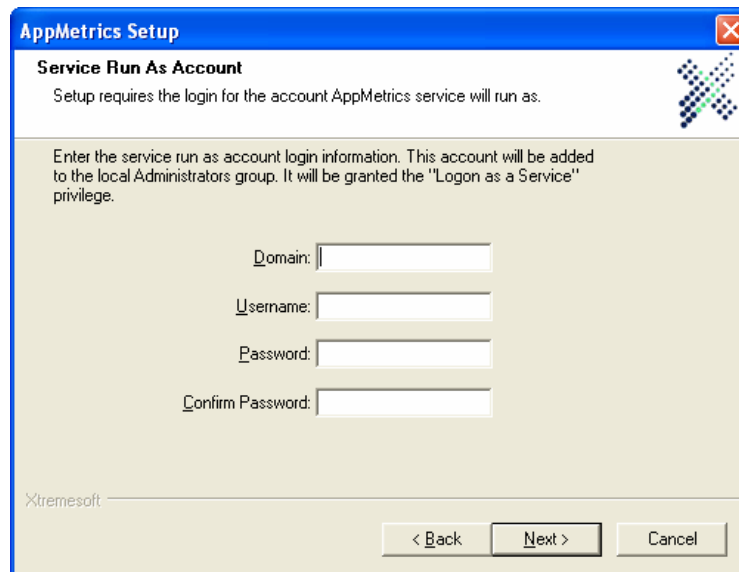


Figure 9

Notes:

- To use a local account for the machine, you must specify the machine name in the **Domain** field.
 - This screen asks for the account password and a confirmation of it in the next field. It verifies that the passwords you enter do indeed match. However, this screen does not validate the account and password on your domain or machine. So if you enter an incorrect username and password combination, the AppMetrics service will use this combination when it tries to start on the machine, and it will fail because the account information is incorrect. You can correct the account information after the install by using the Services administrative tool in Windows.
10. Click **Next**. The install program copies the files to the selected folder location.
 11. The Monitoring Templates wizard screen appears (see Figure 10). Click **Next**. The “**Setup is complete**” screen appears.

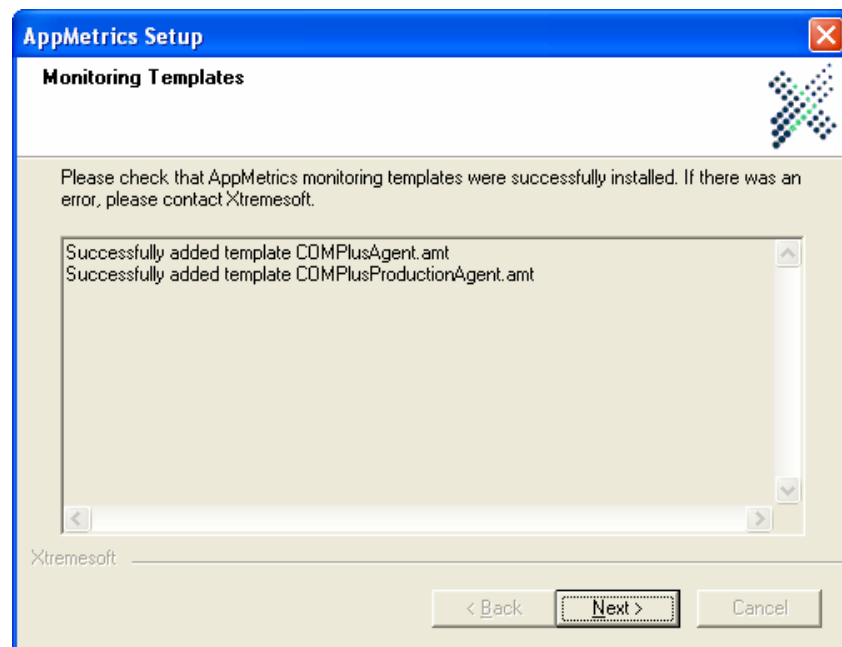


Figure 10

12. Next, choose the **Reboot Now** option for rebooting the server and click **Next** (see Figure 11).

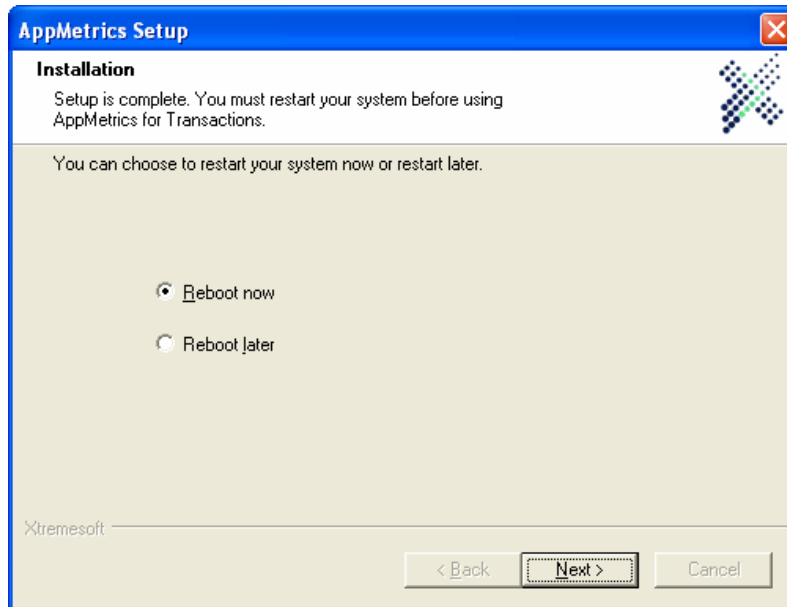


Figure 11

Note: AppMetrics will *not* function unless the system is rebooted. Click **Finish**.

Step 3: Install an AppMetrics for Transactions Manager

1. Login to the destination Manager machine with an account that belongs to either the local machine's Administrators group or the domain's Administrators group. Close any other programs currently running on the machine.
2. Place the AppMetrics CD-ROM in the CD-ROM drive, and using Windows Explorer, locate the folder corresponding to your particular setup type (NT 4.0 or Win2K)
3. Double-click on **setup.exe**. Click **OK**. After the initial Xtremesoft splash screen, the **Welcome** screen appears. Click **Next**. The **License Agreement** screen appears.
4. After reading the license agreement, click **Yes**. The **Customer Information** wizard screen appears.
5. Enter your name, company name, and the serial number you were provided by your Instructor (see Figure 12). Click **Next**.

Figure 12

6. Browse to select the folders in which to install the software and data files, or click **Next** to accept the default locations.

Note: You may receive prompts if the specified folders do not exist on the machine. Click **Yes** to these prompts.
7. Choose the setup type. In this case click on the **Manager** radio button (see Figure 13). Click **Next**.

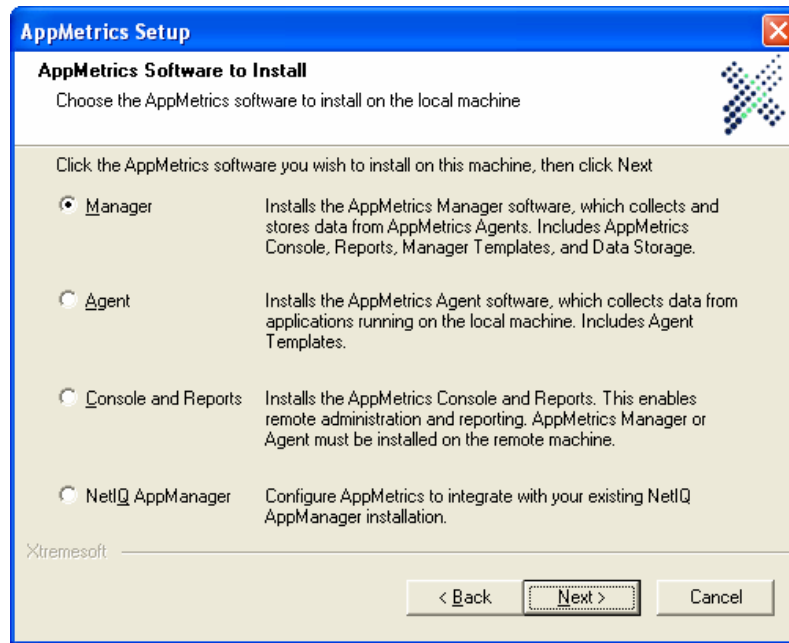


Figure 13

8. Choose the install type **Typical** (recommended for most users). Click **Next**.
9. Specify the Windows user account for AppMetrics (account and password information should be provided by your Instructor). (See Figure 14.) In Windows both the AppMetrics Service as well as DCOM will run under the context of the provided user account.

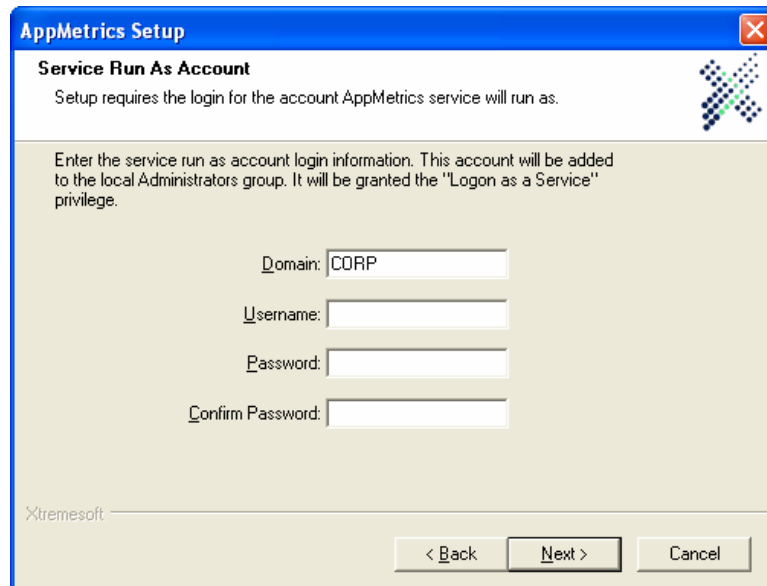
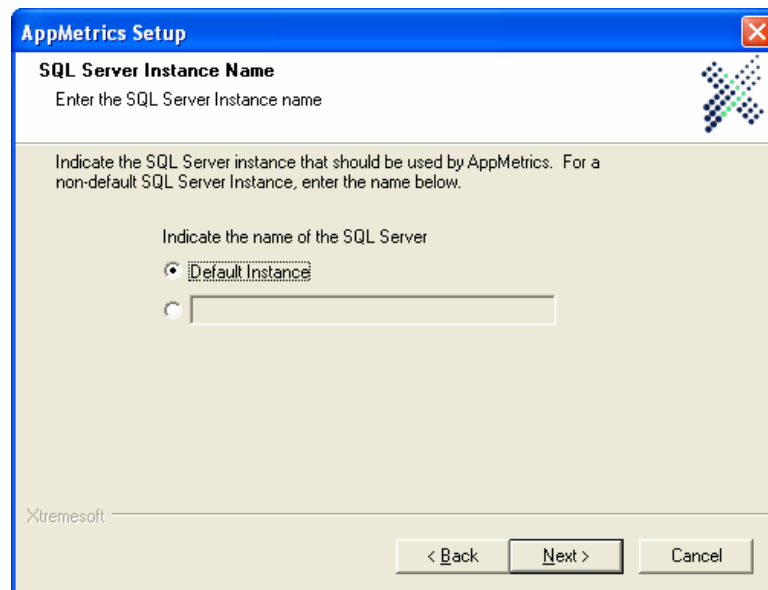


Figure 14

Notes:

- To use a local account for the machine, you must specify the machine name in the **Domain** field.
- This screen asks for the account password and a confirmation of it in the next field. It verifies that the passwords you enter do indeed match. However, this screen does not validate the account and password on your domain or machine. So if you enter an incorrect username and password combination, the AppMetrics service will use this combination when it tries to start on the machine, and it will fail because the account information is incorrect. You can correct the account information after the install by using the Services administrative tool in Windows.

10. AppMetrics Manager installations require a SQL database. The install will prompt you for the name of the SQL Server instance.



11. The install will prompt you for the SQL Server account and password or you can use Windows Authentication. This account has already been pre-configured on your machine.

(see Figure 17). Click **Next**.

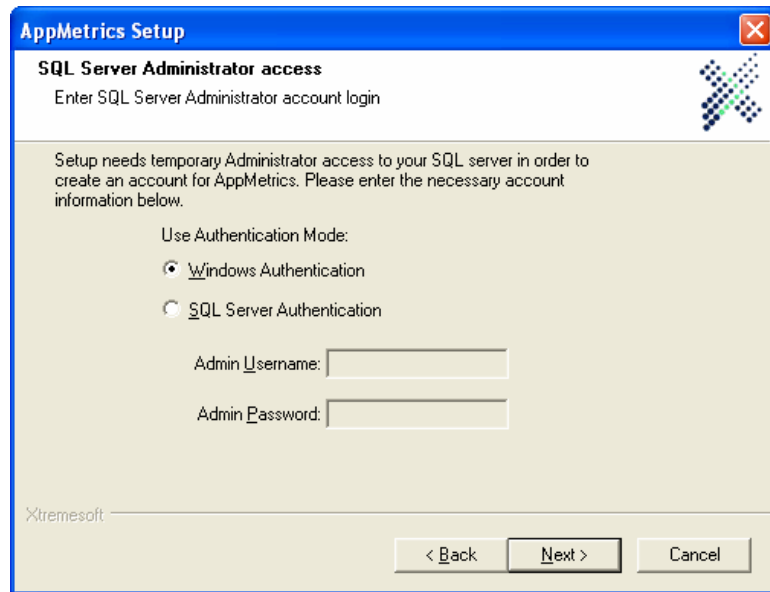


Figure 17

12. If your SQL Server Administrator information is valid, you will then be prompted to provide a password for the AppMetrics account in SQL Server. Leave this field blank (see Figure 18).

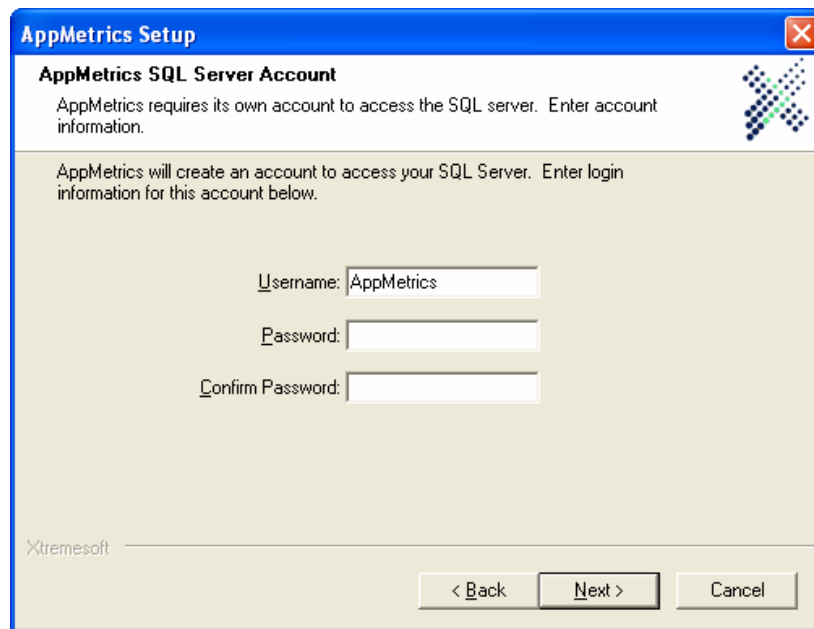


Figure 18

13. Click **Next**. The install program copies the files to the selected folder location.
14. The Monitoring Templates wizard screen appears. Click **Next**. The "Setup is complete" screen appears.

15. Next, choose the **Reboot Now** option for rebooting the server and click **Next**.

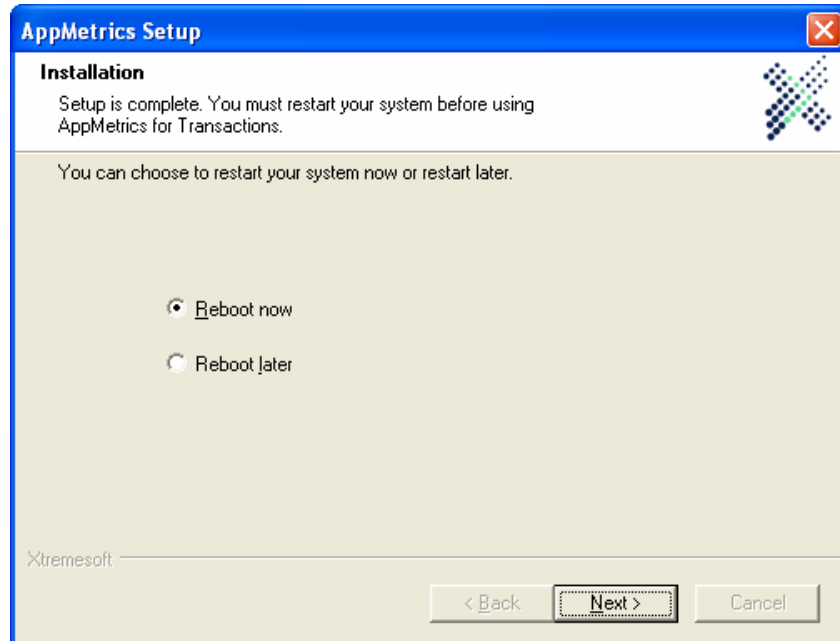
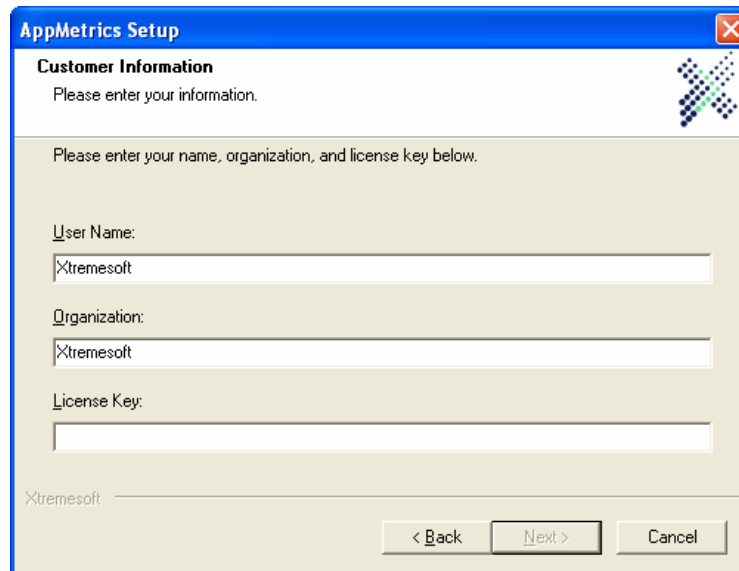


Figure 19

Note: AppMetrics will *not* function unless the system is rebooted. Click **Finish**.

Step 4: Install an AppMetrics for Transactions Console/Reports User

1. Login to the destination **Console/Reports** machine with an account that belongs to either the local machine's Administrators group or the domain's Administrators group. Close any other programs currently running on the machine.
2. Place the AppMetrics CD-ROM in the CD-ROM drive, and using Windows Explorer, locate the folder corresponding to your particular setup type (NT 4.0 or Win2K)
3. Double-click on **setup.exe**. Click **OK**. After the initial Xtremesoft splash screen, the **Welcome** screen appears. Click **Next**. The **License Agreement** screen appears.
4. After reading the license agreement, click **Yes**. The **Customer Information** wizard screen appears.
5. Enter your name, company name, and the serial number you were provided by your Instructor (see Figure 20). Click **Next**.



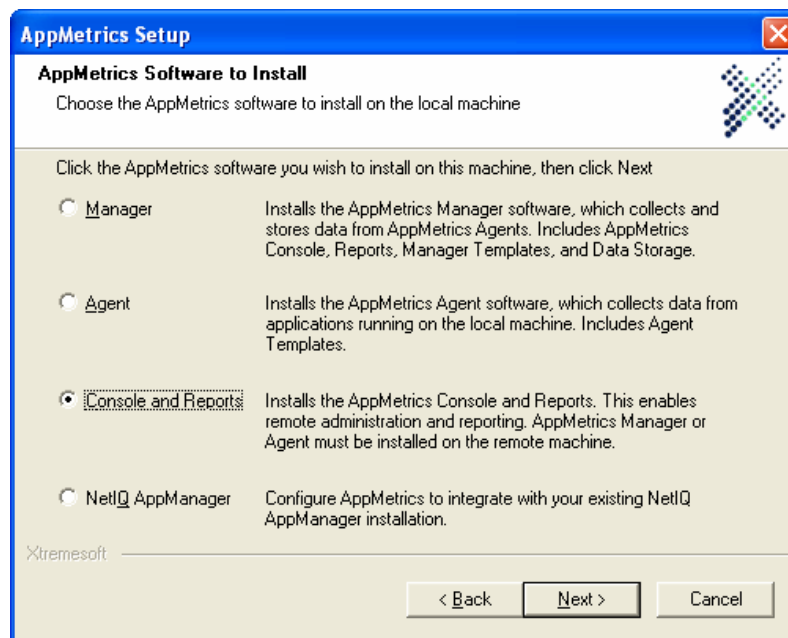
The image shows a Windows-style dialog box titled "AppMetrics Setup". The main heading is "Customer Information" with a sub-instruction "Please enter your information." Below this, there is a larger instruction: "Please enter your name, organization, and license key below." There are three text input fields: "User Name:" containing "Xtremesoft", "Organization:" containing "Xtremesoft", and "License Key:" which is empty. At the bottom left, there is a text box with "Xtremesoft" and a cursor. At the bottom right, there are three buttons: "< Back", "Next >", and "Cancel".

Figure 20

6. Browse to select the folders in which to install the software and data files, or click **Next** to accept the default locations.

Note: You may receive prompts if the specified folders do not exist on the machine. Click **Yes** to these prompts.

7. Choose the setup type. In this case click on the **Reports and Console** radio button (see Figure 21). Click **Next**.



The image shows a Windows-style dialog box titled "AppMetrics Setup". The main heading is "AppMetrics Software to Install" with a sub-instruction "Choose the AppMetrics software to install on the local machine". Below this, there is a larger instruction: "Click the AppMetrics software you wish to install on this machine, then click Next". There are four radio button options, each with a description: "Manager" (installs AppMetrics Manager software), "Agent" (installs AppMetrics Agent software), "Console and Reports" (selected; installs AppMetrics Console and Reports), and "NetIQ AppManager" (configures integration with existing NetIQ AppManager). At the bottom left, there is a text box with "Xtremesoft" and a cursor. At the bottom right, there are three buttons: "< Back", "Next >", and "Cancel".

Figure 21

8. Choose the install type **Typical** (recommended for most users). Click **Next**.

9. Click **Next**. The install program copies the files to the selected folder location.
10. Next, click the **Finish**.



Review Questions

1. Why don't we install SQL Server on the AppMetrics Agent machines?
2. Why doesn't a *Reports and Console* installation request a domain account for the AppMetrics service?

Section 5: Creating Monitors

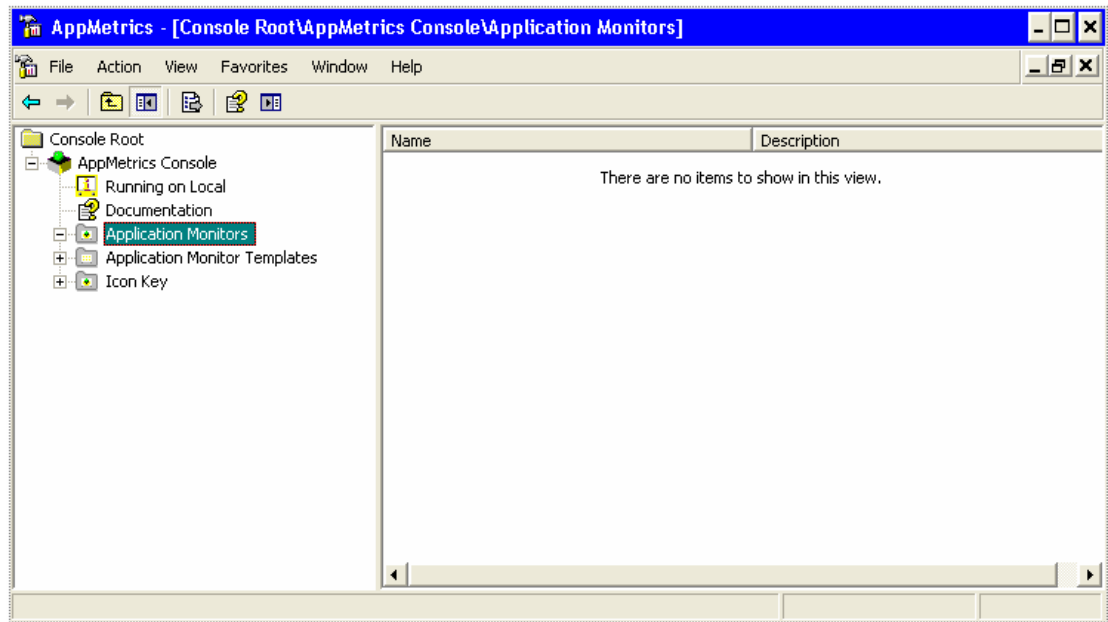
Production v. Diagnostic Monitors

- **Production monitor** – Use this monitor to help you manage your deployed MTS packages and COM+ applications. Production monitors display resource metrics (such as CPU, Memory, Threads, etc) about each package/application. They also display metrics about how different parts of your packages/applications are running at the end of each interval. In particular, you can view metrics about all the transactions, individual transactions, and components in the package/application. Complementing the metrics are the notification options. When the activity level of an item passes a notification threshold, the monitor can create an entry in the Windows Event Log, send an e-mail notification, or set an SNMP trap.
- **Diagnostic monitor** – Use this monitor when you require maximum detail concerning every event within the MTS or COM+ system. If the application under investigation has problems that otherwise defy analysis, the Diagnostics monitor lets you run the package/application and record time-correlated data about it. The Diagnostic monitor can generate large, detailed log files. With the aid of separate analytical software such as a database and a spreadsheet, these data sets enable you to review the state of the system at any point during the monitoring session.

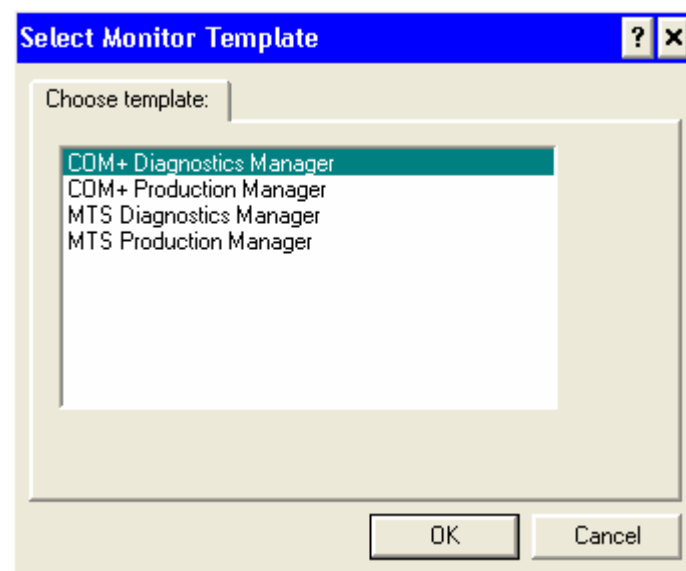
Creating a Production Monitor

You can create a Production Monitor using one of the Production Monitor templates, from any AppMetrics Manager machine or from an AppMetrics Console/Reports client. A Production monitor template is available for MTS, and another is available for COM+. These templates are base-level monitors configured with default settings. You can reconfigure these monitors to perform only those tasks of interest to you.

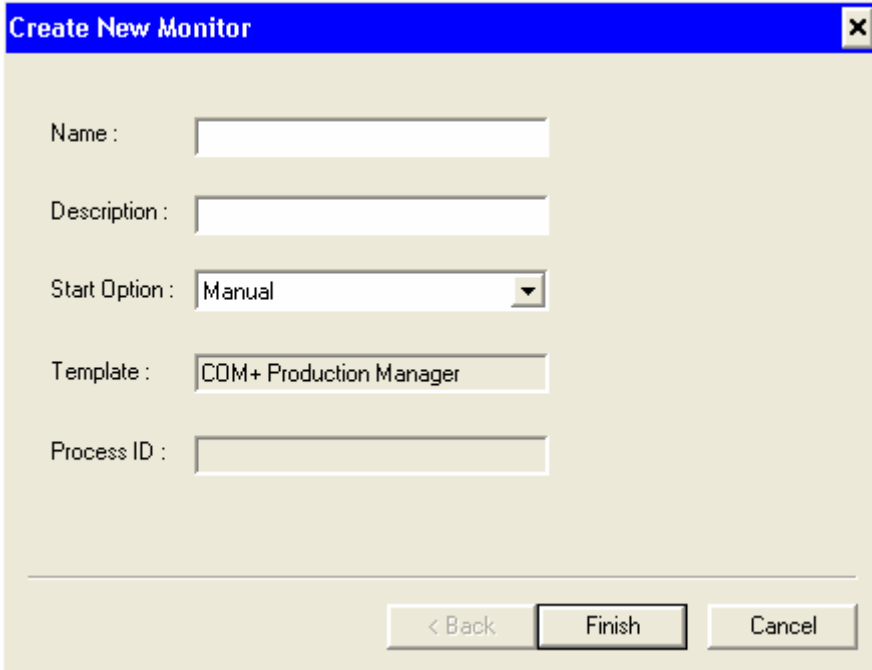
The following illustration shows the Application Monitors node, where you can view your current monitors and create a new Production Monitor based on a template:



The following illustration shows the New Application Monitor dialog, where you can select the template for the new monitor:



When creating a monitor through a template, you must specify a unique name for it. The following illustration shows the General dialog, where you can specify the name and additional information for a new monitor:



The screenshot shows a dialog box titled "Create New Monitor". It contains the following fields and controls:

- Name :** A text input field.
- Description :** A text input field.
- Start Option :** A dropdown menu with "Manual" selected.
- Template :** A text input field containing "COM+ Production Manager".
- Process ID :** A text input field.
- Buttons:** "< Back", "Finish", and "Cancel" are located at the bottom right.

Creating a Diagnostic Monitor

The process of creating a Diagnostics monitor is similar to the one for creating a Production monitor. You select one of the Diagnostics templates to serve as your base-level monitor, and then you name and configure it to suit your requirements.

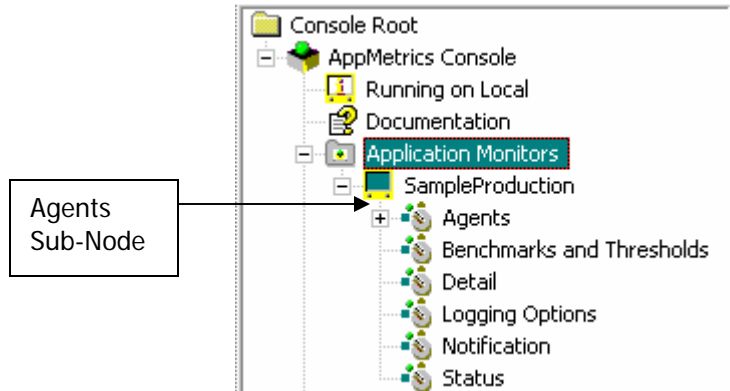
Adding an Agent Monitor

After you create a monitor running on a Manager machine, you must connect it with a monitor running on an Agent machine. The agent monitor collects data about the packages/applications, and it sends the data to the manager monitor. In turn, the manager monitor processes and stores the data.

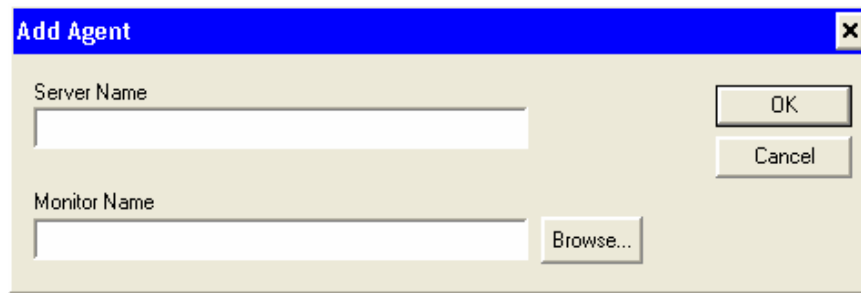
These monitors communicate with each other, performing their unique, specialized roles to ultimately produce metrics about the packages/applications on the Agent machine.

A one-to-one relationship exists between a manager monitor and an agent monitor. This means that a manager monitor collects data from only one agent monitor. It also means that the agent monitor sends its data exclusively to the manager monitor.

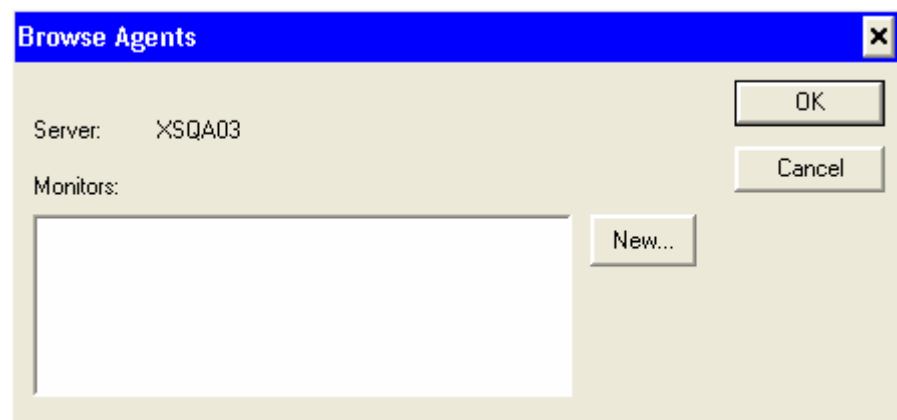
The following illustration shows the Agents node in a manager monitor. The Agent node is where you can access and add an agent monitor:




When you add an agent monitor, you must specify both the Agent machine where it resides and its name. You can do this in the Add Agent dialog:



If you do not know of any agent monitors on the Agent machine, you can use the Browse Agents dialog:



If no monitors are available yet on the Agent machine, you can create a new one in the Create New Agent dialog:



The screenshot shows a dialog box titled "Create New Agent" with a close button (X) in the top right corner. The dialog contains the following fields and controls:

- Server:** A text field containing the value "TRIBBLE1".
- Monitor Name:** An empty text input field.
- Template:** A dropdown menu currently displaying "COM+ Production Agent".
- Description:** An empty text input field.
- Start Option:** A dropdown menu currently displaying "Manual".
- Buttons:** Two buttons are located on the right side: "OK" and "Cancel".

Section 6: Lab 2 - Creating Production & Diagnostic Monitors

Objectives

- Create an AppMetrics Production Monitor on the AppMetrics Manager machine
- Create an AppMetrics Agent on the new Production Monitor
- Create an AppMetrics Diagnostic Monitor on the AppMetrics Manager machine
- Create an AppMetrics Agent on the new Diagnostic Monitor

Prerequisites

- The installation of an AppMetrics Agent on at least one team machine
- The installation and configuration of the Sample Bank application on an AppMetrics Agent machine.
- The installation of an AppMetrics Manager on at least one team machine
- The installation of an AppMetrics Console/Reports client on at least one team machine

Preparation


Students should perform this Lab individually or as a team on an AppMetrics Manager machine console, or on an AppMetrics Console/Reports client console.



Note:

- While technically feasible, an AppMetrics Monitor should *never* be created directly on an AppMetrics Agent. Monitors are always created on an AppMetrics Manager (or via the Console/Reports client) and AppMetrics will automatically handle the creation of Agent components.
- Students performing the lab on an AppMetrics Console/Reports installation should note the special instructions related to this client-type.

Step 1: Creating a Production Manager Monitor

1. If you are running AppMetrics from a Manager machine, your information node (the  icon) will indicate that you are either **'Running on Local'** or **'Running on <computername>'**. You can skip to #2 below.

If you are running AppMetrics from a Console/Reports installation, the first time you start AppMetrics you will get an informational warning that you must specify which AppMetrics Manager you wish to administer (see Figure 1). You should now configure the Console to select your team's AppMetrics Manager.

Do this by selecting the **AppMetrics Console** and choosing the menu option **Action** → **Properties**. This will open the AppMetrics Console Properties dialog box. In the **Server Name** field input the machine name of your team's AppMetrics Manager machine.



Note: Do *not* input the name of an AppMetrics Agent machine in this field. While this is technically possible, you should never manage an AppMetrics Agent machine directly, either from a Console/Reports client or from the Agent machine itself.

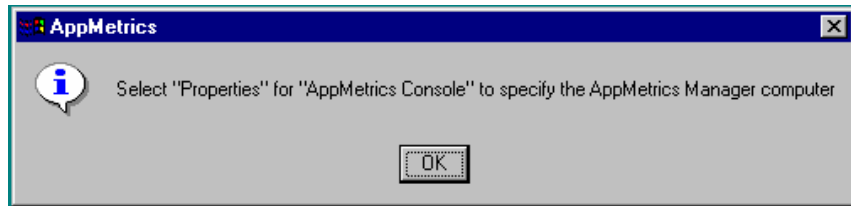


Figure 1

2. Begin creating the new monitor. Select the **Application Monitors** icon and then pull-down **New** → **Application Monitor** from the **Action** menu.

You may also right-click the **Application Monitors** icon and then pull-down **New** → **Application Monitor** (see Figure 2).

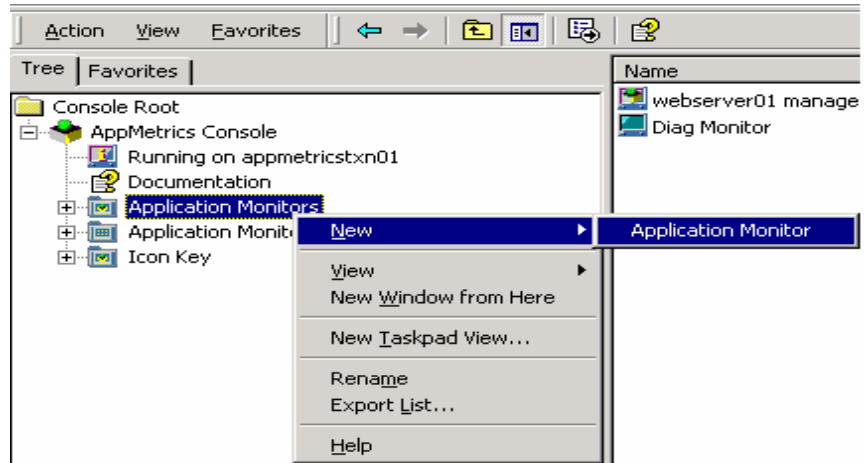


Figure 2

3. Select the appropriate Monitor type template from the pop-up list box. For NT 4 servers you will select an MTS template, and for Windows 2000 servers you should select a COM+ template. See the table below for specific guidance on selecting the correct template.

If the monitored application runs on:	And your chosen monitor type is:	Then use this template for the Manager Monitor:	And AppMetrics will use this template for the Agent Monitor:
Windows NT 4	Production	MTS Production Manager	MTS Production Agent
	Diagnostics	MTS Diagnostics Manager	MTS Diagnostic Agent
Windows 2000	Production	COM+ Production Manager	COM+ Production Agent
	Diagnostics	COM+ Diagnostics Manager	COM+ Diagnostics Agent

In this exercise you should select the **Production Manager** template that matches the server type you have configured in Lab 1: NT4 – MTS, or W2K – COM+ (see Figure 3).

Click **OK**.

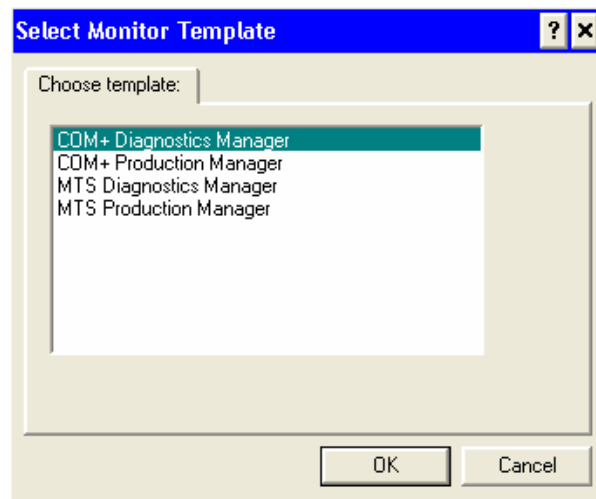


Figure 3

4. At the new monitor dialog you should type in a meaningful name for the Monitor, such as 'Server COM04 Prod Manager'. Monitor names must be alphanumeric and have a maximum length of 30 characters.

Next you may optionally type in a description for the Monitor in the **Description** field.

For the purposes of this Lab select the **Start Option** of **Manual**. In a true production environment you would probably select a start option of Automatic so that every time the Manager restarted the Monitor would also be started.

Click the **Finish** button to complete the creation of this Monitor (see Figure 4).

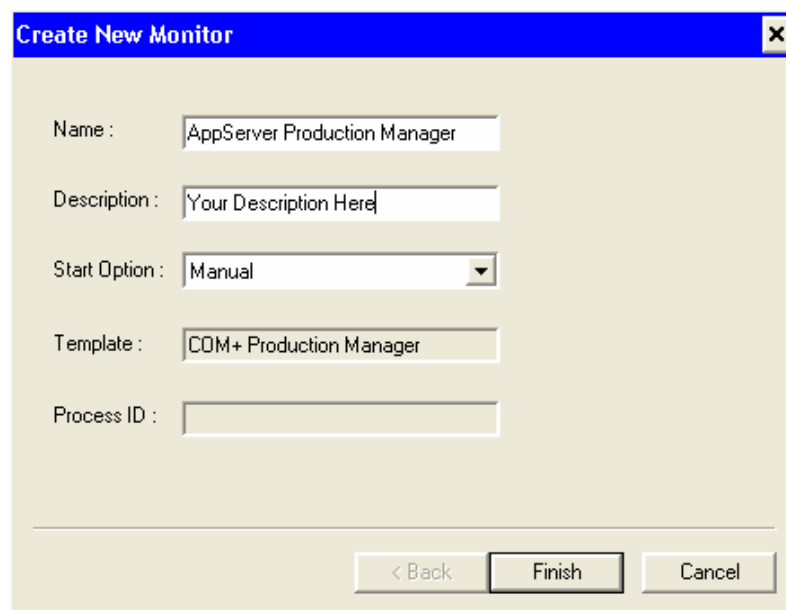


Figure 4

5. A Manager Monitor will not receive any data to work upon until you create an Agent within it.



Note: In the AppMetrics model there is a **Monitor** on both the Agent and Manager machine. When we refer to the **Manager Monitor** we are referring to the monitor software running on the Manager machine. It is this object that we generally refer to when we say 'Monitor'. However there is also monitor software running on the Agent machine, and this is specifically referred to as the **Agent Monitor**. The Agent Monitor is always administered via the Manager and we generally refer to it simply as the 'Agent'.

Expand the Manager Monitor and select the **Agents** icon, then pull-down **New** → **Agent** from the **Action** menu (see Figure 5).

You may also right-click the **Agents** icon and then pull-down **New** → **Agent**.

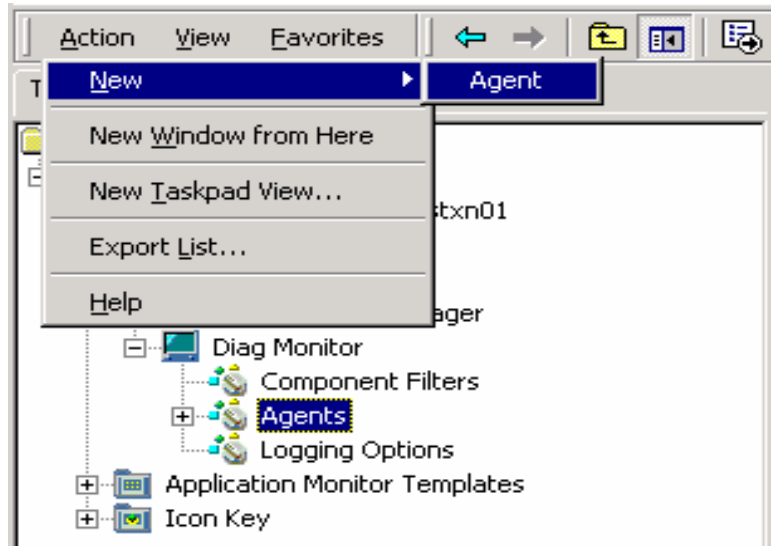


Figure 5

6. At the add agent dialog box **Server Name** field you should type in the computer name of the AppMetrics Agent machine in which you wish to create an Agent Monitor (see Figure 6).

Click the **Browse** button to the right of the **Monitor Name** field.

Click the **New** button on the browse agents dialog box.

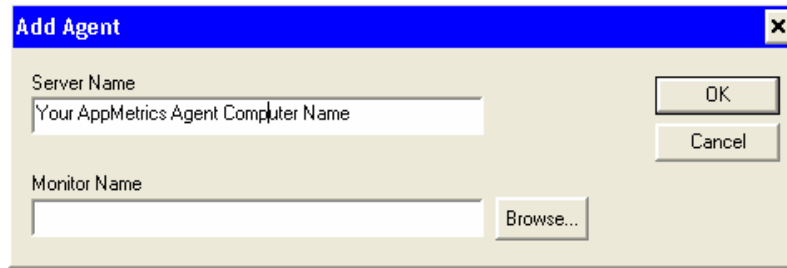


Figure 6

- At the create new agent dialog box, type in a meaningful Agent Monitor name in the **Monitor Name** field, such as 'Server COM04 Prod Agent'. Agent Monitor names must be alphanumeric.

Next you may optionally type in a description for the Agent Monitor in the **Description** field.

For the purposes of this Lab select the **Start Option** of **Manual**. In a true production environment you would probably select a start option of Automatic so that every time the Agent machine restarted the Agent Monitor would also be started.

Click the **OK** button to complete the addition of this Agent Monitor (see Figure 7).

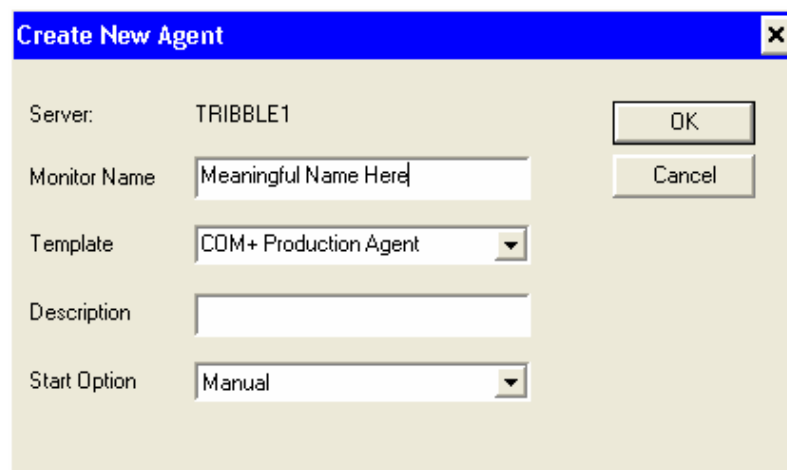


Figure 7

Click **OK** on the add agent dialog box.



Note: Do *not* attempt to start your new Manager Monitor! Before starting a Manager Monitor you should complete all configuration steps. Some configuration defaults will be inadvertently latched-in if you attempt to start your new monitor before configuration is completed.

Step 2: Creating a Diagnostics Manager Monitor

1. Begin creating the new monitor. Select the **Application Monitors** icon and then pull-down **New** → **Application Monitor** from the **Action** menu.

You may also right-click the **Application Monitors** icon and then pull-down **New** → **Application Monitor** (see Figure 8).

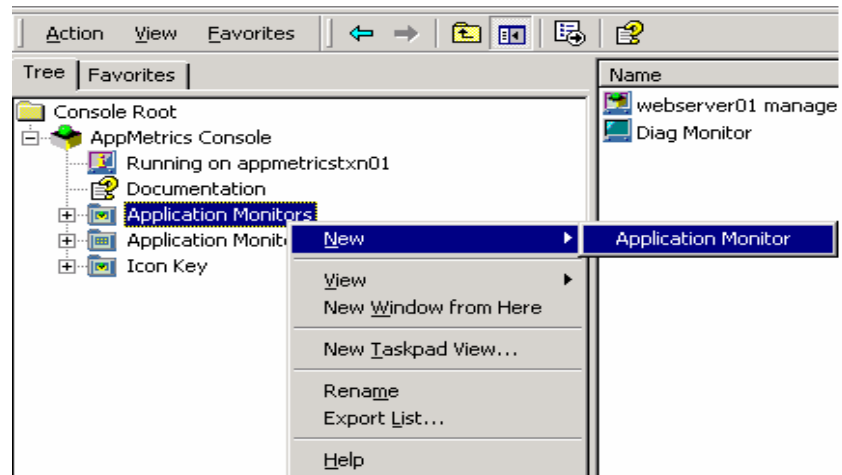


Figure 8

2. Select the appropriate Monitor type template from the pop-up list box. If the Agent machine runs NT 4, then you must use an MTS Manager monitor on the manager machine. On the other hand, if the Agent machine runs Windows 2000, you must use a COM+ Manager monitor on the manager machine.

In this exercise you should select the **Diagnostics Manager** template that matches the server type you have configured in Lab 1: NT4 – MTS, or W2K – COM+ (see Figure 9).

Click **OK**.

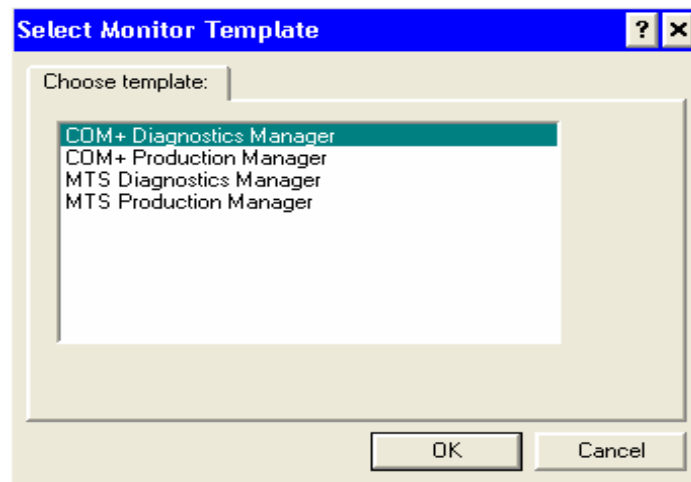


Figure 9

3. At the new monitor dialog box you should type in a meaningful name for the Monitor, such as 'Server COM04 Diag Manager'. Monitor names must be alphanumeric and have a maximum length of 30 characters.

Next you may optionally type in a description for the Monitor in the **Description** field.

For the purposes of this Lab select the **Start Option** of **Manual**. In a true production environment you would probably select a start option of Automatic so that every time the Manager restarted the Monitor would also be started.

Click the **Finish** button to complete the creation of this Monitor (see Figure 10).

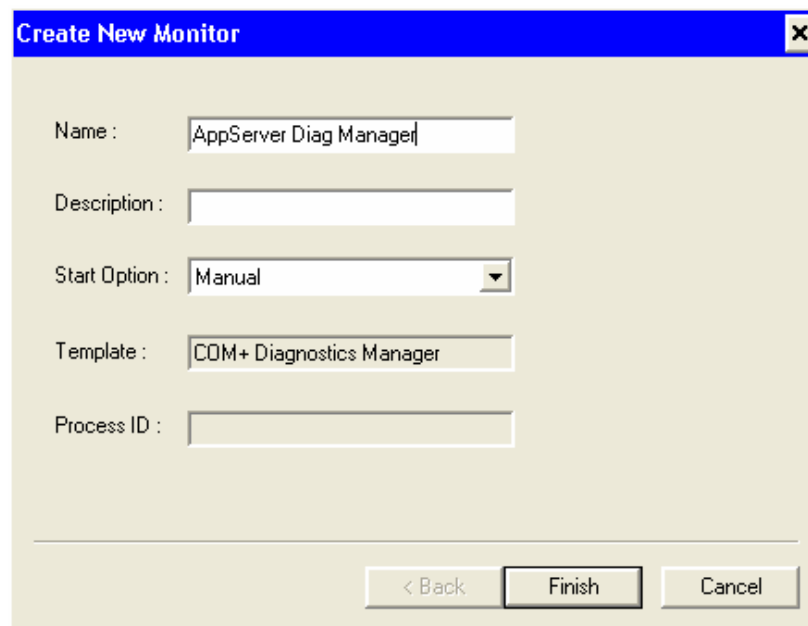


Figure 10

5. A Manager Monitor will not receive any data to work upon until you create an Agent within it.

Expand the Manager Monitor and select the **Agents** icon, then pull-down **New** → **Agent** from the **Action** menu (see Figure 11).

You may also right-click the **Agents** icon and then pull-down **New** → **Agent**.

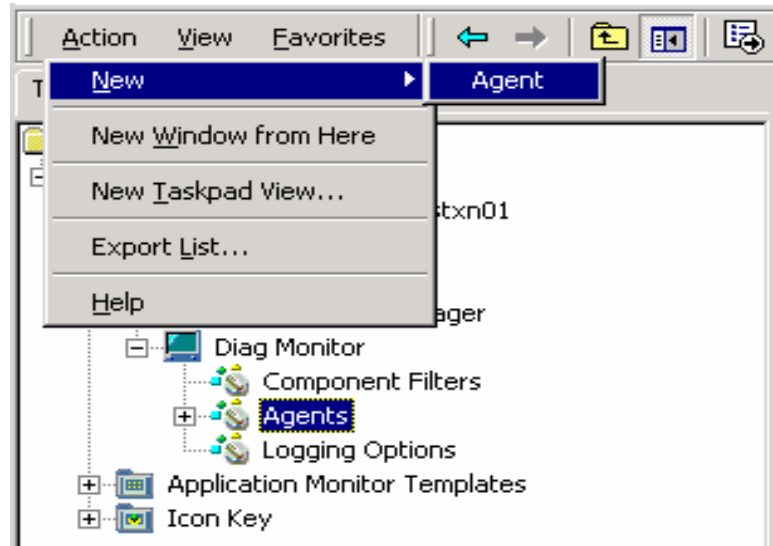


Figure 11

6. At the add agent dialog box **Server Name** field you should type in the computer name of the AppMetrics Agent machine in which you wish to create an Agent Monitor (see Figure 12).

Click the **Browse** button to the right of the **Monitor Name** field.

Click the **New** button on the browse agents dialog box.

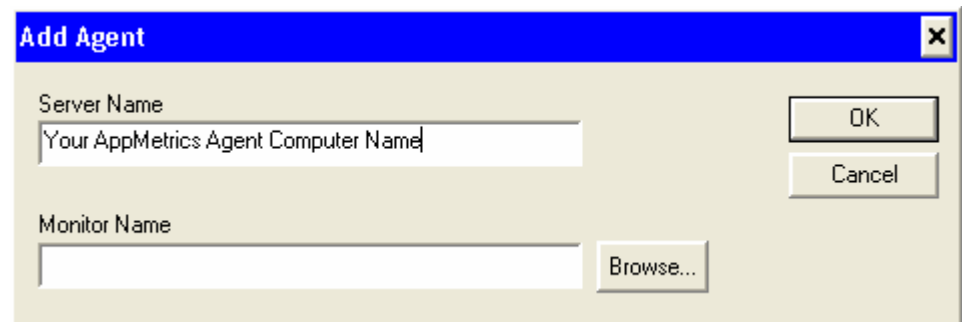


Figure 12

7. At the create new agent dialog box, type in a meaningful Agent Monitor name in the **Monitor Name** field, such as 'Server COM04 Diag Agent'. Agent Monitor names must be alphanumeric.

Next you may optionally type in a description for the Agent Monitor in the **Description** field.

For the purposes of this Lab select the **Start Option** of **Manual**. In a true production environment you would probably select a start option of Automatic so that every time the Agent machine restarted the Agent Monitor would also be started.

Click the **OK** button to complete the addition of this Agent Monitor (see Figure 13).

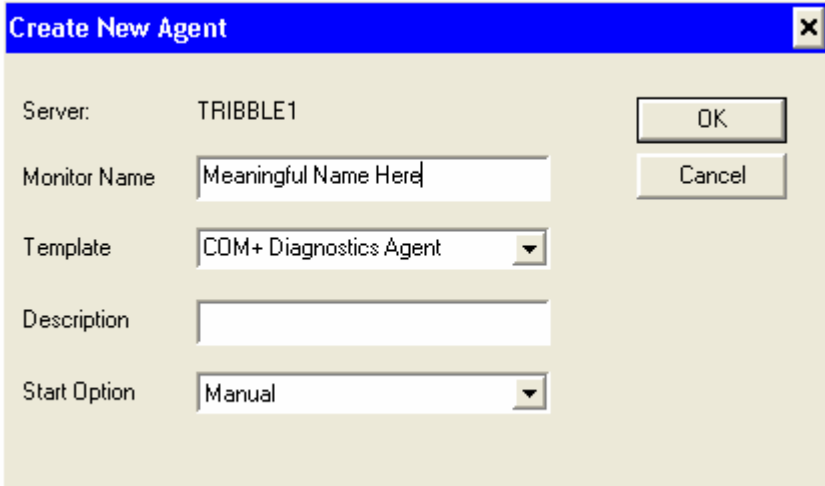


Figure 13

Click **OK** on the add agent dialog box.



Note: Do *not* attempt to start your new Manager Monitor! Before starting a Manager Monitor you should complete all configuration steps. Some configuration defaults will be inadvertently latched-in if you attempt to start your new monitor before configuration is completed.



Review Questions

1. Which type of Monitor would you use in routine analysis of a particular transaction?
2. Which type of Monitor is more suited for long-term trend analysis?
3. Which type of Monitor generates a more detailed set of data?
4. Which type of Monitor averages data over a particular interval of time?

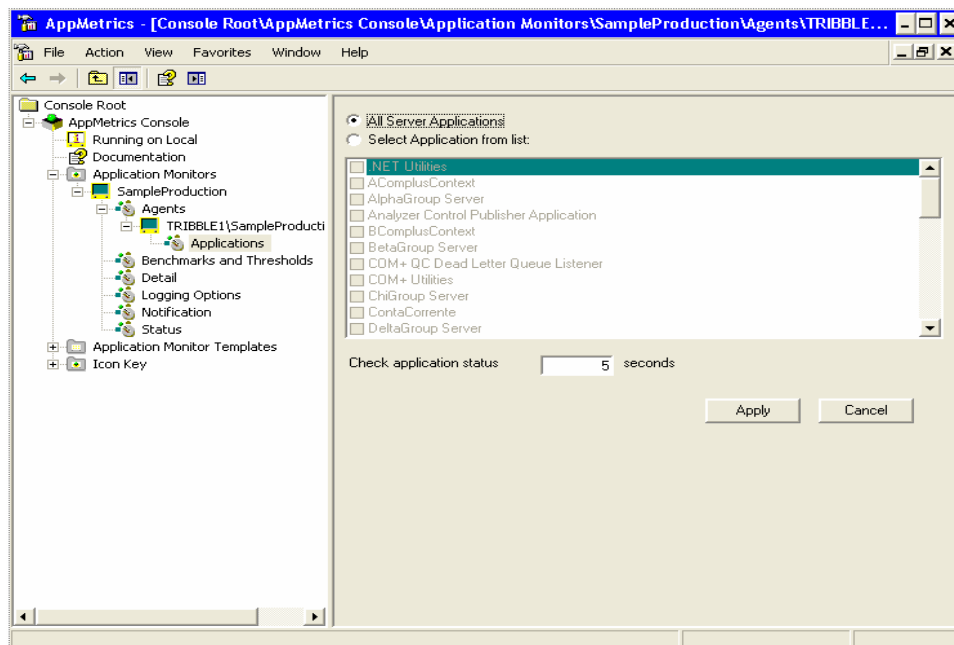
Section 7: Configuring Monitors



Selecting Packages/Applications

You can configure the monitor to collect data about one or more packages/applications on the machine. By default, the Production monitor is set to monitor all packages/applications. You can change the monitor to focus only on those packages/applications of interest to you.

The following illustration shows the Packages/Applications configuration panel, where you can select the packages or applications to be monitored:



Set Intervals

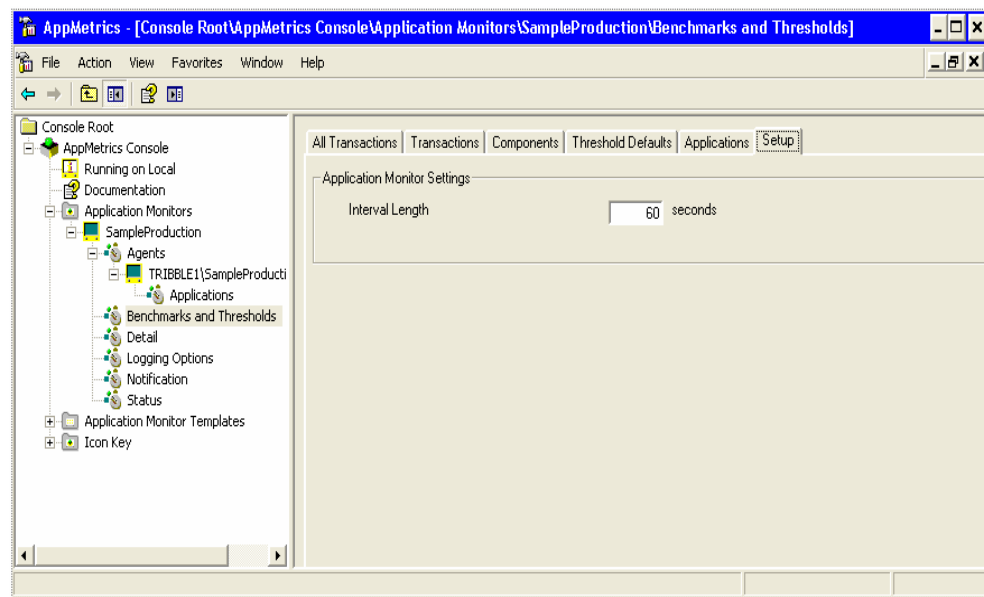
The *interval length* is the time span between calculations of the All Transactions, Transactions, and Components metrics. During an interval, AppMetrics collects data about the activity throughout the monitored packages/applications. At the end of the interval, AppMetrics processes the collected data to calculate totals, averages, and rates for the activity that took place during the interval.

The interval length chosen controls the

- frequency of the updates in the All Transactions, Transactions, and Components runtime panels
- granularity of the data in the All Transactions, Transactions, and Components reports
- timing in which detail throttling is enabled or disabled (COM+ monitors only)
- timing for repeat deliveries of warnings and notifications

Note: This setting differs from the Check Application Status setting in the Applications configuration panel

The following illustration shows the Setup panel, where you can specify the interval length of the monitor.



The *interval length* chosen has important consequences for the later viewing and analysis of data in the AppMetrics Reports.

An interval of 60 seconds, for instance, means that AppMetrics will collect data to calculate totals, averages, and rates for the activity that took place during that 60-second interval. AppMetrics will then write those calculations to its log files with an identifying interval number. An

AppMetrics log file will therefore have an entry for every interval that occurred (in this case every 60 seconds).

When viewing the AppMetrics Reports for this Monitor you will see a data point for every 60 seconds of monitored run-time. The longer the interval chosen in the Monitor, the fewer the number of data points logged, and the fewer the number of data points in the reports. The shorter the interval chosen, the more data points logged and plotted in the reports.

While it might seem at first blush that more data points is better than less, the interval length chosen should be decided based on the following factors:

- **Database Sizing** – the more the data points logged, the larger the database. If it is important to have a relatively fine-grained analysis (more data points), you may have to adjust the data retention period in the logging options of your Monitor to prevent the database from growing too large. If you are using MSDE as your database you are limited to databases no larger than 2GB, and must therefore be careful with the interval length and retention period chosen. Note: your Xtremesoft Support contact can provide you with a Sizing Guidelines document to assisting you in calculating the appropriate size for your database.
- **Package Activity** – choose an interval that provides the best abstraction of your data. For instance, if your transactions and/or components are relatively long-lived, you should choose a longer interval length so that transactions and component instances monitored are more likely to occur within a single interval.
- **Monitoring Objective**– choose an interval that most closely matches your objective in monitoring that application or applications. For instance, if you want to discover approximately when transactions and/or components are logging exceptions, choosing a shorter interval length will more tightly focus your analysis. If your objective is to monitor package CPU or memory utilization, a larger interval may provide a better point of reference. If your objective is to provide notification via AppMetrics of applications that exceed thresholds, you may select a shorter interval to provide more timely notification.

Choosing the right interval length may require some experimentation as you work out the trade offs between database size, data abstraction, and performance monitoring objectives.

Set Detail Levels



User Transaction detail levels determine the amount of data to collect about your monitored packages/applications. You can control the levels of monitoring for your packages/applications by selecting which aspects of user transactions to monitor.

So before discussing the different levels of monitoring, we should review the concept of User Transactions as defined in AppMetrics.

User Transactions - in AppMetrics, a user transaction represents a discrete business activity, such as a deposit made to a bank account or a purchase made on a Web site. A user transaction starts when a client process or a non-monitored package/application makes a call into a component. This component must be monitored either on its own or by virtue of belonging to a monitored package/application. The MTS or COM+ system will then create an instance of the called component, or it will use an existing instance of the component. This instance becomes the *root object* for the user transaction, and it will perform the work of completing the user transaction. While the root object is active, it may make further calls into other components. However, when these latter calls return, the user transaction is not yet complete. The user transaction is complete only when the initial call into the root object returns to the calling process.

Note: User transactions are not identical to DTC transactions in MTS/COM+. A user transaction may or may not be a formal DTC transaction.

As an illustration, consider a transaction for transferring money between bank accounts. The diagram below shows an example of such a

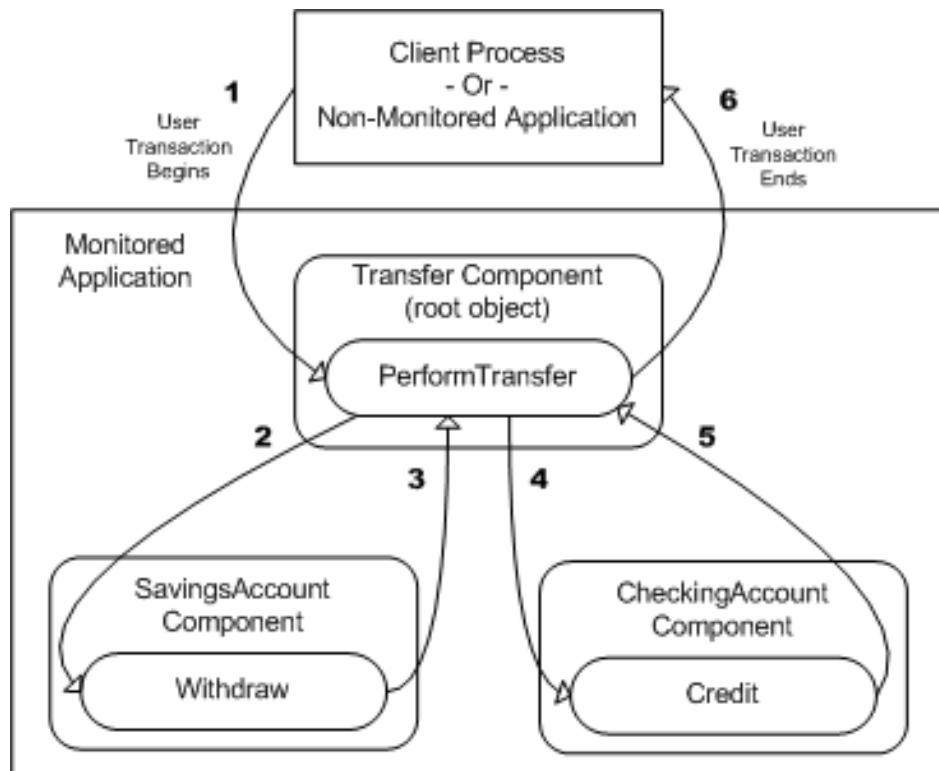
transaction. In the example, the monitored application has a component to perform the transaction. The component is named "Transfer" with a method named "PerformTransfer."

In **Step 1**, the user transaction starts when the client process or non-monitored application calls the PerformTransfer method on the Transfer component. This call results in an instance of the Transfer component. The instance serves as the root object for the user transaction. When performing the user transaction, the root object calls other components to perform additional tasks for completing the transaction.

In **Step 2**, the PerformTransfer method of the root object calls "Withdraw" on the "SavingsAccount" component. This call is required to obtain the funds for the transfer.

In **Step 3**, Withdraw returns to PerformTransfer, but the user transaction is not yet complete.

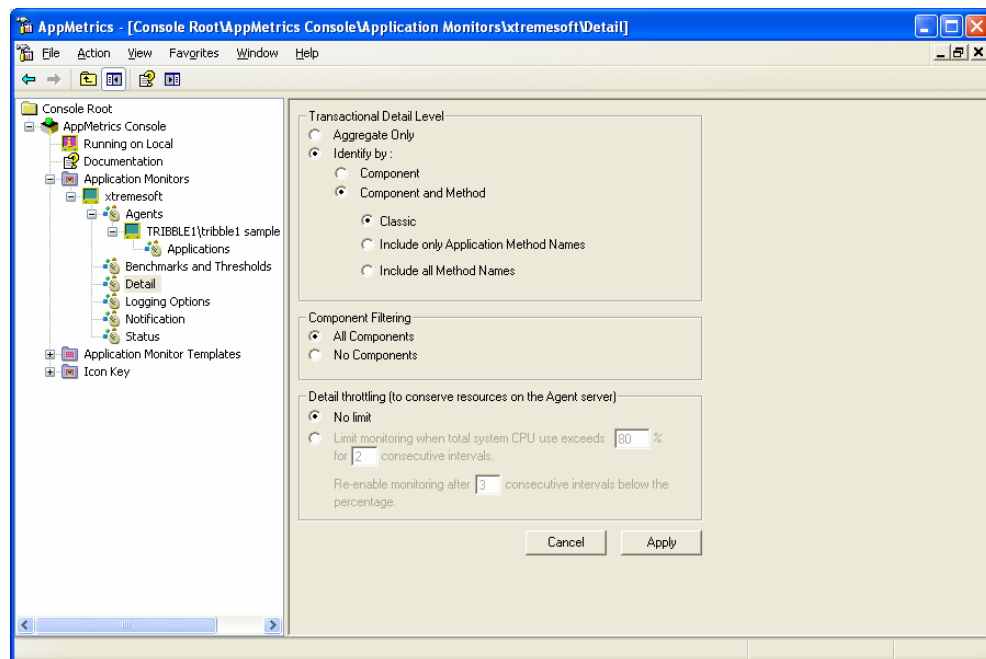
In **Step 4**, the root object calls "Credit" on the CheckingAccount component. This call is required to deposit the funds from the withdrawal into the checking account. When the call to the Credit method returns, as in **Step 5**, the user transaction is still not yet complete. the user transaction will be complete only when the initial call to the PerformTransfer method on the root object returns to the client process or non-monitored application, as in **Step 6**.



User Transaction detail levels - determine the amount of data to collect about your monitored packages/applications.

- The **transactional detail level** controls data collection about individual User Transactions occurring within your packages/applications. If you choose to collect data about individual transactions, you can further control how these transactions are identified: Either by the name of the called component in the transaction or by the name of the called method in the transaction.
- The **component filtering level** controls data collection for all the components called within your monitored packages/applications.

The following illustration shows the Detail configuration panel, where you can choose the settings for the different detail levels:



There are three **transactional detail levels** to choose from:

- **Aggregate Only** - All transactions will be monitored, but details about their components and methods will be ignored. As a result, the Transactions panels and reports will display no data. This detail level is used primarily for package process analysis.
- **Identify by Component** - All transactions will be monitored, and each individual transaction will be specified by its first component. **Note:** If an ASP called the component, AppMetrics will neither collect nor show information about the ASP's URL. This detail level is used primarily for transaction analysis where drill-down into component and/or method data is not necessary.

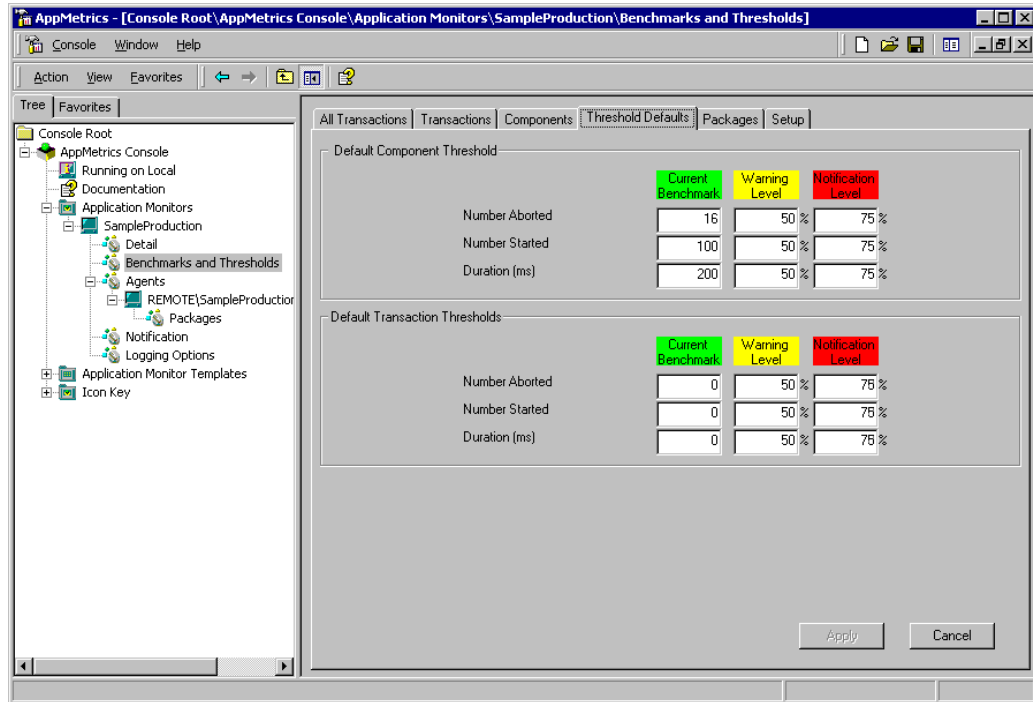
- **Identify by Component and Method** - All transactions will be monitored, and each individual transaction will be specified by the combination of its first component and first method invoked by that component. **Note:** If an ASP page called the component and method, AppMetrics will detect the URL of the ASP page (on COM+ Agent installations only). This detail level is used primarily for transaction analysis where drill-down into component and/or method data is required.

The following table provides additional detail about the levels of monitoring and their configuration settings:

Level of Monitoring	Setting to Enable this Level of Monitoring	Affects the Following Items:	Description
System Resource Usage	None. Always monitored.	<ul style="list-style-type: none"> • Packages/Applications runtime panel • Packages/Applications Report 	This is the usage of resources on a machine. It includes CPU% usage, memory usage, page faults, etc.
All Transactions	None. Always monitored except during a throttle down.	<ul style="list-style-type: none"> • All Transactions runtime and configuration panel • All Transactions Report 	This is the number of user transactions that occurred during the interval.
Aggregate Only	Transactional Detail level area – Aggregate only setting.	<ul style="list-style-type: none"> • Transactions runtime and configuration panel • Transactions Report 	This stops monitoring for individual user transactions. It prevents data from appearing in the Transactions panel and report.
Transactions Identified by Component	Transactional Detail level area – Identify by Component setting.	<ul style="list-style-type: none"> • Transactions runtime and configuration panel • Transactions Report 	This identifies the root component involved in each user transaction. At this level of detail, an AppMetrics User Transaction corresponds to the lifecycle of an MTS/COM+ component instance (from object creation until it goes out of scope).
Transactions Identified by Method	Transactional Detail level area – Identify by Component and Method setting.	<ul style="list-style-type: none"> • Transactions runtime and configuration panel • Transactions Report 	This identifies the root component and the called method involved in each user transaction. At this level of detail, an AppMetrics User Transaction corresponds to the lifecycle of a root method invoked on an MTS/COM+ component. The method may delegate calls to additional methods (sometimes on additional components), but the Transaction corresponds to the root method invoked.

Set Default Thresholds

When components and transactions are added to a monitor, AppMetrics does not apply default values to their thresholds settings. You can specify your own default values in the Threshold Defaults configuration panel.



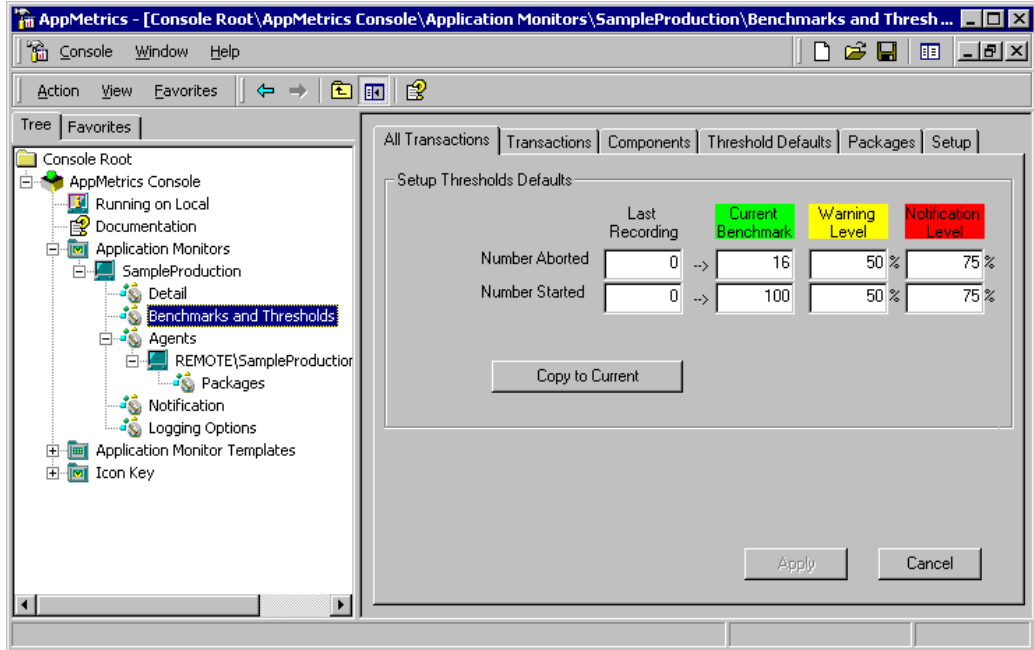
Set Specific Thresholds

The monitor lets you set warning and notification thresholds on individual transactions and components. It also lets you set thresholds for all the transactions occurring on the machine.

Set All Transactions Thresholds

In the monitor, the All Transactions configuration panel lets you specify the warning and notification thresholds for the total number of Aborted and Started transactions that took place during an interval.

The following illustration shows the panel where you can specify the threshold settings for All Transactions.



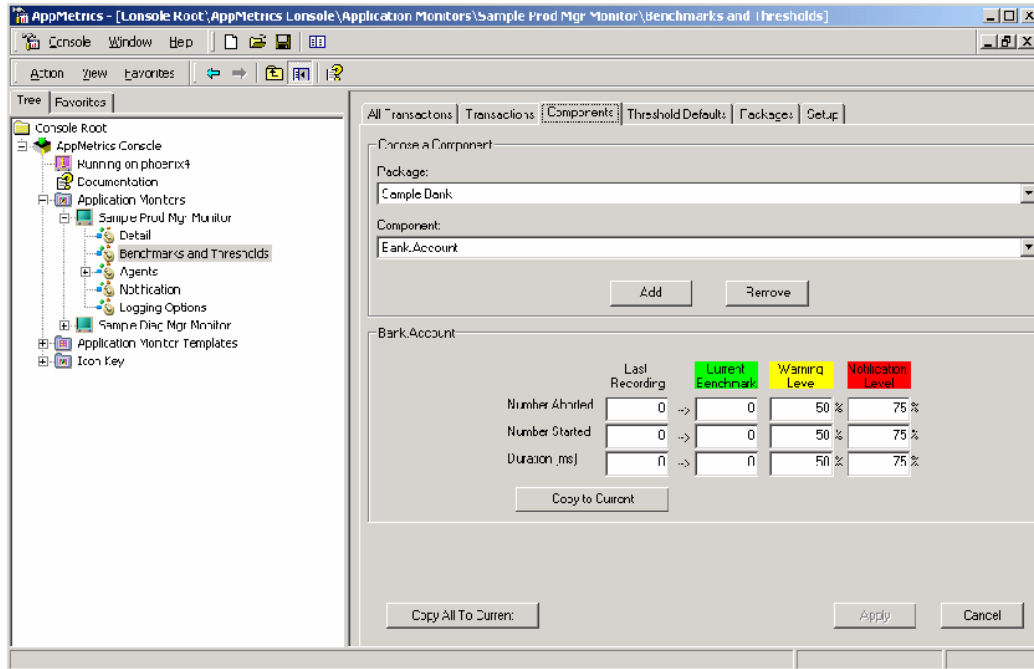
You can set warning and notification thresholds on the following All Transactions metrics:

Metric for All Transactions	Description
Number Aborted	Set thresholds to warn or notify when the system experiences transaction failures more than a specific number of times during an interval.
Number Started	Set thresholds to warn or notify when the system starts transactions more than a specific number of times during an interval. This lets you know when your system is handling more activity than normal.

Set Component Thresholds

During each interval, the Production monitor produces metrics about individual active components. You can apply thresholds to each component. This enables the monitor to warn or notify you about abnormal levels of activity for the component.

The following illustration shows the Components configuration panel, where you can set warning and notification thresholds for individual components:



For each individual component, you can configure warning and notification thresholds on the following metrics:

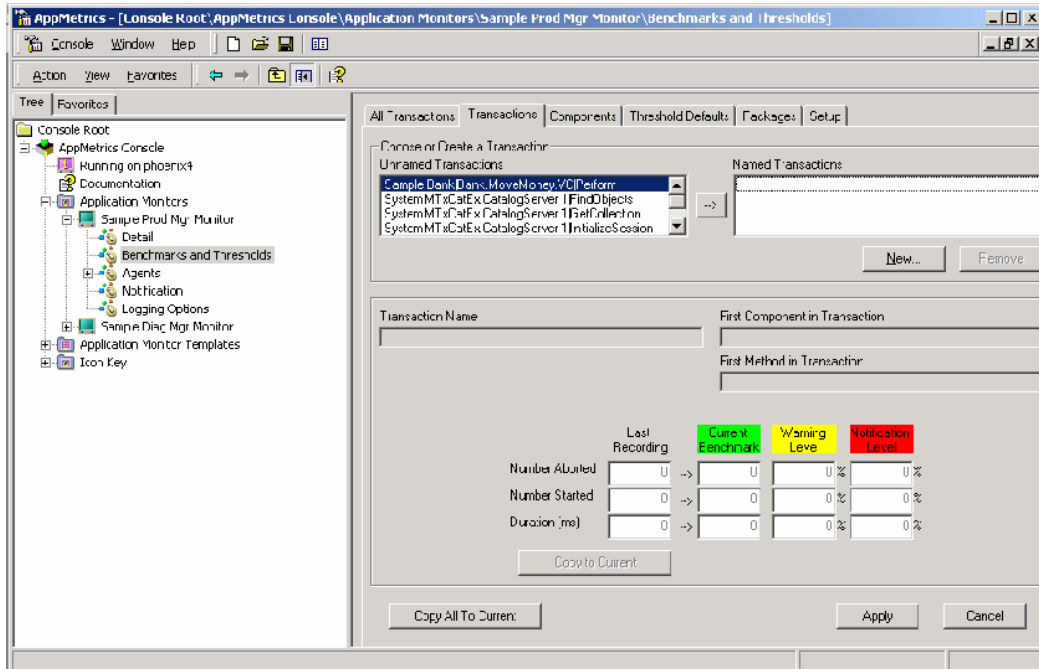
Metrics for a Component	Description
Number Aborted	Set thresholds to warn or notify when the component fails more than a specific number of times during an interval.
Number Started	Set thresholds to warn or notify when the component is started more than a specific number of times during an interval.
Duration (ms)	Set thresholds to warn or notify when the average duration of the component lasts longer than a specific number of milliseconds during an interval.

Set Transactions Thresholds

The Production monitor is able to produce metrics about individual transactions that were active during each interval. You can apply

thresholds to some of the metrics for each transaction. This enables the monitor to warn or notify you about abnormal levels of activity for the transaction.

The following illustration shows the Transactions configuration panel, where you can set warning and notification thresholds for individual transactions.

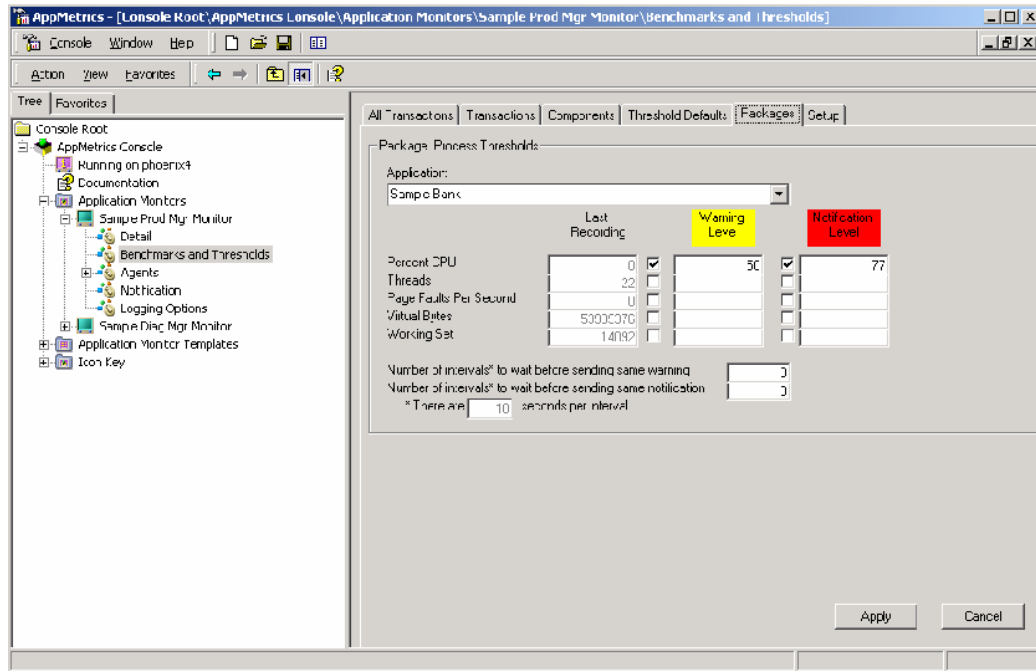


For each individual transaction, you can configure warning and notification thresholds on the following metrics:

Metrics for a Transaction	Description
Number Aborted	Set thresholds to warn or notify when the transaction fails more than a specific number of times during an interval.
Number Started	Set thresholds to warn or notify when the transaction is started more than a specific number of times during an interval.
Duration (ms)	Set thresholds to warn or notify when the average duration of the transaction lasts longer than a specific number of milliseconds during an interval.

Set Package/Application Process Thresholds

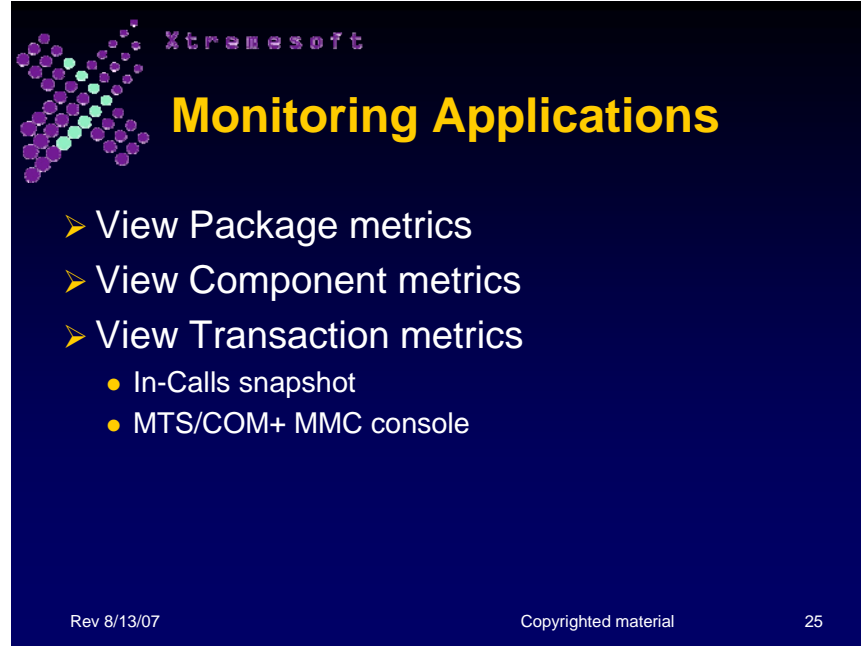
The Package/Application Process Thresholds panel is where you set warning and notification levels for an individual package/application. When a package/application exceeds a specific level of resource usage, AppMetrics can send a notification about the incident.



Naming Transactions

When a monitor sees a transaction for the first time, it will automatically name the transaction after the package/application, component, and perhaps the first called method. The monitor will then list this transaction in the **Unnamed Transactions** list. You can assign a business-friendly name to the unnamed transaction. This name will then be used for the transaction throughout other areas in the monitor, such as the Transactions runtime panel and the Transactions report.

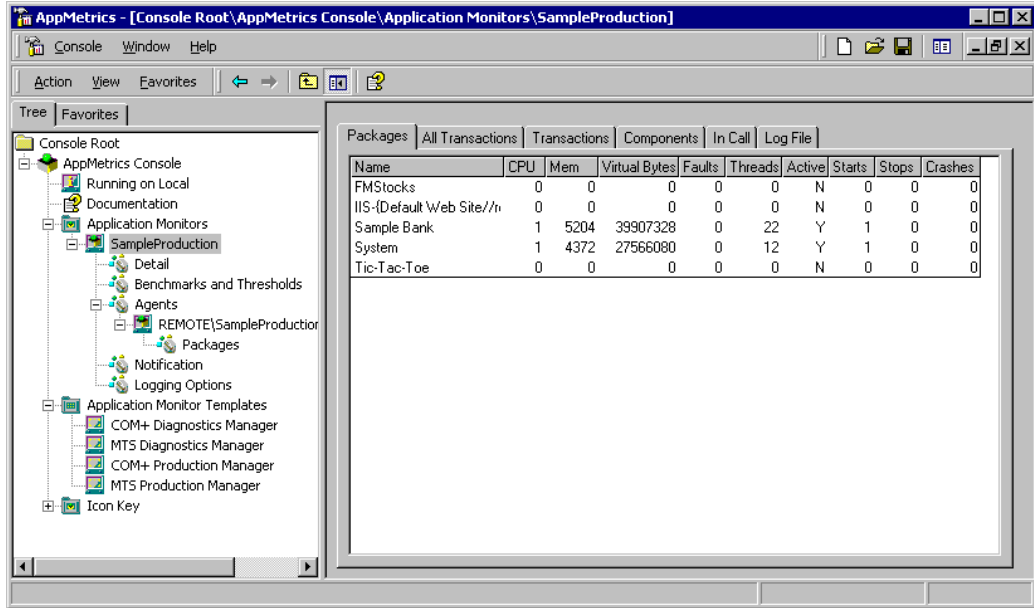
Section 8: Monitoring Applications



After you configure the monitor, you can start it to generate metrics about your monitored packages/applications. Several runtime panels show metrics for different items within the packages/applications. This section shows you where you can view the most recently generated metrics.

View Package Metrics

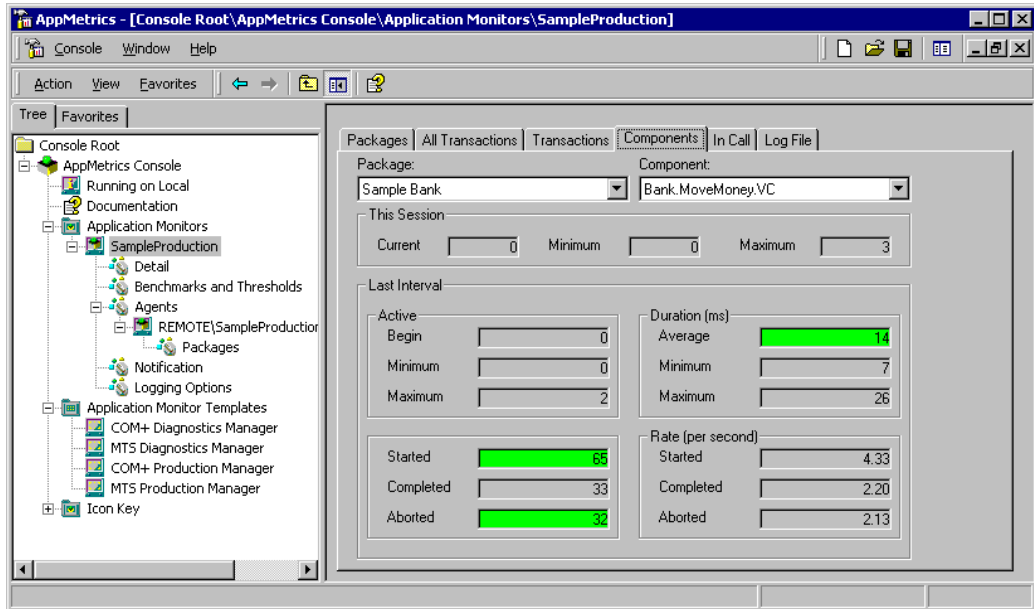
The Applications Panel shows which packages/applications are monitored, as well as some metrics for each. The format is similar to the Windows Task Manager. The metrics are updated every 5 seconds by default.



View Component Metrics

While the monitor is running, you can view the metrics for each monitored component. These metrics reflect the totals and rates of the component during both the current monitoring session and the most recently completed interval.

The following illustration shows the Components runtime panel:



The Package/Application drop-down list contains the names of all the monitored server packages/applications. The Component drop-down list contains all the components that were active within the selected package/application during earlier intervals.

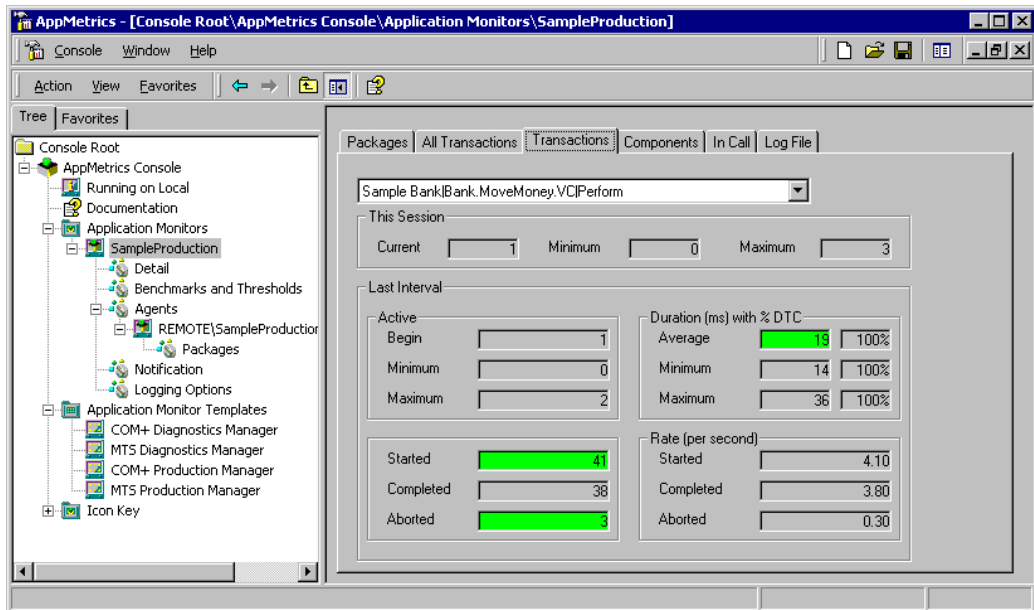
The Component drop-down may also contain components of library packages/applications, but only if they were called within the context of the selected server package/application.

Notice the Started, Aborted, and Duration Average metrics. These metrics were configured for warning and notification thresholds in the Components configuration panel. Whenever any of these three metrics passes a threshold, this panel will change the background color of the metric to reflect the warning or notification condition.

View Transaction Metrics

While the monitor is running, you can view the metrics for each monitored transaction. These metrics reflect the totals and rates of the transaction during both the current monitoring session and the most recently completed interval.

The following illustration shows the metrics for an individual type of transaction.



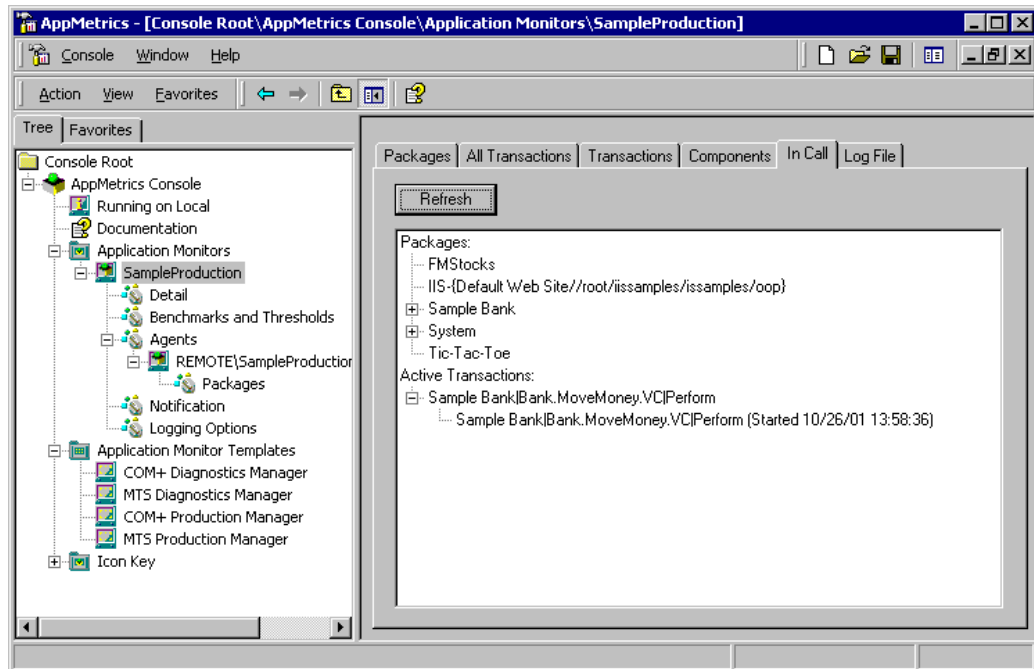
Take note of the Started, Aborted, and Duration Average metrics. These metrics were configured for warning and notification thresholds in the Transactions configuration panel. Whenever any of these three metrics passes a threshold, this panel will change the background color of the metric to reflect the warning or notification condition.

In-Calls Snapshot

On a Production Monitor, the In Call panel shows all active packages/applications, components, and transactions in a tree view. On a Diagnostics Monitor method names will appear instead of transactions in the tree view.

This panel is useful when you are looking for stalled packages/applications or components. For this purpose, the panel shows the method/transaction start time. To update the display, click the Refresh button.

The following illustration shows the In Call runtime panel:



Using the MTS/COM+ Console

AppMetrics provides you with maximum visibility into your MTS and COM+ applications, however there are times when you may need to change an application or components properties, or validate the settings or data that AppMetrics is reporting for an application or component.

You make these changes or validations directly via the MTS/COM+ MMC snap-in. MTS is managed by the MTS Transaction Server Explorer (accessible from the Start -> Programs -> NT 4 Option Pack -> Transaction Server -> Transaction Server Explorer), and COM+ by the Component Services MMC snap-in (found in the Administrative Tools menu of Windows 2000).

In both MTS and COM+ you can change an application or component properties by selecting the object and choosing Properties from the

Action menu. A word of caution is called for however, because a simple property change can disable or cause unpredictable behavior in a previously functioning application or component.

One of the more valuable functions of the MTS/COM+ MMC console is to validate the data being provided by AppMetrics.

For instance, if you have a component that has remained in your AppMetrics In-Calls display for more than a few intervals, you may want to validate whether that component is really still instantiated. The MTS/COM+ console Status View can verify that.

In the example below, we have selected the Status View of the Sample Bank components folder. If a component is still instantiated the In Call column will have at least an 1 in it. Other useful data in this view includes the number of objects instantiated, the number of components activated, and the length of time the component has been In Call.

Similarly, you can select the package/application folder and get a Status View that can verify which applications are active at a given time.

The screenshot shows the MTS/COM+ MMC console. The left pane shows a tree view with the following structure:

- Console Root
 - Component Services
 - Computers
 - My Computer
 - COM+ Applications
 - COM+ QC Dead Letter Queue
 - COM+ Utilities
 - IIS In-Process Applications
 - IIS Out-Of-Process Pooled Applications
 - IIS Utilities
 - Sample Bank
 - Components
 - Bank.Account
 - Bank.Account.VC
 - Bank.CreateTable
 - Bank.GetReceipt
 - Bank.MoveMoney
 - Bank.MoveMoney.VC
 - Bank.UpdateReceipt
 - Bank.UpdateReceipt.VC
 - BankCreateAComponent...

The right pane shows the Status View for the selected 'Components' folder, displaying a table with 9 objects:

Prog ID	Objects	Activated	Pooled	In Call	Call Time (ms)
Bank.Account					
Bank.Account.VC	1	1		1	10
Bank.CreateTable					
Bank.GetReceipt					
Bank.MoveMoney					
Bank.MoveMoney.VC	1	1		1	10
Bank.UpdateReceipt					
Bank.UpdateReceipt.VC	0	0		0	0
BankCreateAComponent...	0	0		0	0



Section 9: Lab 3 - Configuring Monitors & Viewing Metrics

Objectives

- **Configure an AppMetrics Production Monitor**
- **Configure an AppMetrics Diagnostic Monitor**
- **View real-time metrics on an AppMetrics Production Monitor**
- **View real-time metrics on an AppMetrics Diagnostics Monitor**

Prerequisites

- The installation and configuration of the *Sample Bank* application on an AppMetrics Agent machine.
- The creation of an AppMetrics Production Monitor on the Manager machine and the addition of an Agent to that monitor
- The creation of an AppMetrics Diagnostics Monitor on the Manager machine and the addition of an Agent to that monitor

Preparation

Students should perform this Lab individually or as a team on an AppMetrics Manager machine console, or on an AppMetrics Console/Reports client.

Students should ensure that the *Sample Bank* application on the AppMetrics Agent machine is still running and generating load before beginning the sections of the lab dealing with the viewing of real-time metrics.



Note: Do *not* attempt to start your Manager Monitor before beginning the configuration steps below!

First, configuration changes cannot be defined or changed while a monitor is running. Second, some undesired configuration defaults may be inadvertently latched-in if you attempt to start your new monitor before configuration is completed.

Step 1: Configuring a Production Monitor

1. The first step in configuring a Production Monitor is to select which packages/applications you wish to collect data about. By default, a Production Monitor is set to monitor all packages/applications on the Agent machine.

For this lab we will configure the Production Monitor you created in Lab 2, and select the **Sample Bank** package and the **System** package as the packages/applications to monitor.

You can configure this Production Monitor from either the AppMetrics Manager machine, or from one of the AppMetrics Console/Reports clients you created in Lab 1.

Select these applications by expanding the tree display of **Application Monitors**, then expanding the tree display of your **Production Monitor**, and finally expanding the tree display of **Agents** in that Production Monitor.

Select the **Applications** node under the Agent, and a list of recognized Applications will appear in the right panel in a check-box list format.

Click on the **Select Package from list** radio button. Now place a check in the applications **Sample Bank** and **System** (see Figure 1).

Then click the **Apply** button.

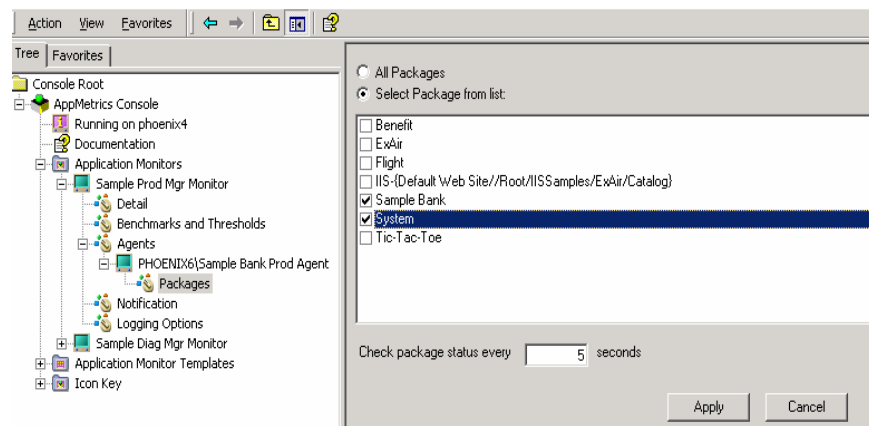


Figure 1

2. We will now configure the **Interval Length**, which is the time span between calculations of the All Transactions, Transactions, and Components metrics.

Select the **Benchmarks and Thresholds** node underneath your Production Monitor. Then click on the **Setup** tab to switch to the Setup panel (see Figure 2).



By default, a Production Monitor's Interval Length is set to 60 seconds. This means that AppMetrics will collect data from that Agent every 60 seconds and at the end of that interval AppMetrics will process the data to calculate totals, averages, and rates.

Interval Length also applies to the display of these transactional and component metrics, which will be updated every interval. An exception to this are **Package/Application** metrics, which by default are updated approximately every 5 seconds.

For the sake of this Lab, we will set our Interval Length to 10 seconds so that we can easily see the updating of metrics on our AppMetrics run-time panels.

Type **10** into the **Interval Length** field.

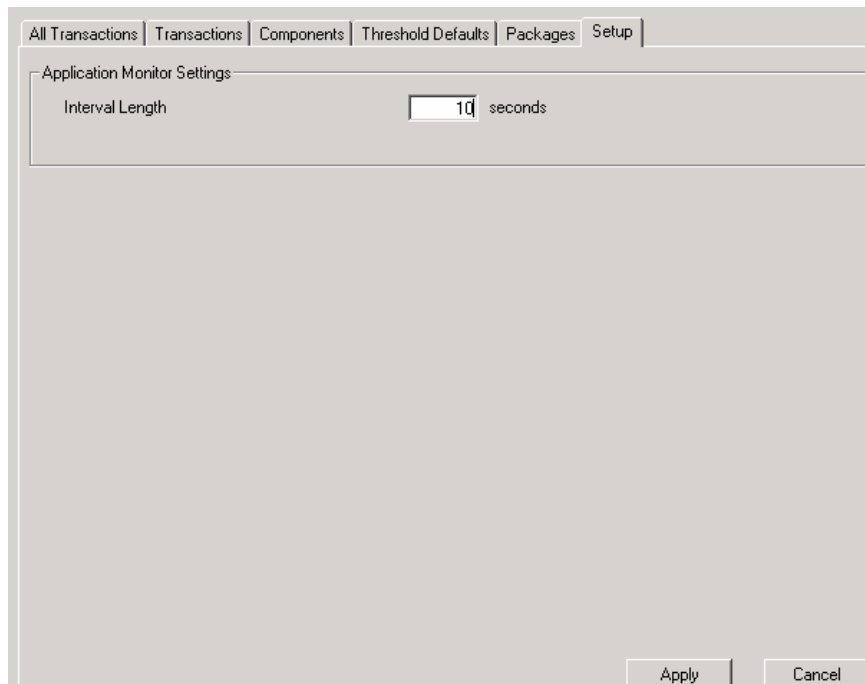


Figure 2

- Next we will select the desired level of detail for our Production Monitor to collect. The **Detail Levels** (transactional and component) determine the amount of data that our Production Monitor will collect.

For this Lab we will select a **Transactional Detail Level of Component and Method**, and a **Component Filtering Level of All Components**. In other words, we will be selecting the maximum level of detail possible for a Production Monitor.

Click on the **Detail** node under your Production Monitor tree display, and you will see the Detail Level configuration panel in the right hand pane.

Click on the **Identify by:** and the **Component and Method** radio buttons in the Transactional Detail Level section.

Next click on the **All Components** radio button on the Component Filtering section.

Click **Apply** (see Figure 3).

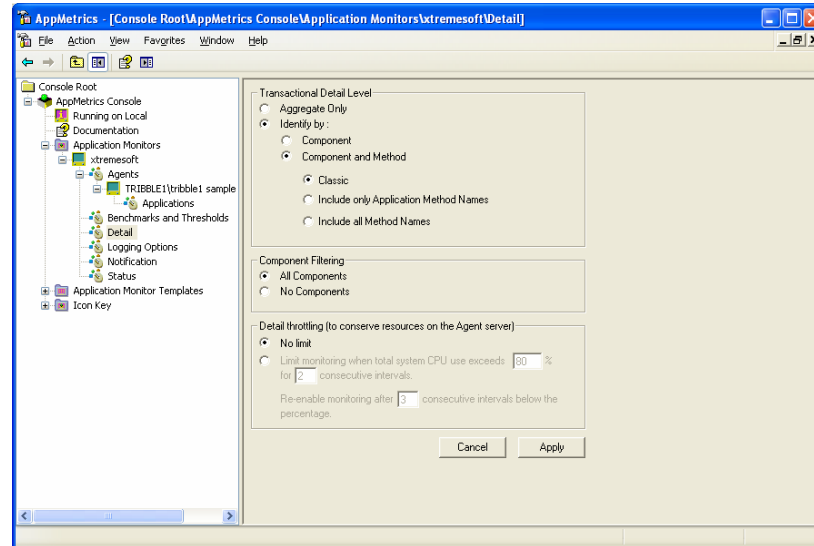


Figure 3

- When components and transactions are added to a monitor, AppMetrics applies default values to their benchmark and threshold settings.



A benchmark is an absolute value against which a particular metric will be compared. For instance, the benchmark for transaction lifetimes is specified in milliseconds in the **Duration (ms)** field of the Threshold Defaults configuration panel.

A threshold is configured by inputting the particular percentage of a benchmark at which a notification or warning will be generated. For example, if the benchmark of Duration field of a transaction is set to 200, and the Notification Level threshold is set to 75%, this means that if the duration of a transaction exceeds 150 ms, a notification will be generated.

Note: **Package Process Thresholds** differ from others because the Warning and Notification Level thresholds in this case are set by absolute value, not by percentage of benchmark.

Default values for benchmarks and thresholds are specified in the **Threshold Defaults** configuration panel. These values are assigned to any component or transaction, respectively, until the user explicitly

overwrites them in the **Components** or **Transactions** configuration panels.

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

For this Lab we will set default thresholds to zero.

Click on the **Benchmarks and Thresholds** node in the Production Monitor tree list. Next click on the **Threshold Defaults** tab in the configuration panel (see Figure 4).

	Current Benchmark	Warning Level	Notification Level
Default Component Threshold			
Number Aborted	0	0 %	0 %
Number Started	0	0 %	0 %
Duration (ms)	0	0 %	0 %
Default Transaction Thresholds			
Number Aborted	0	50 %	75 %
Number Started	0	50 %	75 %
Duration (ms)	0	50 %	75 %

Figure 4

- Applications, components and transactions are not recognized, and thus not listed in a Monitor, until the Monitor is started. So before we begin the next parts of this Lab - setting specific thresholds on applications and components, as well as naming a transaction - we must start our Production Monitor.

First, we will start a specific load-generation test in **Sample Bank**.

If Sample Bank is already processing a test stop it by selecting **Stop** from the **Action** menu. From the Sample Bank console select **Account -> Debit** from the **Test** menu. Start the new test by selecting **Start** from the **Action** menu.

Returning to AppMetrics, select the Production Monitor icon in the console tree. Then choose **Start Monitor** from the **Action** menu.

Note: Click **OK** if you receive a pop-up dialog asking if you wish to apply your configuration changes.

6. We will continue with our configuration changes by setting a specific process threshold on the **Sample Bank** application.

Note: Package Process Thresholds (i.e. specific application thresholds) differ because the Warning and Notification Level thresholds in this case are set by absolute value, not by percentage of benchmark, as is the case with other thresholds.

After starting your Monitor allow it to run for a minute or so to collect some working data. Then stop the Monitor so that we can make configuration changes upon it again.

To stop the Monitor select the Production Monitor icon in the console tree. Then choose **Stop Monitor** from the **Action** menu.

For the purposes of this Lab we will set a **Warning Level** threshold of 50% of CPU and a **Notification Level** threshold of 77% of CPU.

Note: These numbers are arbitrary and not representative of any appropriate or inappropriate package CPU utilization.

To set our **Package Process Thresholds** for **Sample Bank** you should click the check box next to the **Last Recording** column of the **Percent CPU** threshold row. This will enable a Warning Level on this threshold, and also set the threshold by default to the value found in the Last Recording column. Now manually input **50** in the **Warning Level** column. Similarly, set the **Notification Level** column to **77** (see Figure 5).

Package Process Thresholds

Application: Sample Bank

	Last Recording	Warning Level	Notification Level
Percent CPU	77 <input checked="" type="checkbox"/>	50 <input checked="" type="checkbox"/>	77 <input checked="" type="checkbox"/>
Threads	16 <input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Page Faults Per Second	2 <input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Virtual Bytes	55779328 <input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Working Set	25036 <input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Number of intervals* to wait before sending same warning:

Number of intervals* to wait before sending same notification:

* There are seconds per interval

Apply Cancel

Figure 5

7. Next we will set benchmarks and thresholds for a specific component.

Since we have already run our Production Monitor it will have recognized most if not all of the components in our load-generation application **Sample Bank**.

In this Lab we will set a specific benchmarks and thresholds for the Component **Bank.MoveMoney.VC**. Although you should note that the settings we use are essentially arbitrary.

Click on the **Benchmarks and Thresholds** node in the Production Monitor tree list. Next click on the **Components** tab in the configuration panel. Select **Sample Bank** from the **Package** pull-down list, and **Bank.MoveMoney.VC** from the **Component** pull-down list. Now the defaults thresholds should appear in the component's configuration panel, as well as the last recorded values for this component in the **Last Recording** column (see Figure 6).

For this Lab input the benchmark for **Number Aborted** to **1**, and the Notification Level threshold for this benchmark to **100%**. So we will be notified if one or more instances of this component are aborted.

Set the benchmark for **Number Started** to **0**, since we are not interested in being warned or notified based on component instance starts. Lastly, set the benchmark for component instance **Duration** to **200**. This means we will be warned when the average duration for a component instance exceeds 100 ms, and notified when the average duration for a component instance exceeds 150 ms.

The screenshot shows the configuration panel for the component **Bank.MoveMoney.VC**. The panel is titled "Bank.MoveMoney.VC" and contains a table with the following data:

	Last Recording	Current Benchmark	Warning Level	Notification Level
Number Aborted	0	1	0 %	100 %
Number Started	15	0	50 %	75 %
Duration (ms)	58	200	50 %	75 %

Below the table is a "Copy to Current" button. At the bottom of the panel are "Copy All To Current", "Apply", and "Cancel" buttons.

Figure 6


8. One of the useful features of AppMetrics is the ability to give transactions a business-friendly name. By default, when a Monitor sees a transaction for the first time it will automatically name the transaction after the package/application, component, and perhaps the first called method. The monitor will then list this transaction in the **Unnamed Transactions** list. It will prove a useful aid to you to name frequent transactions so that AppMetrics will use that name when presenting metrics in the run-time panels as well as in the AppMetrics reports.

In this Lab we will name one transaction to demonstrate this functionality. Then we will set specific benchmarks and thresholds on this new named transaction.

While still on the **Benchmarks and Thresholds** node in the Production Monitor tree list, click on the **Transactions** tab in the configuration panel.

Note the Unnamed Transactions in the list at the top left of the panel. This list demonstrates how AppMetrics will by default label a transaction with the Package | Component | Method name of the first discovered object in that transaction.

In this Lab we will provide a business-friendly name for the transaction **Sample Bank | Bank.Account.VC | Post** – we will name it “**Show Me the Money**”.

Click on the unnamed transaction in the listbox, then click the  button at the right of the unnamed transactions listbox (see Figure 7).

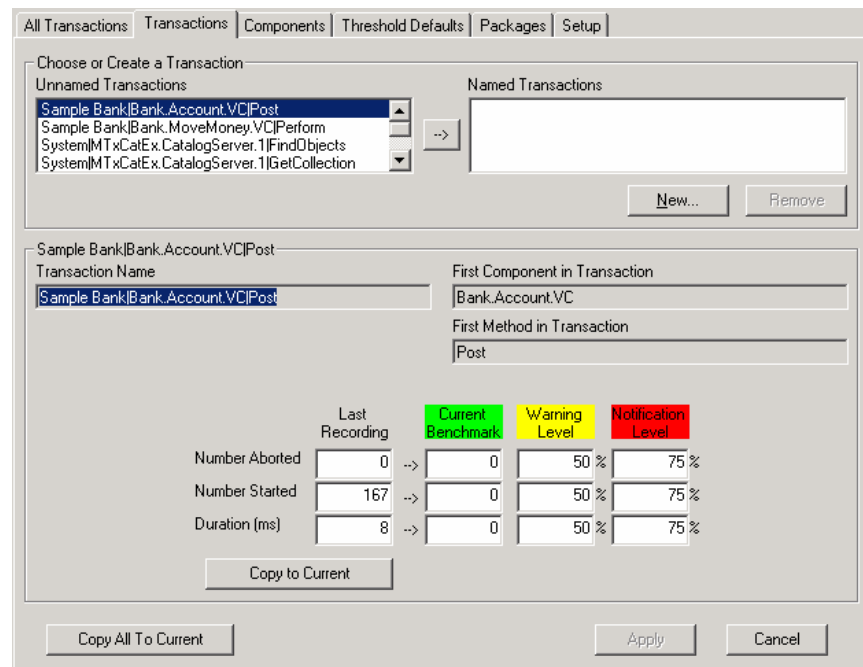


Figure 7

The **Name Transaction** dialog box will popup. Input the name “Show Me the Money” in the **Transaction Name** field (see Figure 8).

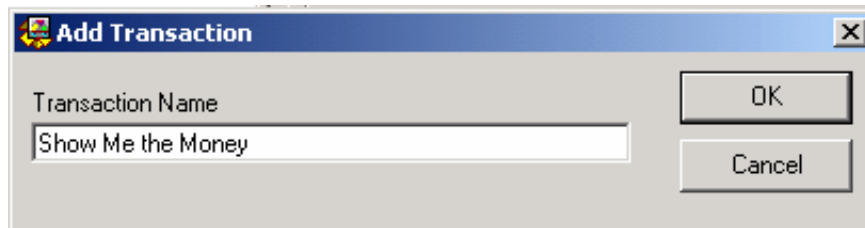


Figure 8

Now input the following benchmarks and threshold levels for this new named transaction **Show Me the Money**, (see Figure 9).

Set the current benchmark for **Number Aborted** to **1**, warning level to **0**, and notification level to **100**.

Set the current benchmark for **Duration** to **12**.

Click **Apply**.

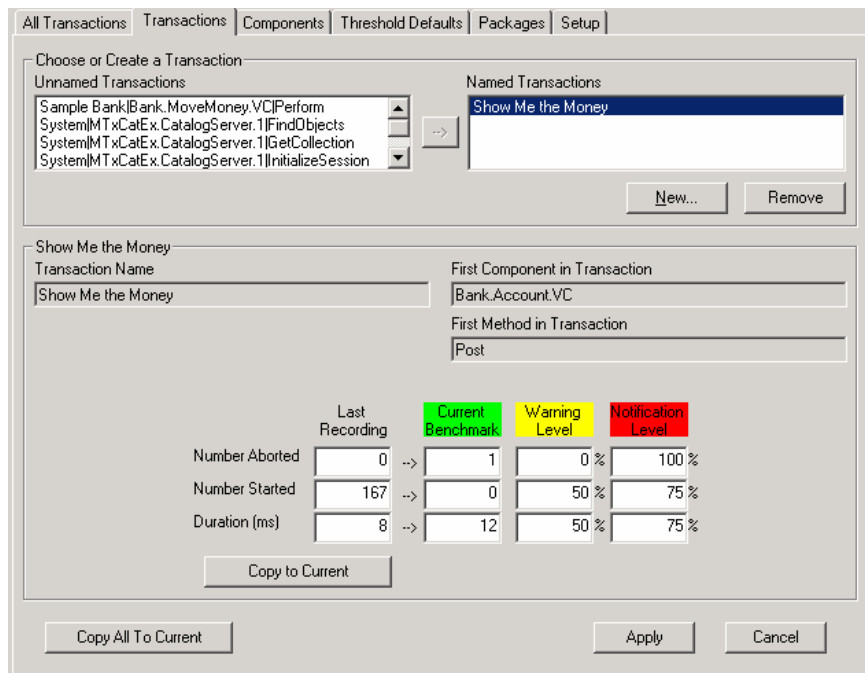


Figure 9

Step 2: Configuring a Diagnostics Monitor

1. A Diagnostics Monitor, as you discovered in Lab 2, is different than a Production Monitor.

A Diagnostics Monitor collects and analyzes data real-time, so there is no configuration of a Monitoring Interval.

A Diagnostics Monitor also does not perform any warnings or notifications, so the monitor has no need to configure benchmarks and thresholds. Similarly, a Diagnostics Monitor does not permit the naming of transactions.

The configuration changes necessary in a Diagnostics Monitor are to select which packages/applications you wish to monitor, and to select which components to filter out from the **Component Filters** configuration panel. As in a Production Monitor, by default a Diagnostics Monitor is set to monitor all packages/applications on the Agent machine, and by default to monitor all components

For this lab we will configure the Diagnostics Monitor you created in Lab 2, and select the **Sample Bank** package and the **System** package as the packages/applications to monitor.

You can configure this Diagnostics Monitor from either the AppMetrics Manager machine, or from one of the AppMetrics Console/Reports clients you created in Lab 1.

Select these applications by expanding the tree display of **Application Monitors**, then expanding the tree display of your **Diagnostics Monitor**, and finally expanding the tree display of **Agents** in that Diagnostics Monitor.

Select the **Applications and Detail** node under the Agent, and a list of recognized Applications will appear in the right panel in a check-box list format.

Click on the **Select Package from list** radio button. Now place a check in the applications **Sample Bank** and **System** (see Figure 10). If not already checked, click on the check box **Method Call Detail**.

Then click the **Apply** button.

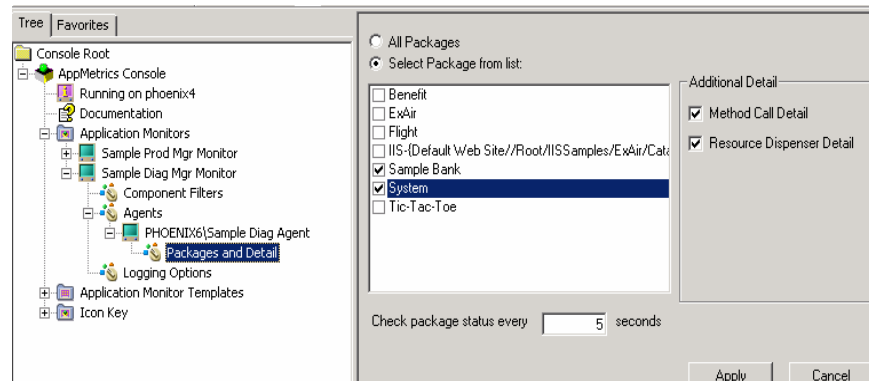


Figure 10

Step 3: View Metrics on an AppMetrics Production Monitor

1. To view metrics we must start the Production Monitor.

Select the Production Monitor icon in the console tree. Then choose **Start Monitor** from the **Action** menu.

Note: Click **OK** if you receive a pop-up dialog asking if you wish to apply your configuration changes.

2. The first run-time panel we want to view is the **Packages/Applications**. This run-time panel shows which packages/applications are monitored, as well as some metrics for each. The format is similar to Windows Task Manager.

Select the Production Monitor icon in the AppMetrics Console tree list. The AppMetrics run-time panels will appear in the pane on the right.

Click on the **Applications** tab (in COM+ this tab is called the **Applications** tab). You will now observe a live display of package level metrics (see Figure 11).

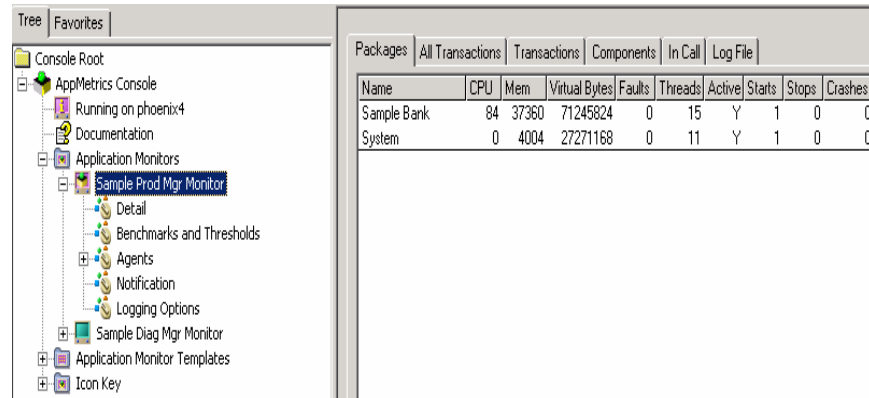


Figure 11

3. Next select the **All Transactions** run-time panel. The All Transactions runtime panel displays summary metrics about the transactions on your Agent machine. All transactions are included — not just those that are explicitly marked transactional (see Figure 12).

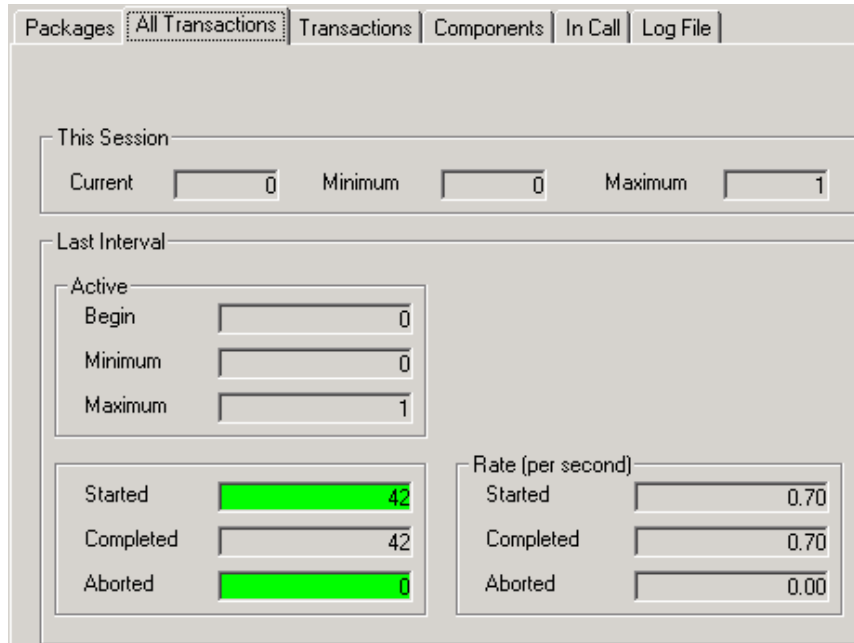


Figure 12

- Next select the **Transactions** run-time panel. The Transactions runtime panel displays metrics about a specific user transaction (see Figure 13).



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.

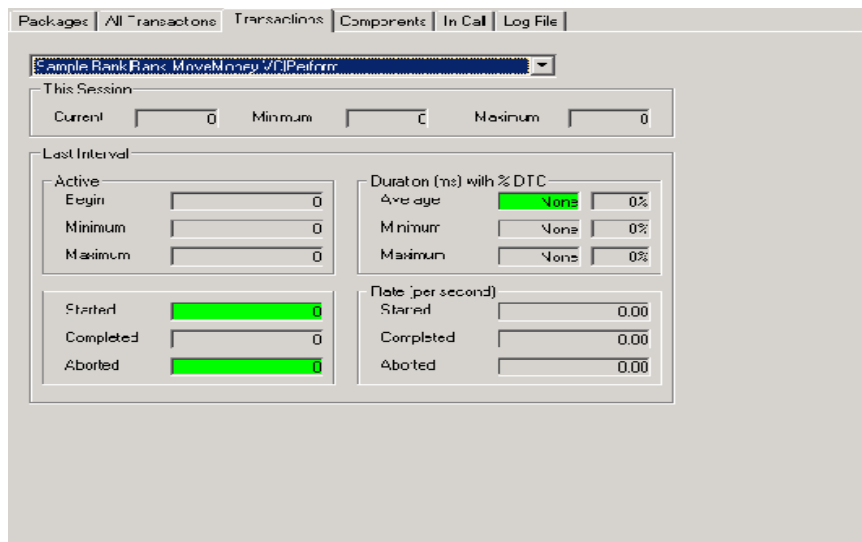


Figure 13

- Next select the **Components** run-time panel. The Components Runtime Panel enables the user to select the metrics to be displayed by component (see Figure 14).

The Application drop-down list contains the names of all the server applications that were selected in the Applications configuration panel. The Component drop-down contains all the components that were active during the prior session within the selected server application.

The Component drop-down may also contain components of library applications if they were called within the context of the selected server application.

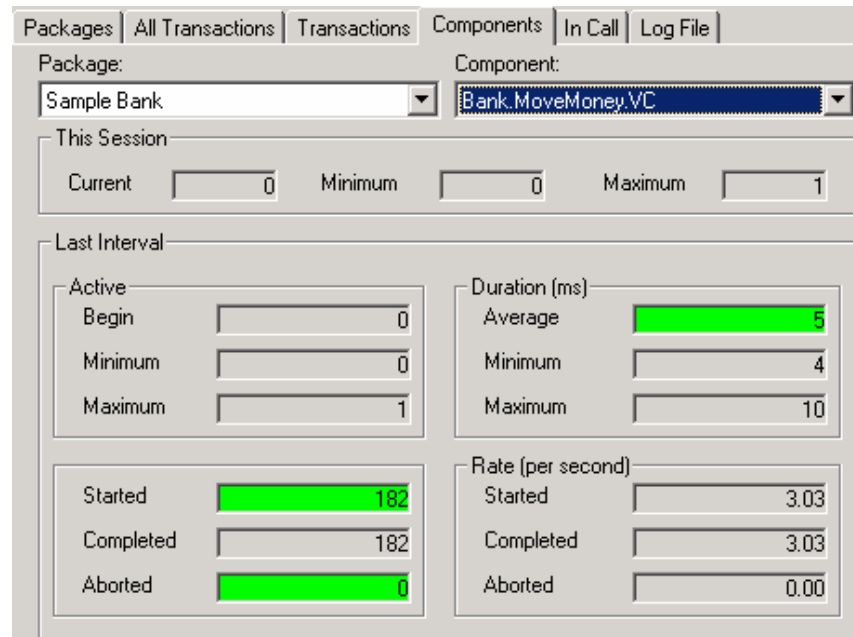


Figure 14

- Next select the **In Call** panel. The In Call panel shows all active applications, components, and methods in a tree view (see Figure 15).

This panel is useful when looking for stalled applications or components. For this purpose, the panel shows the method start time. To update the display, click the Refresh button.

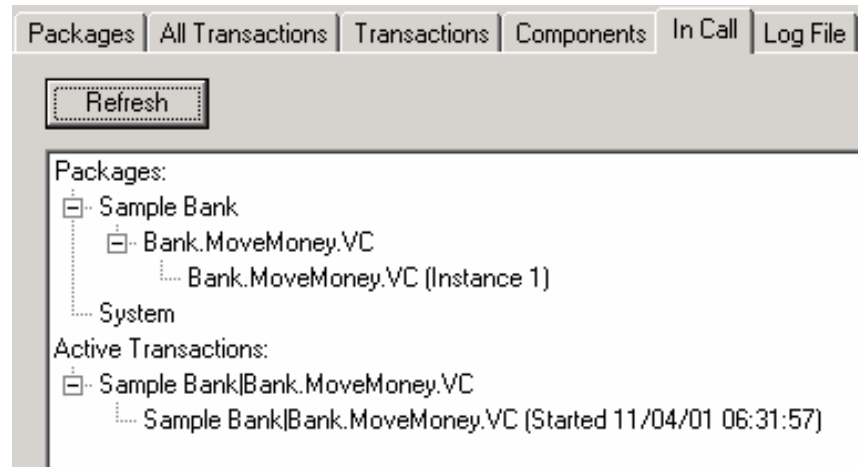


Figure 15

Step 4: View Metrics on an AppMetrics Diagnostics Monitor

1. To view metrics we must start the Diagnostics Monitor.

Select the Diagnostics Monitor icon in the console tree. Then choose **Start Monitor** from the **Action** menu.

Note: Click **OK** if you receive a pop-up dialog asking if you wish to apply your configuration changes.

2. There are only two run-time panels of interest in the Diagnostics Monitor – the **Packages/ Applications** panel, and the **In Call** panel.

These panels provide no additional information or functionality over the same panels in the Production Monitor.


So for the purposes of this Lab review items 2 and 6 from Step 3 above for explanations and a walk-through of the purpose and functionality of these Diagnostic Monitor run-time panels.



Review Questions

1. When does AppMetrics apply default benchmarks and thresholds to a Component in a Production Monitor?
2. What types of functionality are present in a Production Monitor but not in a Diagnostics monitor?
3. What impact would a shorter interval length have on the calculation of a Component Duration value in a Production Monitor? How would this differ in a Diagnostics Monitor?
4. In which configuration panel does the calculation of thresholds differ from that used in the Threshold Defaults panel?
5. What relation does the Interval Length have to the display of metrics in the Packages/Applications run-time panel?

Section 10: Configuring Notifications



Configuring Notifications

- What is a Notification?
- Delivery mechanisms
 - SMTP
 - SNMP (connect to EDS' Unicenter)
 - Event Log
 - Component
- Configuring a delivery mechanism
 - How
 - Who
 - Logging

Rev 8/13/07 Copyrighted material 26

What is a Notification?

When the monitor is running, and any of your All Transactions, Components, and Transactions thresholds are exceeded at the notification level, the monitor is able to send a notification to the personnel responsible for the system. The monitor can also send a notification whenever your packages/applications exceed their warning or notification thresholds for resource usage.

Delivery Mechanisms

When the monitor detects conditions resulting in notifications, it can send the notifications through the following delivery mechanisms:

- SMTP
- SNMP
- Event Log
- Custom Component

The following three subsections describe each delivery mechanism.

SMTP

The SMTP delivery mechanism enables the monitor to send notifications via e-mail. Additionally, if an e-mail account is configured to send messages to electronic pagers, the monitor can page the recipient about the notification condition.

SNMP

When the monitor detects a notification condition, it can send notifications via traps generated by its SNMP agent to an SNMP management system.

Event Log

When the monitor detects a notification condition, it can create an event in the Applications log of the Windows Event Viewer. If you have a system management product that monitors the Windows Event Log, then AppMetrics can be integrated with that management tool.

Custom Component

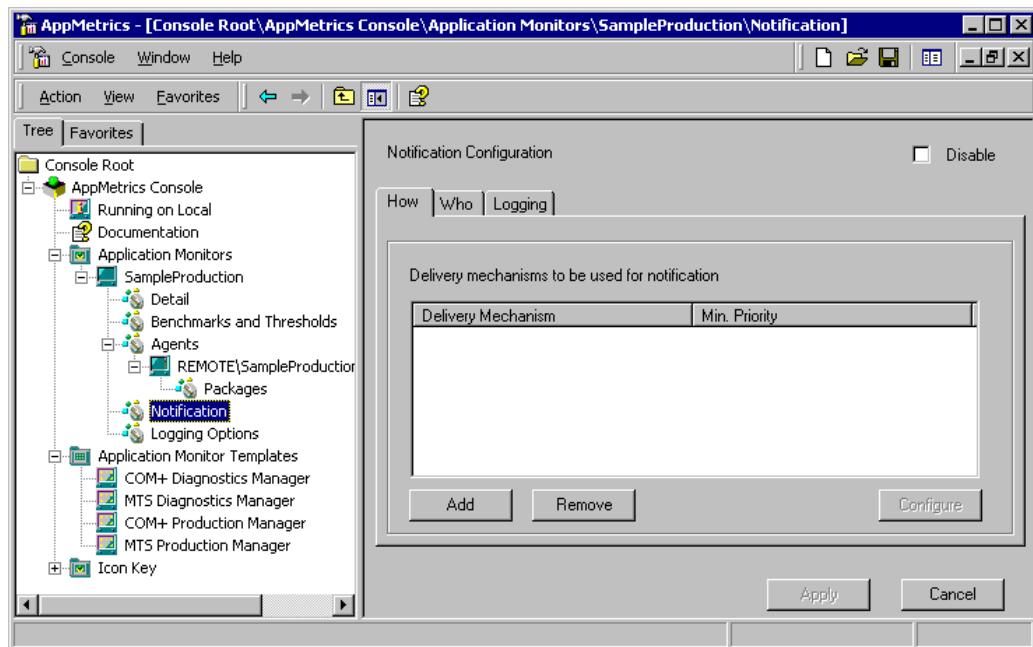
When the monitor detects a notification condition, it can start a custom component that you or Xtremesoft has created. This script can do anything that a normal script will allow.

Configuring a Delivery Mechanism

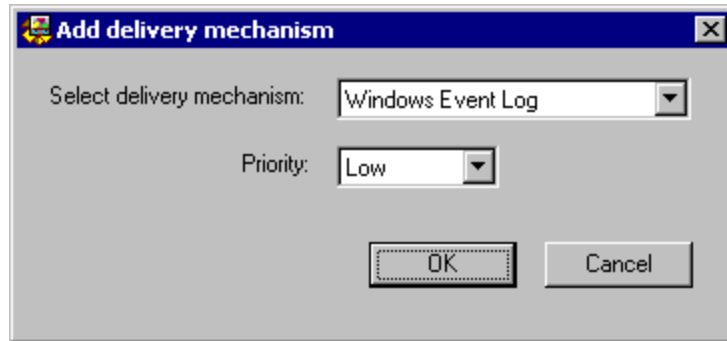
All the work to add and configure a delivery mechanism is performed in the Notification Configuration panel.

How Tab

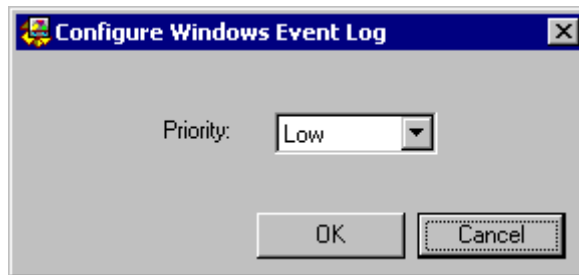
The How tab in the Notification Configuration panel lets you add and configure delivery mechanisms. The following illustration shows the How tab:



When you click the Add button, this opens the Add delivery mechanism dialog:



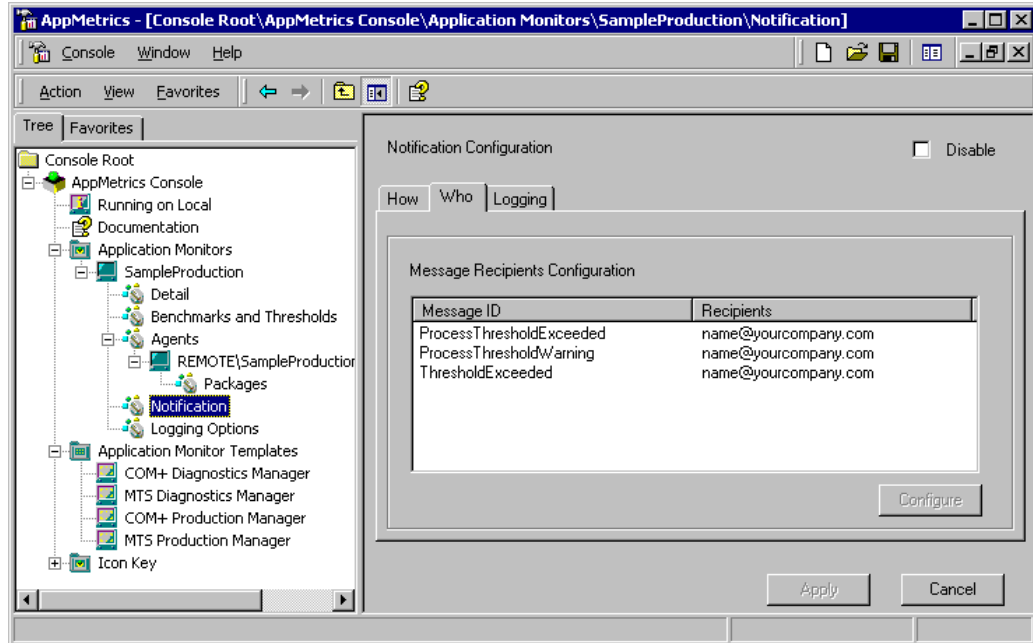
After you add the delivery mechanism, you can configure it in the Configuration dialog. The following illustration shows the Configure Windows Event Log dialog:



A different Configuration dialog is available for the other types of delivery mechanisms.

Who Tab

The Who tab is where you specify the e-mail recipients for the notifications. You only need to complete this screen if you use SMTP mail as a delivery mechanism for notifications.



In the Message Recipients Configuration list, three types of notifications are available based on where the notification condition occurred:

- **Process Threshold Exceeded** – This occurs when a package/application has exceeded one of its configured notification thresholds.
- **Process Threshold Warning** – This occurs when a package/application has exceeded one of its configured warning thresholds.
- **Threshold Exceeded** – This occurs when a transaction, component, or All Transactions benchmarked metric has exceeded its notification threshold.

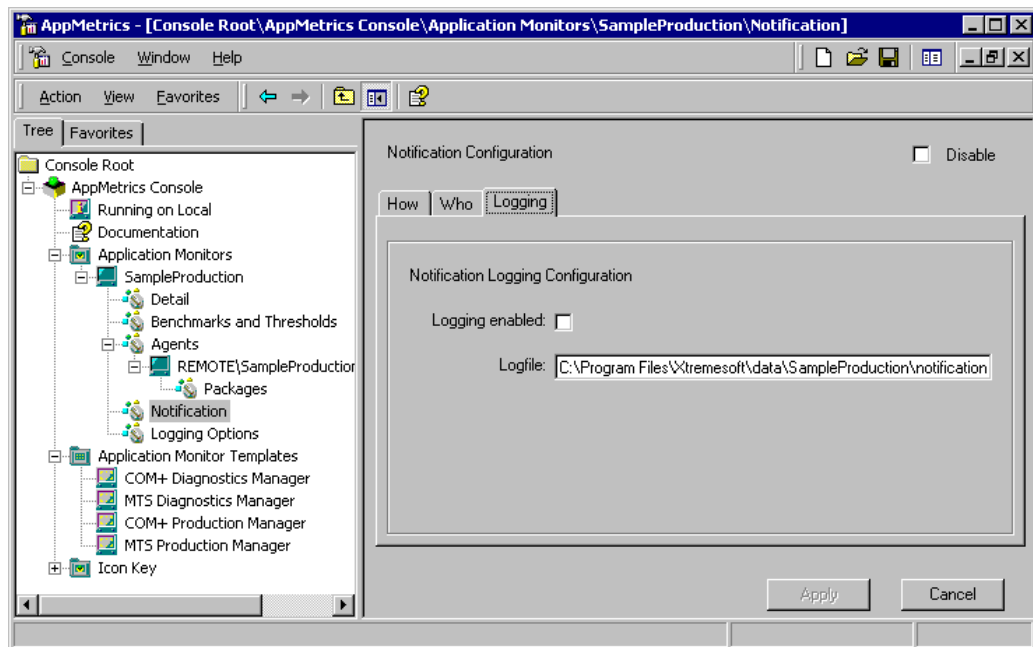
For each type of notification, you can configure a unique list of email recipients. For example, you can send an e-mail to one or more Systems Administrators for the Process Threshold Exceeded notifications, and you can send an e-mail to your Applications Manager for Threshold Exceeded notifications.

Logging Tab

In addition to sending notifications via the various delivery mechanisms, the monitor can also record these notifications into a log. The notification log contains entries for all the notifications sent from the monitor through the configured delivery mechanisms. You can use the notification log as a running tally of all the notifications sent from the monitor.

Once the monitor produces a notification log file, you can view the log file in a text editor such as Notepad.

The following illustration shows the Logging tab in the Notification Configuration panel:





Section 11: Lab 4 - Creating Event Log Notifications

Objectives

- **Configure an Event Log Notification**
- **Adjust benchmarks & thresholds to trigger Notification**
- **Verify Notification**

Prerequisites

- The installation and configuration of the *Sample Bank* application on an AppMetrics Agent machine.
- The creation of an AppMetrics Production Monitor on the Manager machine and the addition of an Agent to that monitor
- The configuration of that AppMetrics Production Monitor to set default thresholds and create a named transaction

Preparation

Students should perform this Lab individually or as a team on an AppMetrics Manager machine console, or on an AppMetrics Console/Reports client.

Students should ensure that the *Sample Bank* application on the AppMetrics Agent machine is still running and generating load before beginning the sections of the lab dealing with the viewing of real-time metrics.

Step 1: Configuring an Event Log Notification

1. An AppMetrics Production Monitor can send a notification whenever any of the All Transactions, Components, and Transactions thresholds you have set are exceeded at the Notification Level. A Production Monitor can also send a notification whenever an Agent's packages/applications exceed their warning or notification thresholds for resource usage.

For all thresholds except Package Process Thresholds, AppMetrics will generate a notification once every five intervals for any thresholds exceeded – regardless of the number of notifications generated.

Package Process thresholds, via the Package configuration panel on the Benchmarks and Thresholds node, can be configured to wait a



specific number of intervals before sending out another notification or warning of the same type.

A Notification can be configured to use one of four delivery mechanisms – **SMTP**, **SNMP**, **Windows Event Log** and a **Custom Component**. Any combination of the four can be configured to run at the same time. Furthermore, a priority of High, Medium, or Low can be specified for the messages produced by each delivery mechanism.

SMTP, SNMP and Custom Component notification configurations are outside of the scope of this Lab, so today we will simply configure an Event Log notification on your Production Monitor for a specific Transaction. For further information on configuring SMTP, SNMP and Custom Component notification services see Chapter 3 “Application Monitors” of the AppMetrics for Transactions documentation set.

The **Sample Bank** package will be the application we use to configure and test our notification.

2. First we will create a new Delivery Mechanism for our Notification.

Select the **Notification** icon in the Production Monitor tree list. By default it will position the Notification Configuration dialog box to the **How** tab.

Click **Add**.

In the Add Delivery Mechanism dialog box, select **Windows Event Log** from the **Select Delivery Mechanism** drop-down list. Then select **High** from the **Priority** drop-down list (see Figure 1).

Click **OK**.

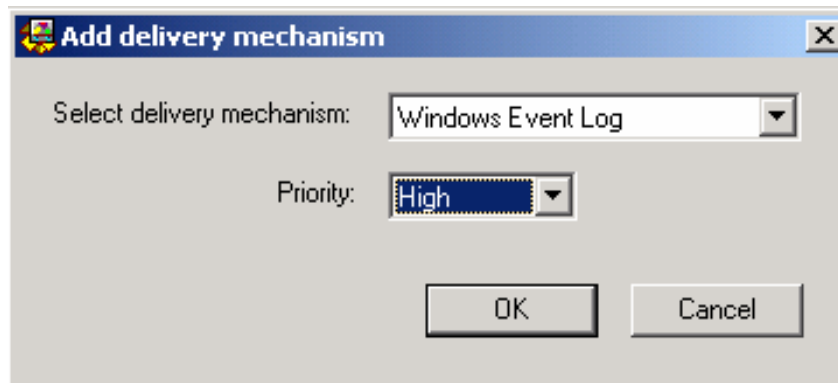


Figure 1

3. Since we are limiting ourselves to Windows Event Log notifications, it is not necessary to configure a specific recipient for notifications (a required step for SMTP configurations only).

However for purposes of this training, you should click the **Who** tab of the Notification Configuration dialog box and select the **ThresholdExceeded** type from the **Message Recipients Configuration** list box, then click **Configure**.

Note that the **Message ID** and **Message Template** fields are preconfigured and thus display-only. The only value a user can modify is the addition of a recipient (see Figure 2).

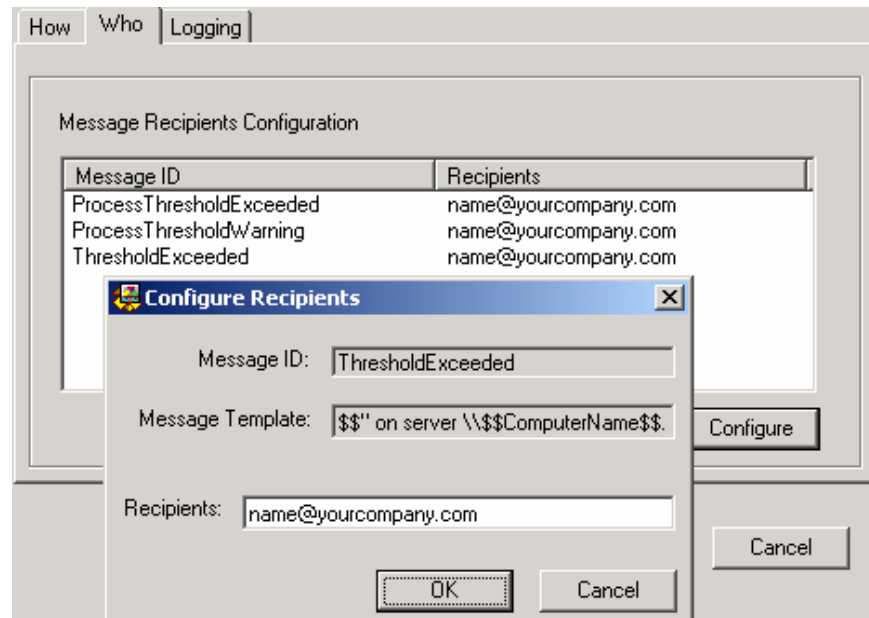


Figure 2

4. You are finished configuring a Notification.
Click **Apply**.

Step 2: Adjust Benchmarks and Thresholds to Trigger Notification

1. First, we will start a specific load-generation test in **Sample Bank**.
If Sample Bank is already processing a test, stop it by selecting **Stop** from the **Action** menu.
From the Sample Bank console select **Account** → **Debit** from the **Test** menu. Start the new test by selecting **Start** from the **Action** menu.
2. We will reuse our **Show Me the Money** Named Transaction from Lab 3 to generate our Notification.

If you have not already started your Production Monitor, do so now by selecting the Production Monitor icon and selecting **Start Monitor** from the **Action** menu.

Allow the Production Monitor to run for one to three minutes, then stop the monitor by selecting **Stop Monitor** from the **Action** menu.

3. Select the Benchmarks and Thresholds icon from the Production Monitor's tree list, and click on the Transactions tab. Select the **Show Me the Money** Named Transaction by clicking on it in the Named Transactions list box.

The latest run-time metrics for this transaction will be listed in the fields at the bottom of the panel. Note the **Last Recording** value for Duration. Set your **Current Benchmark** value to match it, then click **Apply**. This will ensure that at least one Notification will be generated when your Named Transaction **Show Me the Money** meets or exceeds its last recorded value (see Figure 3).

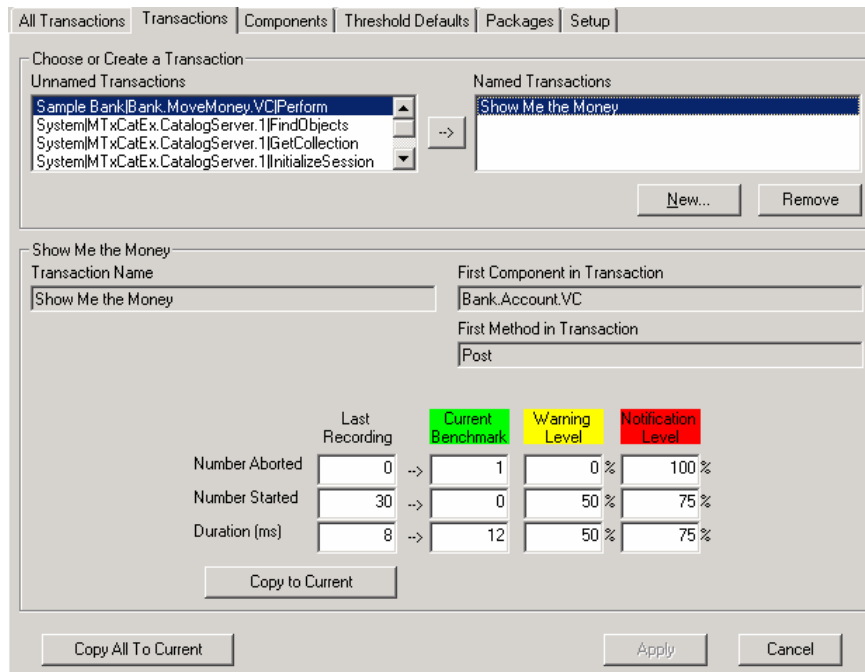


Figure 3

Step 3: Verify the Notification

1. Restart your Production Monitor. Allow the Monitor to run for between one and three minutes.

Now open up your Windows Event Log, and click on the **Application Log** icon.

2. One of the last few entries should be a Notification from AppMetrics with a type of **Error**. Open up one of these errors and confirm that it is from AppMetrics and is a Notification in regard to the Transaction **Show Me the Money** (see Figure 4).

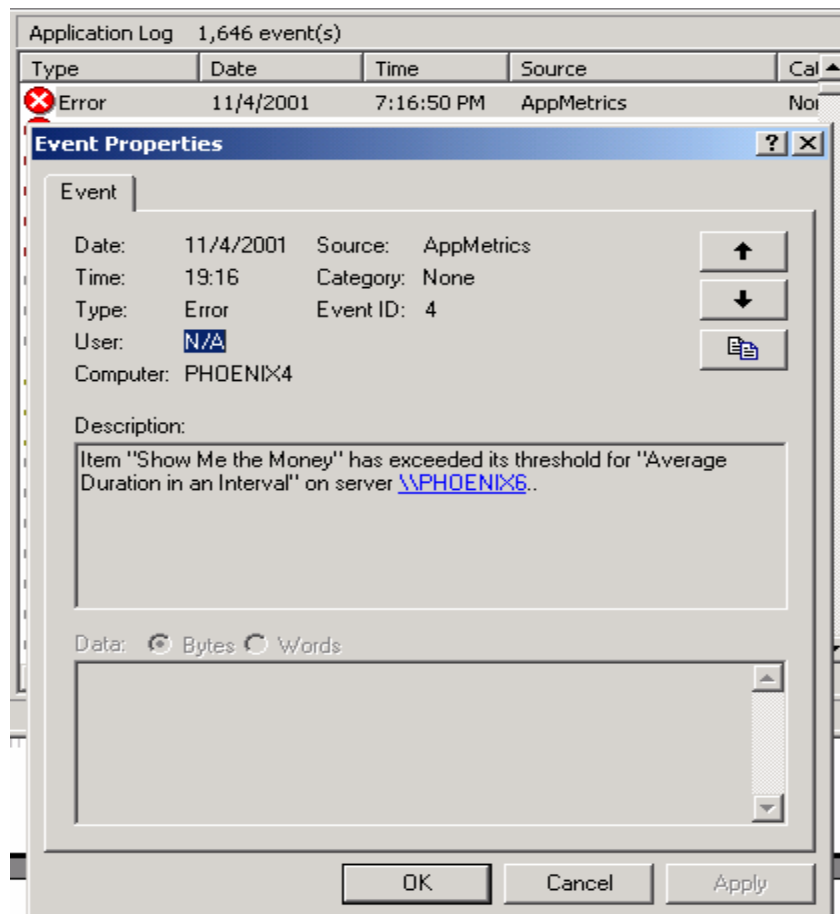


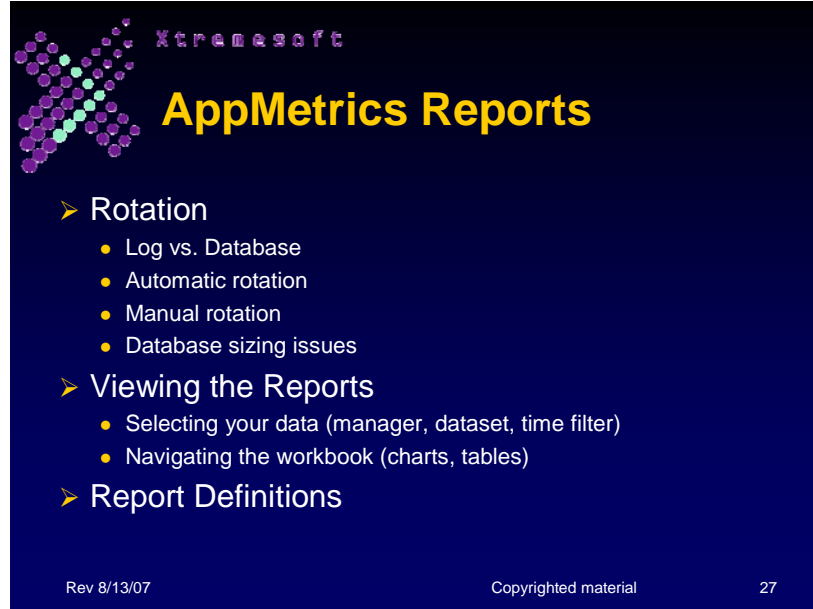
Figure 4



Review Questions

1. What is the one kind of **Warning** that **AppMetrics** will send via the **Notification** delivery mechanisms?
2. When would it make sense to enable the **AppMetrics** **Notification** log?
3. How would you shut off all **Notifications** without changing your benchmarks and thresholds or modifying your **Notification** delivery mechanisms?
4. Why do you think there are no notification mechanisms in a **Diagnostics Monitor**?

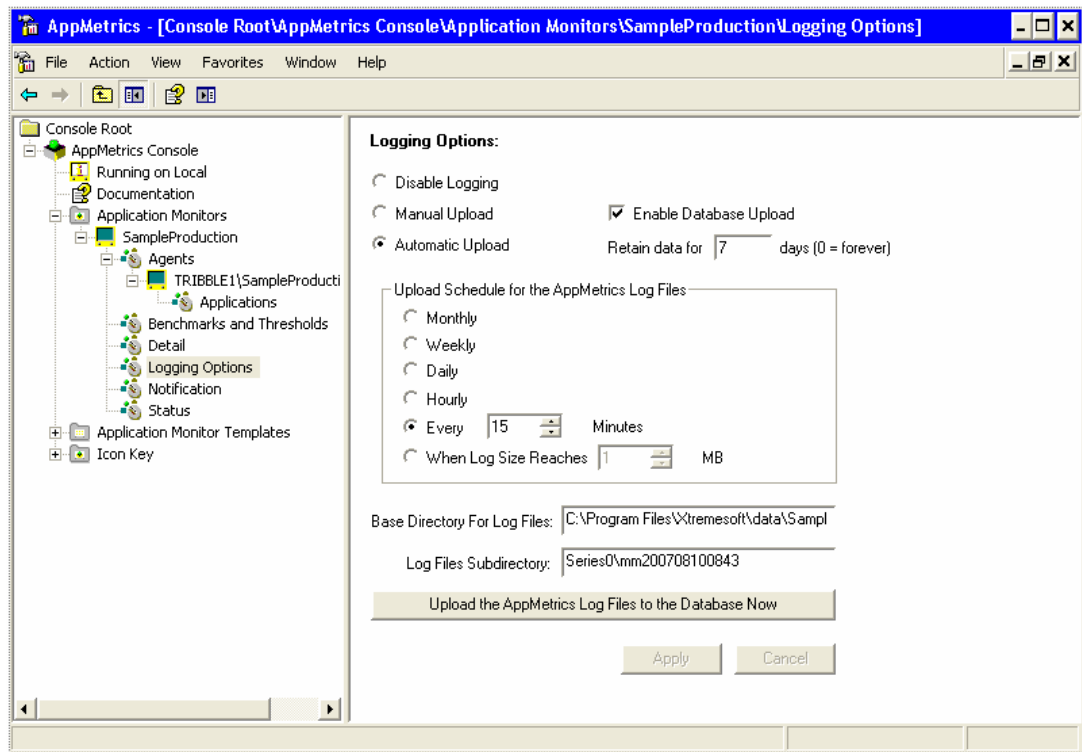
Section 12: AppMetrics Reports



The monitor produces metrics about the monitored packages/applications. It stores the metrics in various stages, first within log files, and then if you prefer, within database files. Once the metrics are stored in the database files, you can view them within the AppMetrics Reports.

This section reviews the process for storing the metrics in the log files and the database files. It then explains how you can view the metrics in the database through AppMetrics Reports.

The following illustration shows the Logging Options panel, where you can configure how the monitor manages the log files and the database files:



Rotation

As already mentioned, the monitor initially stores the generated metrics into a set of log files. These log files can grow to a large size over time. To manage the size of your log files, you can rotate or switch the output to an identical set of log files in a different subdirectory. This enables you to maintain the previously generated metrics in their current set of files while writing newly generated metrics to a fresh set of files.

The monitor can automatically rotate the log file subdirectory based on a timing interval or log file size. The monitor can also rotate the subdirectory as a result of direct user intervention.

Log vs. Database

The monitor saves any metrics that it generates at the end of each interval. It initially saves the metrics into a set of log files on the Manager machine. This simplifies the upload process of the data into the database.

If you want to view the saved metrics in historical reports, you must set up the monitor to upload the metrics to the database. As a result, the monitor will upload the data during each subdirectory rotation.

Note: An active Monitor will always record data every interval, even if the Agent is not reporting data to the Manager. In this case, the Monitor

would write a sequence of 0 (zero) values to the log files, with a valid interval number appended. When trying to diagnose the root cause of zero values in your data reports, it is therefore important to determine if the package was active at the time. An inactive package will report zeros to a Monitor. An active package with zeros in the data would indicate that a connection problem existed between the Manager and Agent machines during those intervals.

Automatic Rotation

You can set the automatic subdirectory rotation based on either a timing interval or the size of the log files.

Manual Rotation

When you set the Subdirectory Rotation to Manual, all metrics are always written to log files in the same subdirectory until a user manually rotates the subdirectory.

Database Sizing Issues

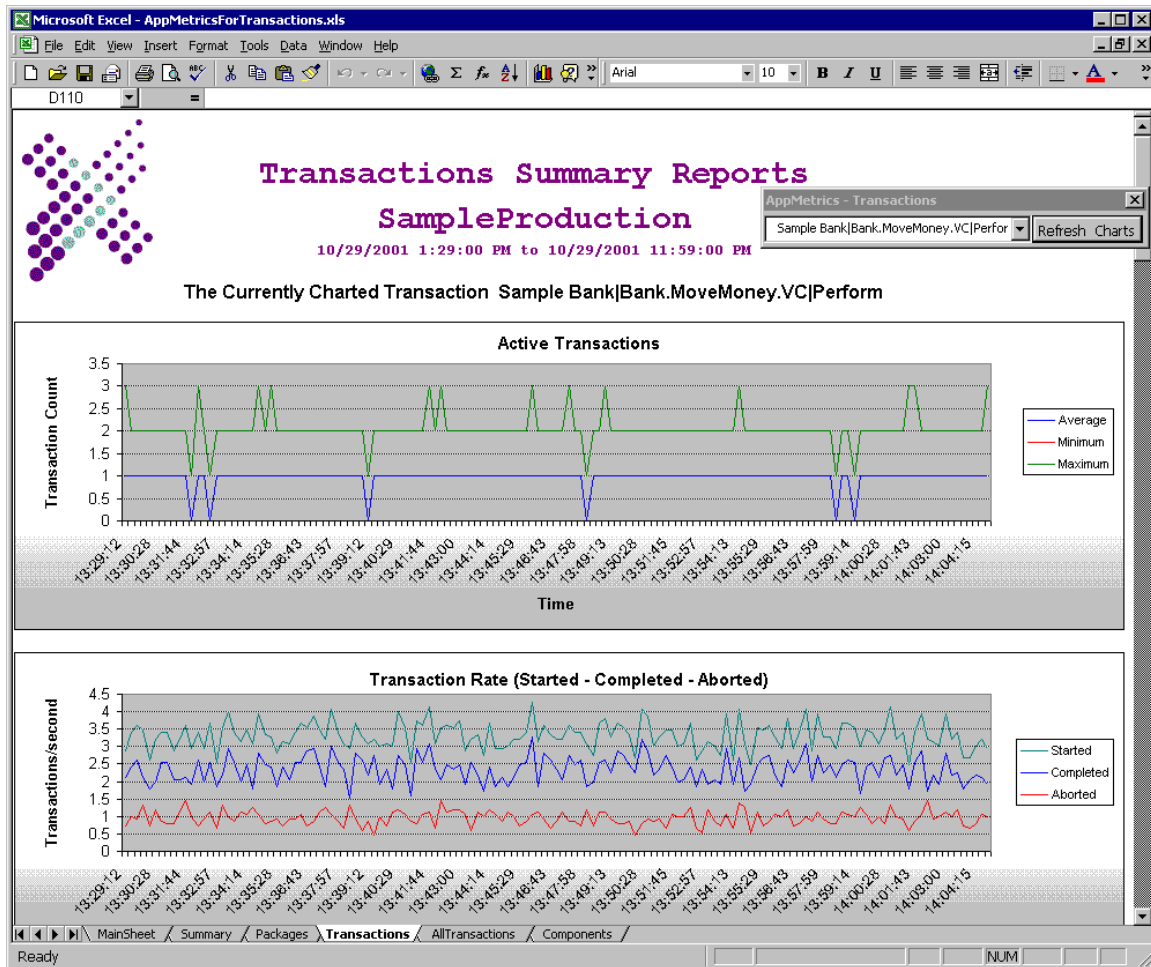
A monitor database uses an amount of disk space that is directly proportional to the number of applications and components being monitored. To more precisely calculate the amount of disk space needed for your organization, you must first know the number of components in each monitored package and the number of transactions that are made within them.

Xtremesoft provides a document that calculates the database size based on these two numbers and other parameters. You can request this document (**ATX Sizing Guidelines.doc**) from Xtremesoft Customer Support.

Viewing the Reports

After a monitor uploads metrics from its log files to its database, you can view the metrics in the AppMetrics Reports using Microsoft Excel. These reports provide a historical view into the performance of your packages/applications.

The following illustration shows an example of a report. It focuses on the historical activity of an individual transaction:



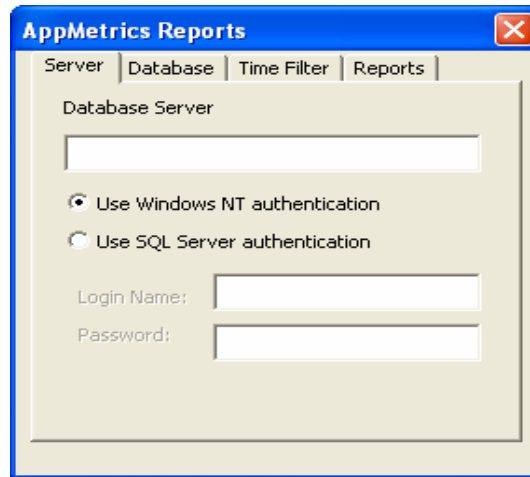
The following subsections describe how to select the data for a report.

Selecting your Data

The AppMetrics Reports lets you view historical data in a monitor. The reports deal with only one monitor at a time. The following subsections show the dialogs where you narrow down your selections for the data in a report. You must fill out each tab in consecutive order.

Manager Tab

The Manager tab is where you choose the machine that runs the monitor. To view reports for a monitor managed by the local machine, leave the field blank. To view reports for a monitor managed by another machine, specify the machine's name.

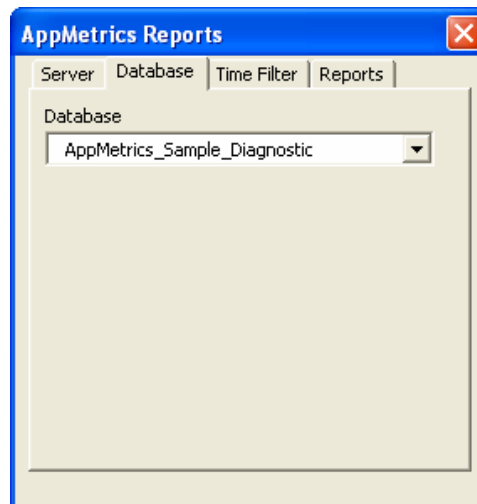


The screenshot shows the 'AppMetrics Reports' dialog box with the 'Manager' tab selected. The 'Database Server' field is empty. Below it, there are two radio buttons: 'Use Windows NT authentication' (selected) and 'Use SQL Server authentication'. At the bottom, there are two text boxes labeled 'Login Name:' and 'Password:'.

Dataset Tab

After specifying the machine name, you can then select the monitor whose reports you want to view. The monitors listed in the Monitor drop-down list have the following attributes:

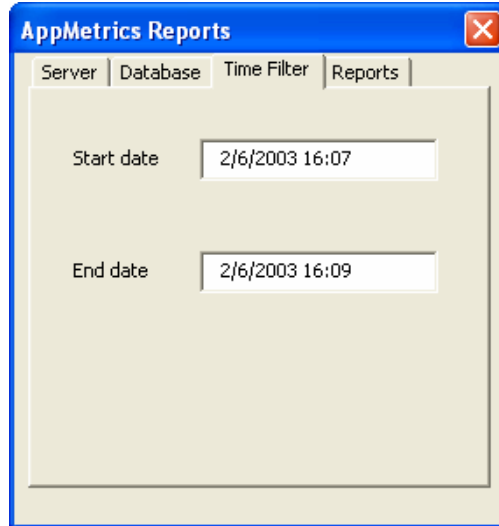
- They reside on the specified machine
- They were configured to upload data to the database



The screenshot shows the 'AppMetrics Reports' dialog box with the 'Dataset' tab selected. The 'Database' field is a dropdown menu showing 'AppMetrics_Sample_Diagnostic'.

Time Filter

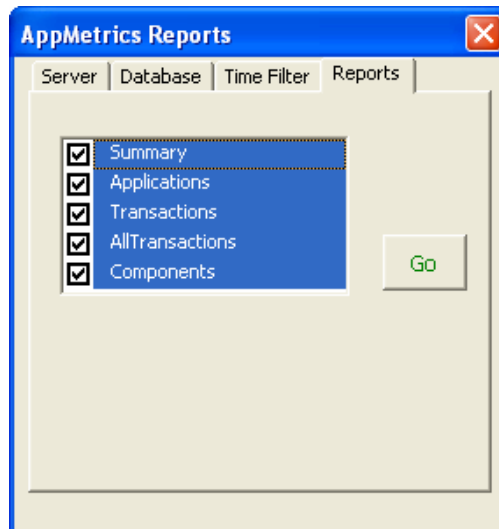
After choosing both the machine name and monitor, you can select the time period of the data to be viewed. The tab provides the available range of dates and times for the chosen monitor. You can change the time period as long as your dates and times fall within the available range.



Reports

After completing the information in the first three tabs, you can then select which reports to generate. The list of available reports in this tab depends on the type of monitor chosen in the Dataset tab.

Each selected report results in its own worksheet in the Excel window. By default, all the reports are selected. If desired, you can deselect the ones you do not want to generate.

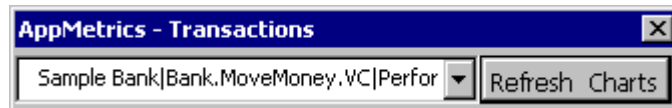


Navigating the workbook

After you select the data and generate the report, you can begin navigating through the worksheets, charts, and tables of the report. The AppMetrics Reports contains several worksheets, one for each report. The worksheets contain charts for the packages/applications, transactions, and components that have been monitored. Some worksheets also provide tables that list each measured item. For example, the Components report contains a table with all the individual components measured by the monitor.

In most of the worksheets, you will have to scroll down to see the additional charts and tables.

The worksheets that include a table also provide a floating dialog, where you can select the individual item to show in the graphs.



Production Reports Definitions

Summary Sheet	
Transaction:	See "Definitions" in Section 2 of this document.
Component:	See "Definitions" in Section 2 of this document.
Starts:	Total number of starts in the reported period. That is, the sum of the counts for all the intervals during the reported period.
Duration:	Total average transactional duration during the reported period (i.e. the sum of the average durations for all intervals during the selected reporting period).
Top Ten Transactions Bar Chart:	Top Ten transactions are those whose duration <i>and</i> frequency are higher than others. These are the transactions having greatest impact on your system. This impact is determined by summing the durations of each transaction and multiplying by the number of times it has been invoked. For example, a transaction that takes 5 ms but occurs 100 times during the reporting period will obviously be more impactful than a transaction that takes 50 ms but occurs only once during the reporting period.
Top Ten Components Bar Chart:	Top Ten components are those whose duration <i>and</i> frequency are higher than others. These are the components having greatest impact on your system. This impact is determined by summing the durations of each component and multiplying by the number of times it has been instantiated. For example, a component that takes 5 ms but is instantiated 100 times during the reporting period will obviously be more impactful than a component that takes 50 ms but is instantiated only once during the reporting period.
Key point:	For each chart, each individual number considered on its own is not very useful. What's important is the appearance of an item on one

	of these charts and its relative position to the other items. Improving the throughput for such items could represent the biggest "bang for buck."
--	--

Applications Sheet	
Package:	See "Definitions" in Section 2 of this document.
Charts:	Charts on this sheet apply to the specific package named in the worksheet header section, and which is selected from the AppMetrics Reports picklist component.
CPU Chart:	Displays average "% Processor Time" process performance counter for each interval in the reported period.
Memory Chart:	Displays average "Working Set" process performance counter (in KB) for each interval in the reported period. A rise can imply a memory leak.
Package Page Faults per Second Chart:	Displays the average number of "Page Faults / per Second" process performance counter for each interval in the reported period. Paging isn't always a bad thing. Typically, one starts with a different chart, such as Memory, and then looks at this one after already suspecting a problem.
Application Threads Chart:	Total Thread Count. The average number of "Threads Count" process performance counter for each interval in the reported period.
Percent CPU Tabular Data Sets:	Displays the average, minimum, and maximum "% Processor Time" process performance counter for each monitored Package during the reported period.
Memory Tabular Data Sets:	Displays the minimum and maximum "Working Set" process performance counter (in KB) for each monitored Package during the reported period. Note: average "Working Set" was excluded in order to make the report's width small enough for printing. A future release likely will work around this limitation in a different manner.
Package Page Faults per Second Tabular Data Sets:	Displays the average, minimum, and maximum "Page Faults / per Second" process performance counter during the reported period.
Application Threads Tabular Data Sets:	Displays the average, minimum, and maximum "Threads Count" process performance counter during the reported period.
Key Points:	<p>In-depth descriptions of process performance counters are available in Microsoft's NT Resource Kit documentation.</p> <p>If you are approaching this report without a particular problem area in mind, start with the tabular data. It will give you a higher-level sense of what occurred during the reported period. If a Package's numbers appear out of the expected range, specify the Package using the floating picker dialog, click refresh, and investigate how it performed over the period.</p>

Transactions Sheet	
Transaction:	See "Definitions" in Section 2 of this document.
Charts:	Charts on this sheet apply to the specific transaction named in the worksheet header section, and which is selected from the AppMetrics Reports picklist component.
Completion:	A Transaction has not completed until the lifecycle of the root method invoked has completed, or the component instance has gone out of scope (on Manager monitors with component-only level detail). Keep in mind that completions may not always appear in the same interval or reporting period in which the transaction, component instance or method was initiated.
Active Transactions Chart:	Displays the average, minimum, and maximum concurrently active counts for a transaction at each interval in the reporting period.
Transaction Rate (Started - Completed - Aborted) Chart:	Displays started, completed, and aborted counts for a transaction at each interval in the reporting period. In most cases, the Aborted count is the significant metric to analyze. In some peak periods, the other metrics can be interesting as well (for example, when transactions take longer than an interval to complete and start queuing).
Transaction Duration Chart:	Displays the minimum, maximum, and average duration for Transaction which occurred at each interval in the reported period. Look for spikes and increases during peak periods.
DTC Transaction time / User Transaction time Chart:	Xtremesoft doesn't recommend using these metrics in version 2.x. Version 3.0 will provide more meaningful metrics.
Active Tabular Data Sets:	Displays the minimum, maximum, and average concurrently active count for a transaction in the reporting period.
Total (selected period) Tabular Data Sets:	Displays the total started, completed, and aborted counts for a Transaction during the reported period. Contrasting numbers from one Transaction to another can often be very revealing. Also, observe the Aborted column. Scan down this column for Transactions that aborted during the reported period.
Duration (ms) Tabular Data Sets:	Displays the minimum, maximum, and average duration for a Transaction during the reported period. Contrasting numbers from one Transaction to another can often be very revealing.
Rate (per second) Tabular Data Sets:	Displays the started, completed, and aborted counts per Transaction <i>per second</i> during the reported period. This is often more useful during problem diagnosis, when the reported period is focused on just a few intervals.
Key Points:	If you are approaching this report without a particular problem area in mind, start with the tabular data. It will give you a higher-level sense of what occurred during the reported period.

All Transactions Sheet	
Transaction:	See "Definitions" in Section 2 of this document.
Charts:	Charts on this sheet apply to all transactions.
Active Transactions Chart:	Displays the total minimum, maximum, and average count for all transactions at each interval in the reported period.
Transaction Rate (Started - Completed - Aborted) Chart:	Displays the total started, completed, and aborted count for all transactions at each interval in the reported period.
Key Points:	This report displays <i>very high-level</i> information. It is useful when you're trying to get a sense of overall system activity, but if you're trying to diagnose a specific problem, you are better off examining the Transactions report.

Components Sheet	
Component:	See "Definitions" in Section 2 of this document.
Completion:	A Component has not completed until its instance has gone out of scope. Keep in mind that completions may not always appear in the same interval or reporting period in which the transaction, component instance or method was initiated.
Charts:	Charts on this sheet apply to the specific component named in the worksheet header section, and which is selected from the AppMetrics Reports picklist component.
Active Component Instances Chart:	Displays minimum, maximum and average counts of concurrent component instances that occurred at each interval in the reported period.
Component Rate (Started - Completed - Aborted) Chart:	Displays started, completed, and aborted counts for component instances at each interval in the reported period. In most cases, the Aborted count is the significant metric to analyze.
Component Duration Chart:	Displays the minimum, maximum, and average duration for component instances at each interval in the reported period. Look for spikes and increases during peak periods.
Active Tabular Data Sets:	Displays the minimum, maximum, and average concurrent component instances during the reported period.
Total (selected period) Tabular Data Sets:	Displays the total started, completed, and aborted counts for Component instances during the reported period. Contrasting numbers from one component to another can often be very revealing. Also, observe the Aborted column. Scan down this column for Components that aborted during the reported period.
Duration (ms) Tabular Data Sets:	Displays the minimum, maximum, and average duration for Component instances during the reported period. Contrasting numbers from one Component to another can often be very

	revealing.
Rate (per second) Tabular Data Sets:	Displays the started, completed, and aborted counts for Component instances <i>per second</i> during the reported period. This is often more useful during problem diagnosis, when the reported period is focused on just a few intervals.
Key Points:	If you are approaching this report without a particular problem area in mind, start with the tabular data. It will give you a higher-level sense of what occurred during the reported period.

Diagnostic Reports Definitions

Things to keep in mind:

- Diagnostics Monitors are not interval based.
- Diagnostics monitors log *large quantities* of data. So if you select a broad reporting period, it may take a while to generate a report. To a lesser degree the same may be said for Production monitors, but since they are interval-based reporting time and other impacts are minimized.

Summary Sheet	
Transaction:	See "Definitions" in Section 2 of this document.
Component:	See "Definitions" in Section 2 of this document.
Duration:	Total duration of an event during the reported period.
Top Ten Transactions Bar Chart:	Calculates the total duration over the reporting period for each transaction, and then charts the Top Ten.
Top Ten Components Bar Chart:	Calculates the total duration over the reporting period for each component instance, and then charts the Top Ten
Top Ten Methods Bar Chart:	Calculates the total duration over the reporting period for each method, and then charts the Top Ten.
Key point:	Like the "Top Ten" reports for Production Monitors, the actual numbers here are somewhat meaningless. What's important is appearance of an item on one of these charts and its relative position to the other items. Improving the throughput for such items could represent the biggest "bang for buck."

Transactions Sheet	
Transaction:	See "Definitions" in Section 2 of this document.
Completion:	A Transaction has not completed until the lifecycle of the root method invoked has completed, or the component instance has gone out of scope (on Manager monitors with component-only level detail). Keep in mind that completions may not always appear in the same interval or reporting period in which the transaction, component instance or method was initiated.

Charts:	Charts on this sheet apply to the specific transaction named in the worksheet header section, and which is selected from the AppMetrics Reports picklist component.
Transaction Duration Chart:	Displays the minimum, maximum, and average duration for a Transaction at each interval in the reported period. Look for spikes and increases during peak periods.
Transaction Completions Chart:	Displays completions for a Transaction in the reported period.
Durations (ms) Tabular Data Sets:	Displays the minimum, maximum, and average duration for a Transaction during the reported period. Contrasting numbers from one Transaction to another can often be very revealing.
Completed Tabular Data Sets:	Displays the total started, completed, and aborted counts for a Transaction during the reported period. Contrasting numbers from one Transaction to another can often be very revealing. Also, observe the Aborted column. Scan down this column for Transactions that aborted during the reported period.
Key Points:	If you are approaching this report without a particular problem area in mind, start with the tabular data. It will give you a higher-level sense of what occurred during the reported period.

All Transactions Sheet	
Transaction:	See "Definitions" in Section 2 of this document.
Charts:	Charts on this sheet apply to all transactions.
Transaction Duration Chart:	Displays the <i>total</i> minimum, maximum, and average count for all transactions during the reported period.
Transaction Completions Chart:	Charts the <i>total</i> started, completed, and aborted counts for all transactions during the reported period.
Key Points:	This report displays <i>very high-level</i> information. It is useful when you're trying to get a sense of overall system activity, but if you're trying to diagnose a specific problem, you're probably better off examining the Transactions report.

Components Sheet	
Component:	See "Definitions" in Section 2 of this document.
Completion:	A Component has not completed until its instance has gone out of scope. Keep in mind that completions may not always appear in the same interval or reporting period in which the transaction, component instance or method was initiated.
Charts:	Charts on this sheet apply to the specific component named in the worksheet header section, and which is selected from the AppMetrics Reports picklist component.

Component Duration Chart:	Charts the minimum, maximum, and average duration for a Component in the reported period. Look for spikes and increases during peak periods.
Component Completions Chart:	Charts completions for a Component in the reported period.
Durations (ms) Tabular Data Sets:	Displays the minimum, maximum, and average duration for a component during the reported period. Contrasting numbers from one component to another can often be very revealing.
Completed Tabular Data Sets:	Displays the Total, Successful, and Exceptions counts for a component during the reported period. Contrasting numbers from one component to another can often be very revealing. Note that the <i>Exception</i> column corresponds not to Aborts, but to components where methods generated exceptions (or non-zero return status)
Key Points:	If you are approaching this report without a particular problem area in mind, start with the tabular data. It will give you a higher-level sense of what occurred during the reported period.

Methods Sheet	
Method:	See "Definitions" in Section 2 of this document.
Completion:	A Method has not completed until its lifecycle has ended. Keep in mind that completions may not always appear in the same interval or reporting period in which the transaction, component instance or method was initiated.
Method Duration Chart:	Charts the minimum, maximum, and average duration for a Method at each interval in the reported period. Look for spikes and increases during peak periods.
Method Calls (Completions) Chart:	Charts completions for a Method in the reported period.
Durations (ms) Tabular Data Sets:	Displays the minimum, maximum, and average duration for a Method during the reported period. Contrasting numbers from one Method to another can often be very revealing.
Completed Tabular Data Sets:	Displays the Total, Successful, and Exception counts for a Method during the reported period. Contrasting numbers from one Method to another can often be very revealing. Note that the <i>Exception</i> column corresponds not to Aborts, but to components where methods generated exceptions (or non-zero return status).
Key Points:	If you are approaching this report without a particular problem area in mind, start with the tabular data. It will give you a higher-level sense of what occurred during the reported period.



Section 13: Lab 5 - Viewing and Analyzing AppMetrics Reports

Objectives

- **Configure your Monitor rotation schedules**
- **View a Production Monitor report**
- **View a Diagnostics Monitor report**
- **Comparing Component-only detail to Component-Method detail in reports**

Prerequisites

- At least five minutes of active monitoring of the *Sample Bank* application by an AppMetrics Production Monitor
- At least five minutes of active monitoring of the *Sample Bank* application by an AppMetrics Diagnostics Monitor

Preparation

Students should perform this Lab individually or as a team on an AppMetrics Manager machine console, or on an AppMetrics Console/Reports client.



Students should ensure that the Macro security level setting in Excel on the AppMetrics Manager machine - and any AppMetrics Console/Reports clients you are using – is set to **Medium** or **Low**.

Check this setting by selecting **Macro -> Security** from the **Tools** menu in Excel.

Step 1: Configuring Monitor Rotation Schedules

1. An application monitor generates data and initially stores this data in a set of files called the Logs. The metrics are directed into separate log files for easy upload into databases.



Any application monitor can be the source for reports, but the monitor must first be configured to upload data from its logs to a SQL database for AppMetrics to report against – a process called ‘rotating’ the data.

Once a monitor is configured to rotate its data, and the first rotation is initiated, AppMetrics creates a database named after the monitor. It will then upload the data from the logs into the database at a

specified **Rotation Frequency**, or when the log file reaches a certain size.

Once data is uploaded from the logs to the database, you can create and view a set of reports in AppMetrics Reports based on the data. To view this data in a manageable and meaningful way, AppMetrics Reports displays the data in various Excel charts and tables.

2. All of the settings for the rotation of a Monitor's logs are found in the **Logging Options** configuration panel.

The format and name of this configuration panel is exactly the same in both Production and 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.

To view and configure the Logging Options for a Monitor, select the **Logging Options** icon from the Monitor's tree list (see Figure 1).

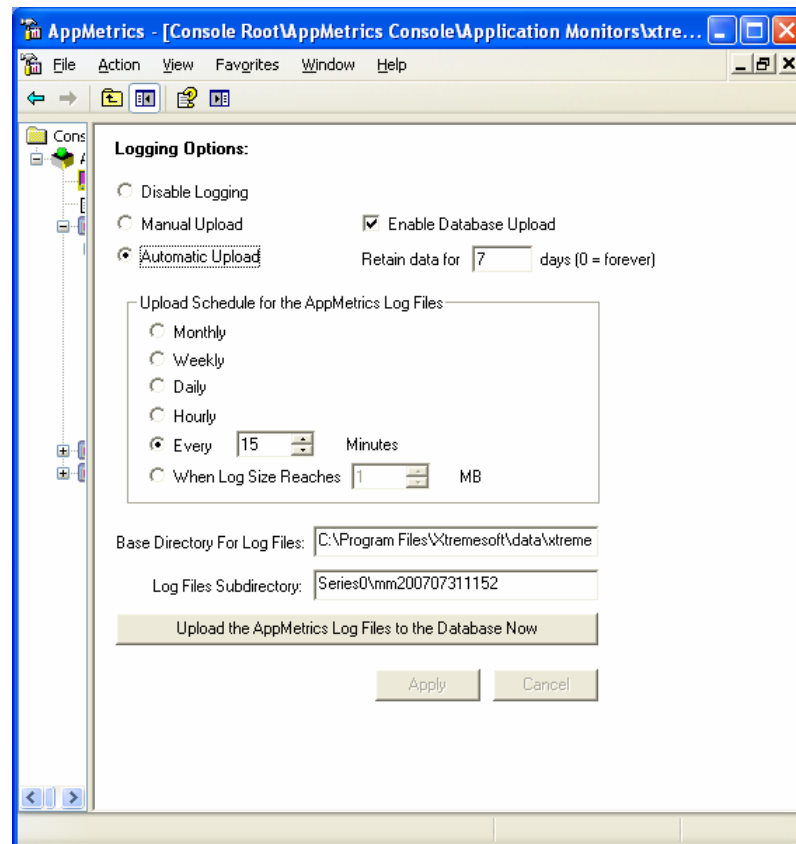


Figure 1

3. By default a Monitor is set for daily automatic rotation, and the database is set to keep 7 days worth of data.

Since AppMetrics requires that your SQL databases are stored on the AppMetrics Manager machine, rotation frequency is not usually

decided based on local machine resources. More likely you will make this decision based upon how often you plan to run reports against the database.

For this Lab we will leave the **Rotation Frequency** set to **Automatic** and **Daily** (which is what most users select), because we will be performing our rotations manually.

When deciding how many days of data to keep for your Monitors, the determining factors are the length of time you wish to trend against, versus the query speed and database size you can live with. The more trending you plan to do, the bigger the database and the slower the queries.

For this Lab we will leave the **Keep Data for ___ Days** at its default value of **7** days.

4. Before we begin trying to view AppMetrics Reports we must ensure that the data that the Monitors have been collecting in their log files are uploaded to the SQL database. Until at least one rotation has occurred there is no data in the database to report against. Consider this step 'priming the pump' for the AppMetrics Reports.

Unlike other configuration steps in the Logging Options panel, you can manually rotate your logs even while the Monitor is running.

Manually rotate your logs by clicking the **Upload the AppMetrics Log Files to the Database Now** button.

Note: The series number in the **Log Files Subdirectory** field increments with each manual rotation. If you are confused as to whether you have had an initial rotation of your logs, check the series number and ensure that it is greater than 0 (zero) (see Figure 2).

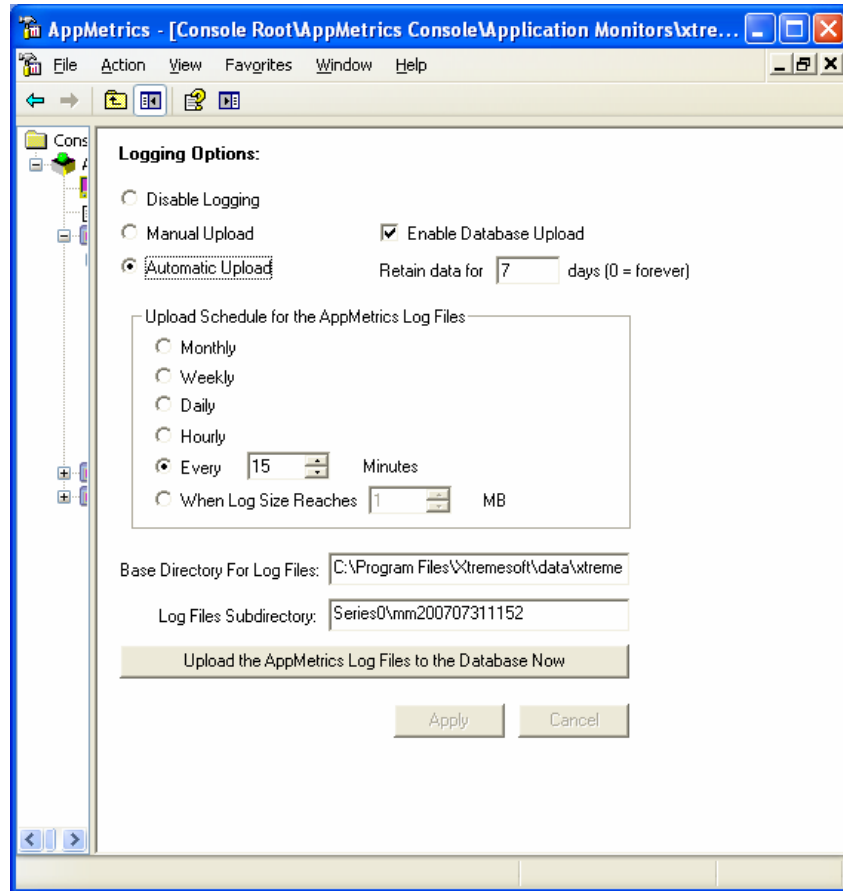


Figure 2

Step 2: View a Production Monitor Report

1. AppMetrics for Transactions provides the AppMetrics Reports tool for use in viewing and analyzing your Monitor's data.

AppMetrics Reports are viewed and managed via a macro-enabled Excel workbook installed with any AppMetrics Manager and Console/Reports client installation.

If you wish, Monitor data can be independently viewed and analyzed by any reporting tools that can read in formatted log files or SQL Server databases. See the AppMetrics for Transactions documentation for information on AppMetrics Monitor data schemas and alternative analysis options.

2. Start the AppMetrics Reports by selecting **AppMetrics for Transactions -> AppMetrics Reports** from the Xtremesoft program group in your Windows Programs menu.

If prompted select the **Enable Macros** option.

3. The AppMetrics Reports dialog appears when you start AppMetrics Reports. The AppMetrics Reports dialog enables you to create a specific set of reports. It contains the tabs where you choose the Manager (machine name), Dataset (Monitor name), Time Filter (time period of report), and report set (specific report components). Once you enter the information in this dialog, you can then create the reports.



Note: You should complete each dialog tab in consecutive order, starting with the Manager tab before proceeding to the next. The following four subsections describe each tab.

The AppMetrics Reports dialog disappears after you create a report. To reopen the dialog at any time, right-click any non-chart cell, and then choose **AppMetrics** in the popup menu.

4. First input the Sql Server instance name of your team's AppMetrics Manager in the **AppMetrics Manager Machine** field of the **Manager** tab (see Figure 3).

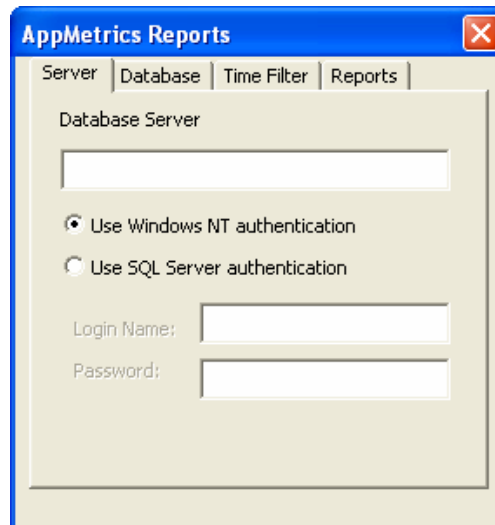


Figure 3

5. Next, click on the **Dataset** tab.

Select the name of your Production Monitor from the drop-down list (see Figure 4).

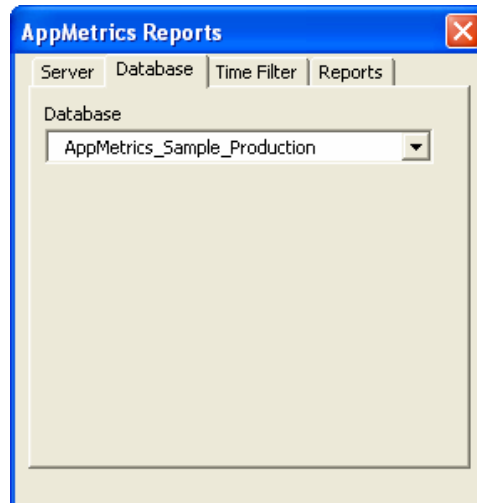


Figure 4

6. Click on the **Time Filter** tab to select the time period of the data to be viewed. This tab provides dates and times that are available (i.e. data was generated and uploaded to the database) for the monitor chosen in the Dataset tab (see Figure 5).

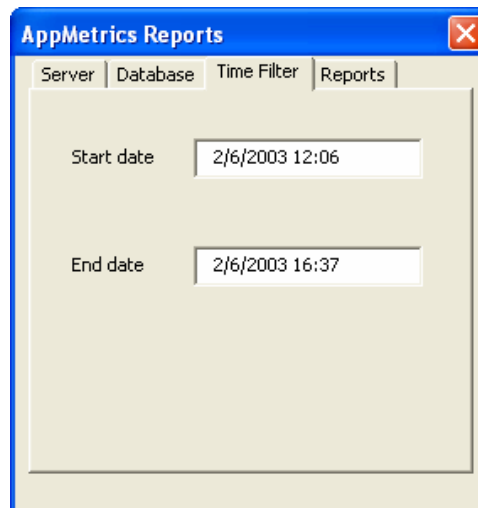


Figure 5

In this Lab we have only one day of data uploaded so you should be able to select the defaults in this tab.

7. Click on the **Reports** tab, and select the specific report components you wish to see generated (see Figure 6).

Each report results in its own worksheet in the Excel window. By default, all the reports are selected. If desired, you can deselect the

ones you do not want to generate. The list of available reports in this tab depends on the monitor type chosen in the Dataset tab.

In this lab we will accept the default and generate all available reports.

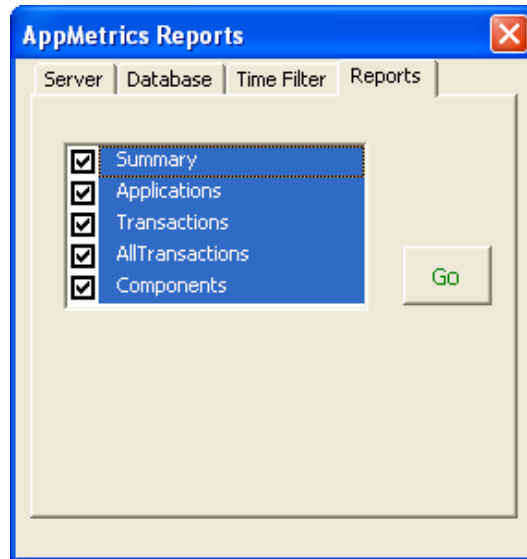


Figure 6

Click the **Go** button.

8. In all of the individual AppMetrics Reports' reports there are four distinct areas to observe.

The upper portion of the report is devoted to the charting of the data your selected AppMetrics Reports dialog (see Figure 7 Callout 'A').

Applications, Transactions, and Components reports can chart and report on individual objects. In these worksheets a floating dialog box enables you to choose which individual object type to show in the charts. Once you select a specific object type from the drop-down list, you can generate new charts by selecting the **Refresh Charts** button (see Figure 7 Callout 'B').

Beneath the last chart, most reports also contain a table that lists the individual objects measured by that monitor. For each object measured during the time period of the report, the table provides an appropriately categorized breakdown of the average, minimum, and maximum values (see Figure 7 Callout 'C').

Lastly, the actual data values used by those charts are placed in the worksheet to the right of the charts (see Figure 7 Callout 'D').

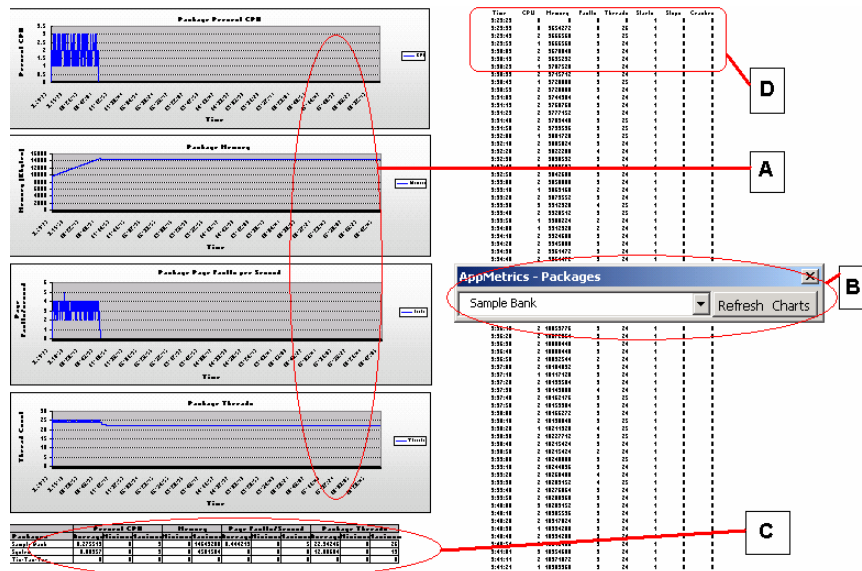


Figure 7

9. The first report to view is the **Summary Report** (see Figure 8).

The Summary Report shows which items consume the most resources on the system. In particular, it displays charts of the top 10 transactions and components. In these charts, AppMetrics calculates the number of starts multiplied by the average duration.

Note that our named transaction **Show Me the Money** made it into the top ten.

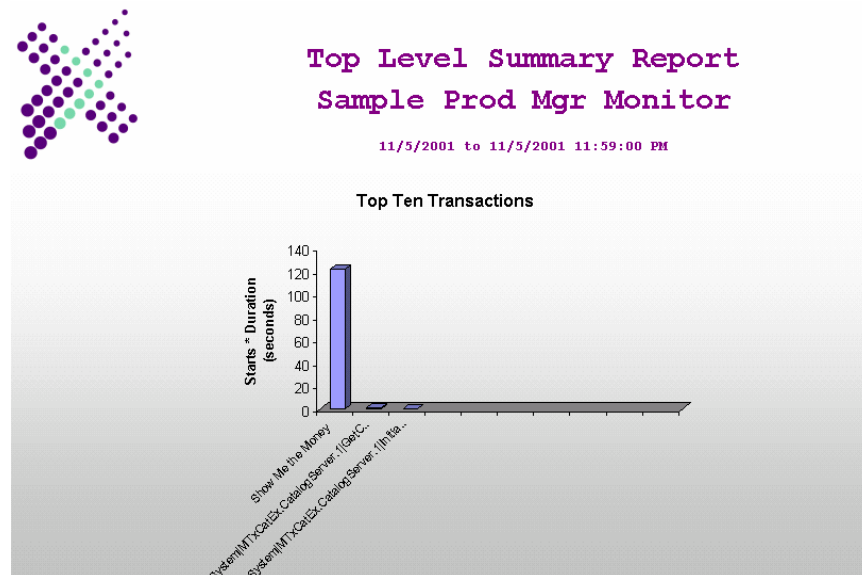


Figure 8

10. The next report to view is the **Packages/Applications Report** (see Figure 9).

The Packages/Applications Report has four charts:



- **Application Percent CPU** – Shows the portion of the CPU used by the application during each interval. It can help determine if the currently charted application is using an excessive amount of CPU.
- **Application Memory** – Shows how much memory was being used by the application during each successive interval. A rising chart may indicate the application has a memory leak.
- **Application Page faults per Second** – Shows the rate of page faults. An excessive rate may result from low amounts of RAM on the server.
- **Application Threads** – Shows how many threads were used by the application during each interval.

Beneath the last chart, the report also includes a table that lists the individual applications measured by the monitor. For each individual application measured during the time period of the report, the table provides the average, minimum and maximum values.

Note: Any value below one millisecond is displayed as a zero value in the table.

Click the drop-down arrow, select the **Sample Bank** application, and then click the Refresh Charts button.

Click on any of the spikes, and you will note that the actual data value will be displayed.

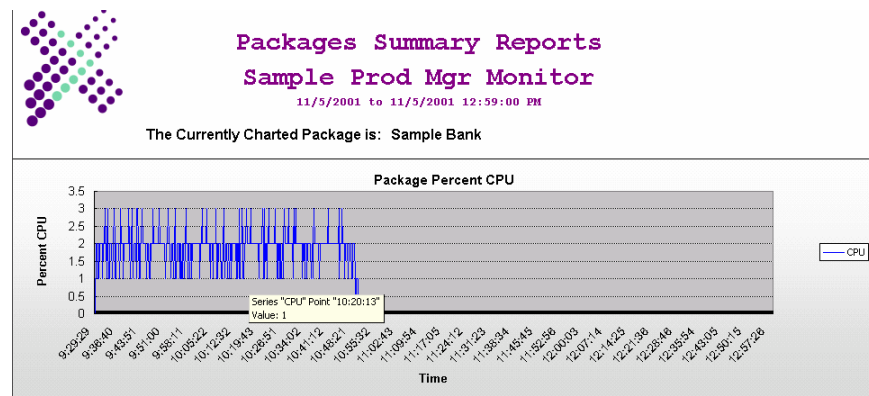


Figure 9

11. The next report to view is the **Transactions Report** (see Figure 10).

The Transactions Report for Production monitors has four charts:

- **Active Transactions**
- **Transaction Rate (Started - Completed - Aborted)**
- **Transaction Duration**
- **DTC Transaction time / User Transaction time**

You can use the charts to determine which transaction types are aborting or taking too long to complete.

Beneath the last chart, the report also contains a table that lists the individual transaction types measured by the monitor. For each individual transaction type measured during the time period of the report, the table provides the average, minimum, and maximum values.

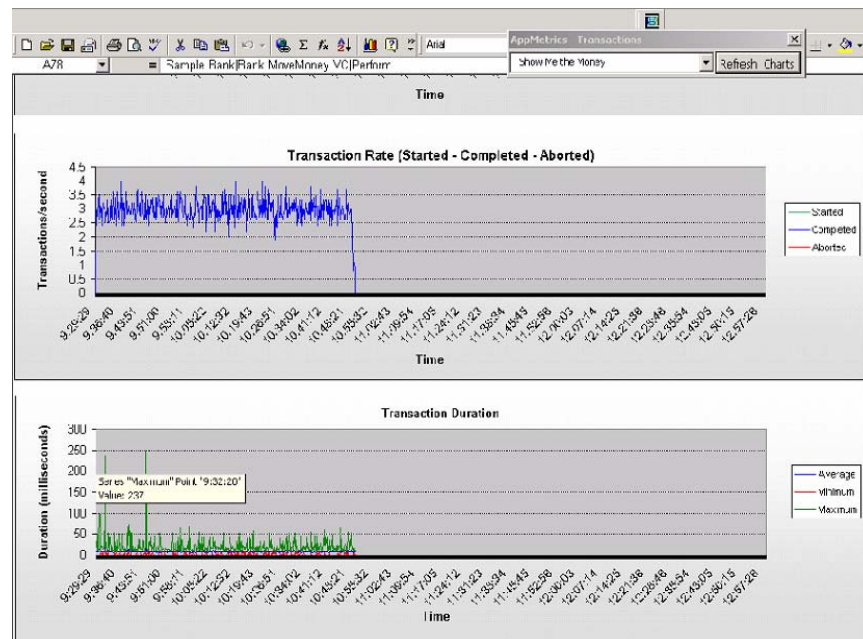


Figure 10

12. The next report is the **All Transactions Report** (see Figure 11).

The All Transaction Report for Production monitors contains the following charts:

- **Active Transactions**
- **Transactions (Starts - Completes - Aborts)**

These are the same as the first two charts in the Transactions report, but the data within them are aggregates of all the transactions.

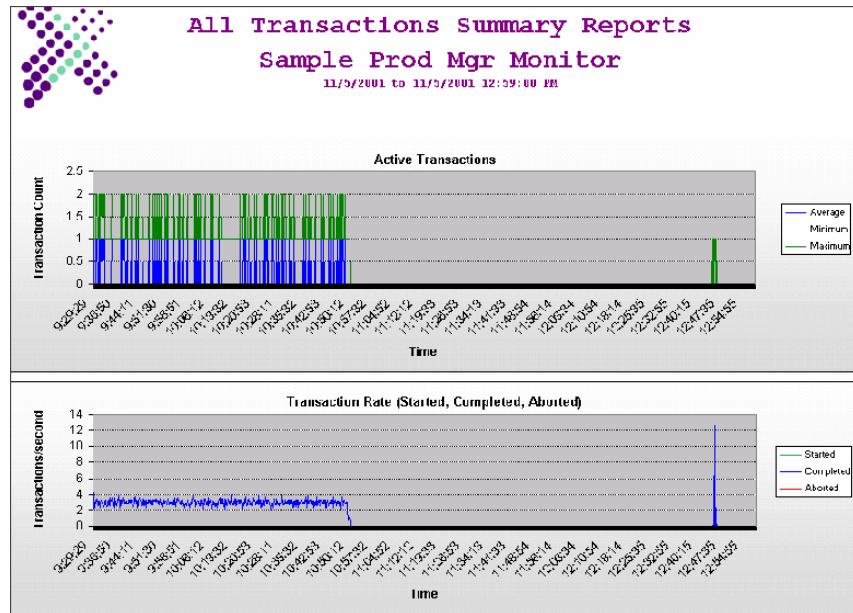


Figure 11

13. The last report to view in our Production Monitor is the **Components Report** (see Figure 12).

The Components Report for a Production monitor has three charts:

- **Active Component Instances**
- **Components (Starts - Completes - Aborts)**
- **Component Duration**

You can use the charts to determine which component types are aborting or taking too long to complete. A large number of active components can show where resources are being consumed.

Beneath the last chart, the report also contains a table that lists the individual component types measured by the monitor. For each individual component type measured during the time period of the report, the table provides the average, minimum, and maximum values.

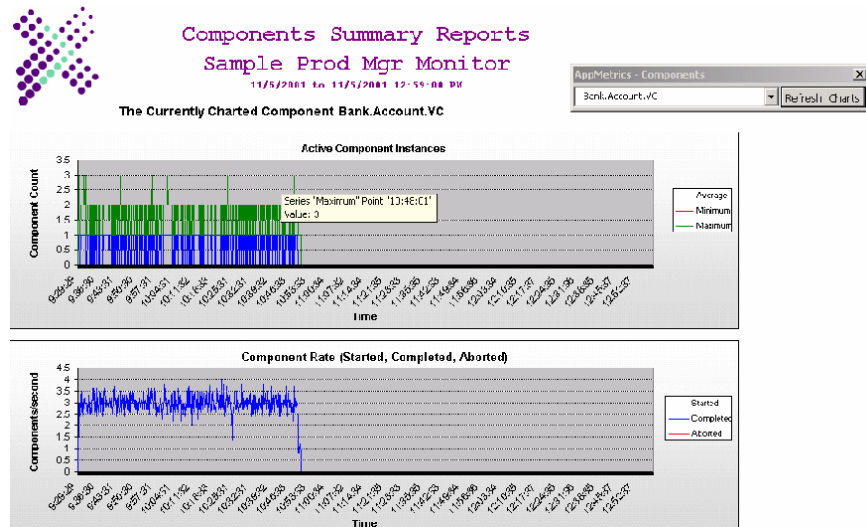


Figure 12

Step 3: View a Diagnostics Monitor Report

1. An AppMetrics for Transactions Diagnostics Monitor report is started and configured in the same manner as Production Monitor reports. The difference is in what types of data that the Diagnostics Monitor provides.

Start the AppMetrics Reports by selecting **AppMetrics for Transactions** → **AppMetrics Reports** from the Xtremesoft program group in your Windows Programs menu.

If prompted select the **Enable Macros** option.

2. The AppMetrics Reports dialog appears when you start AppMetrics Reports. The AppMetrics Reports dialog enables you to create a specific set of reports. It contains the tabs where you choose the Manager (machine name), Dataset (Monitor name), Time Filter (time period of report), and report set (specific report components). Once you enter the information in this dialog, you can then create the reports.



Note: You should complete each dialog tab in consecutive order, starting with the Manager tab before proceeding to the next. The following four subsections describe each tab.

The AppMetrics Reports dialog disappears after you create a report. To reopen the dialog at any time, right-click any non-chart cell, and then choose **AppMetrics** in the popup menu.

3. First input the machine name of your team's AppMetrics Manager in the **AppMetrics Manager Machine** field of the **Manager** tab (see Figure 13).

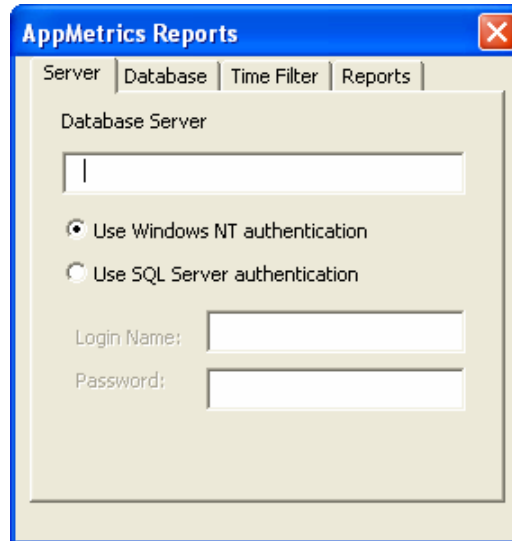


Figure 13

4. Next, click on the **Dataset** tab.

Select the name of your Diagnostics Monitor from the drop-down list (see Figure 14).

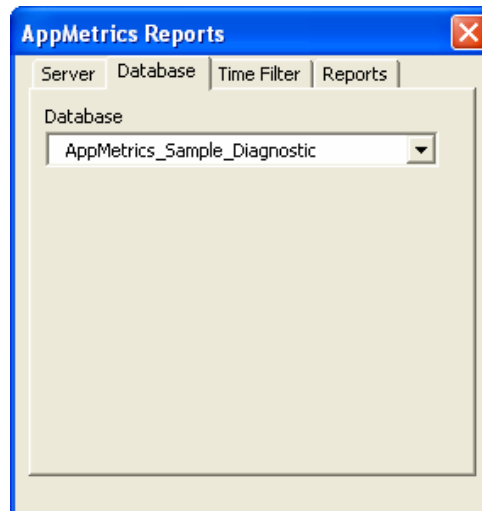


Figure 14

5. Click on the **Time Filter** tab to select the time period of the data to be viewed. This tab provides dates and times that are available (i.e. data was generated and uploaded to the database) for the monitor chosen in the Dataset tab (see Figure 15).

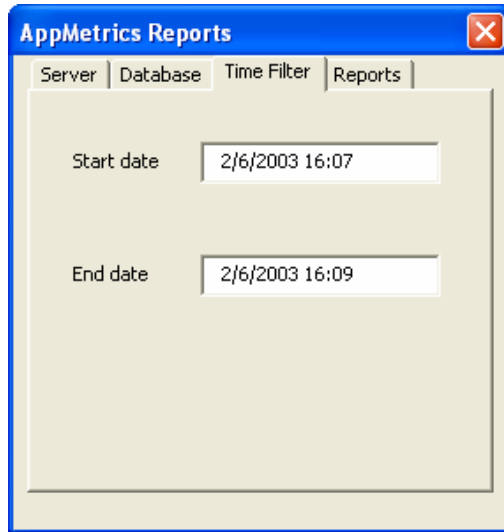


Figure 15

In this Lab we have only one day of data uploaded so you should be able to select the defaults in this tab.

6. Click on the **Reports** tab, and select the specific report components you wish to see generated (see Figure 16).

Each report results in its own worksheet in the Excel window. By default, all the reports are selected. If desired, you can deselect the ones you do not want to generate. The list of available reports in this tab depends on the monitor type chosen in the Dataset tab.

In this lab we will accept the default and generate all available reports.

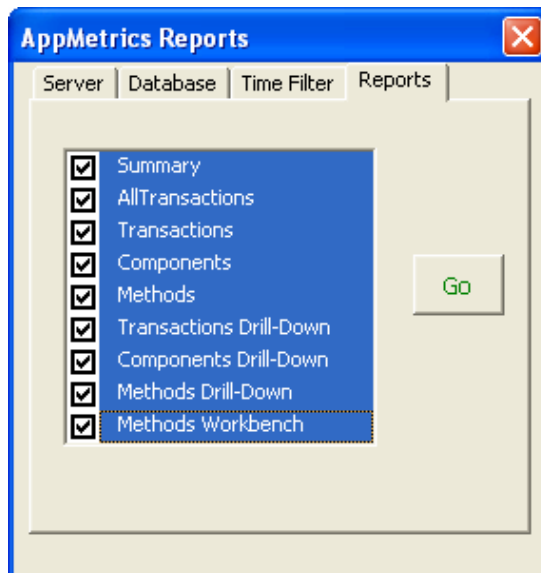


Figure 16

Click the **Go** button.

- The first thing you should note is that a Diagnostics Monitor has a different set of reports that it can generate.

Unlike a Production Monitor, the Diagnostics Monitor reports do *not* have a **Packages/Applications** Report. It does however have a **Methods** Report, which is absent in the Production Monitor reports.

- The first report to view is the **Summary Report** (see Figure 17).

The Summary Report shows which items consume the most resources on the system. In particular, it displays charts of the top 10 transactions, components, *and* methods. In these charts, AppMetrics calculates the number of starts multiplied by the average duration.

Note: A Diagnostic Monitor does not incorporate the functionality of a Named Transaction, so our named transaction **Show Me the Money** does not appear in the chart.

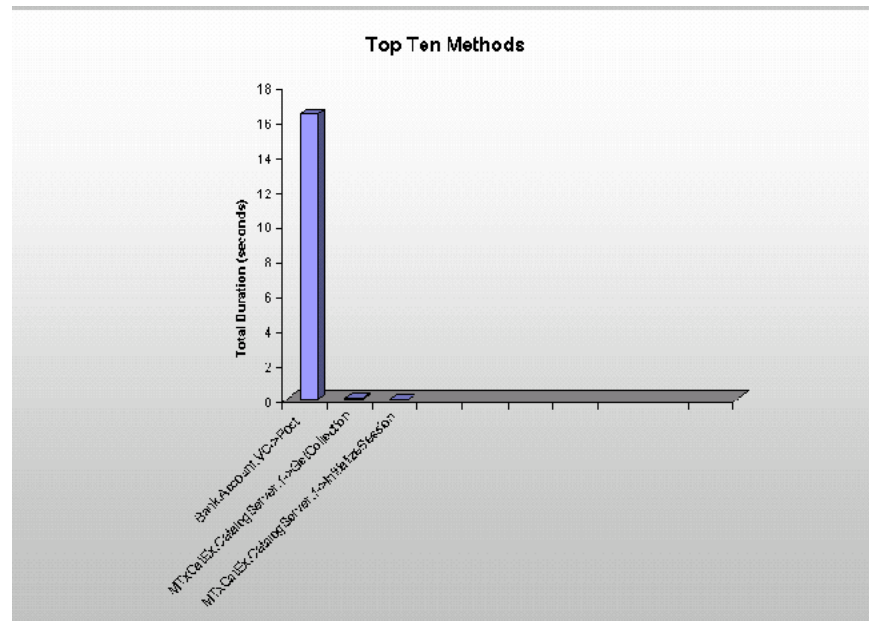


Figure 17

- The next report to view is the **Transactions Report** (see Figure 18).

The Transactions Report for Diagnostic monitors has two charts:

- **Active Transactions**
- **Transaction Duration**

You can use the charts to determine which transaction types are aborting or taking too long to complete.

Beneath the last chart, the report also contains a table that lists the individual transaction types measured by the monitor. For each individual transaction type measured during the time period of the

report, the table provides the average, minimum, and maximum values.

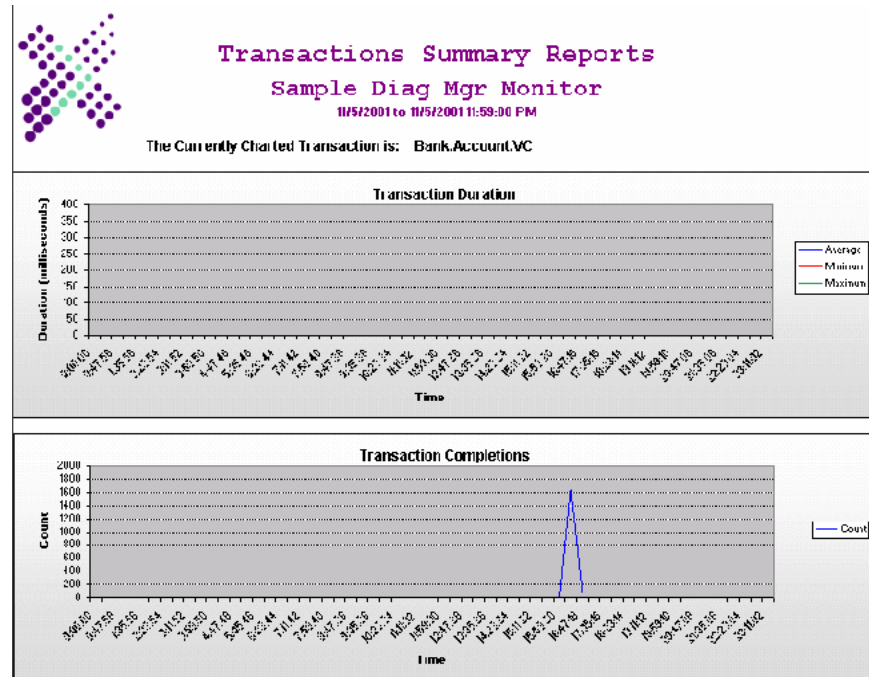


Figure 18

10. The next report to view is the **All Transactions Report** (see Figure 19).

The All Transaction Report for Diagnostic monitors contains the following charts:

- **Transaction Durations**
- **Transaction Completions**

These are the same as the two charts in the Transactions report, but the data within them are aggregates of all the transactions.

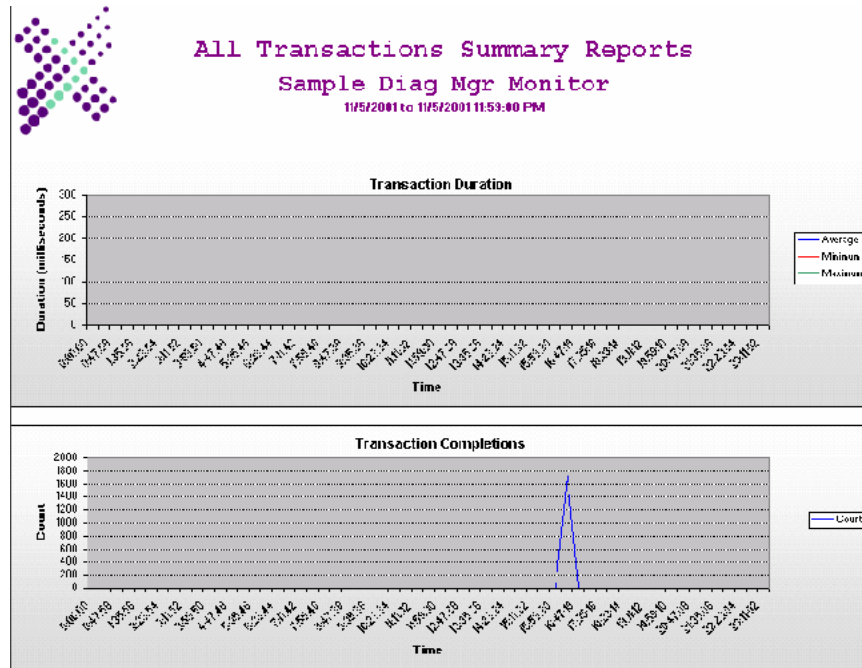


Figure 19

11. The next report to view in our Diagnostics Monitor is the **Components Report** (see Figure 20).

The Components Report for a Diagnostics monitor has two charts:

- **Component Duration**
- **Component Completions**

You can use the charts to determine if the currently charted component is taking too long to complete.

Beneath the last chart, the report also contains a table that lists the individual component types measured by the monitor. For each individual component monitored during the time period of the report, the table provides the average, minimum, and maximum values for duration and completions.

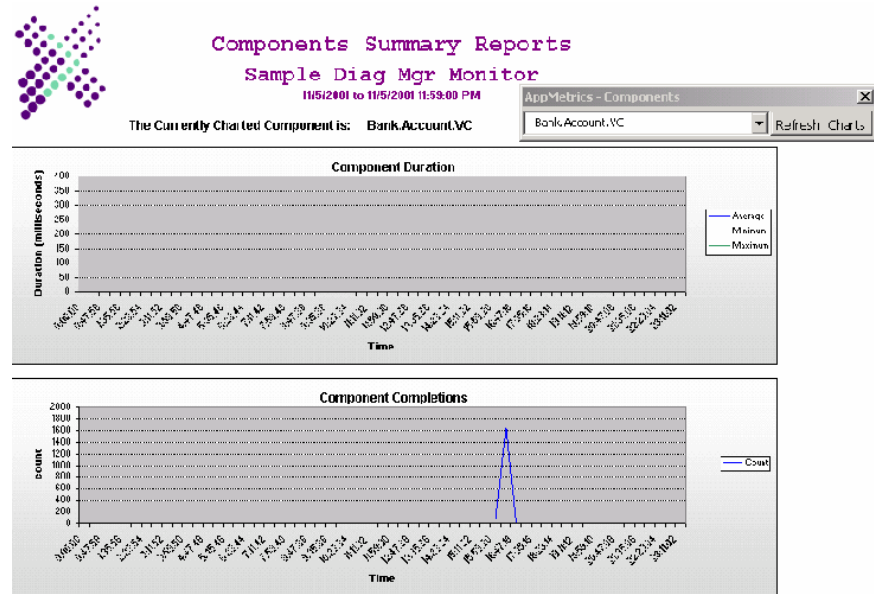


Figure 20

12. The last report to view in our Diagnostics Monitor is the **Methods Report** (see Figure 21).

Note: You will only get a Methods Report if you have selected **Method Call Detail** in the **Applications and Detail** configuration panel of your Diagnostics Monitor.

The Methods Report contains two charts:

- **Method Duration** - shows the average, minimum, and maximum lengths of time for the calls of the currently charted method.
- **Method Calls** - shows how many times the currently charted method was called during each interval.

The charts can provide an indication of which methods types may be running slowly.

Beneath the last chart, the report also contains a table that lists the individual method types measured by the monitor. For each individual method type that was called and monitored during the time period of the report, the table provides the average minimum, and maximum values for duration and completions.

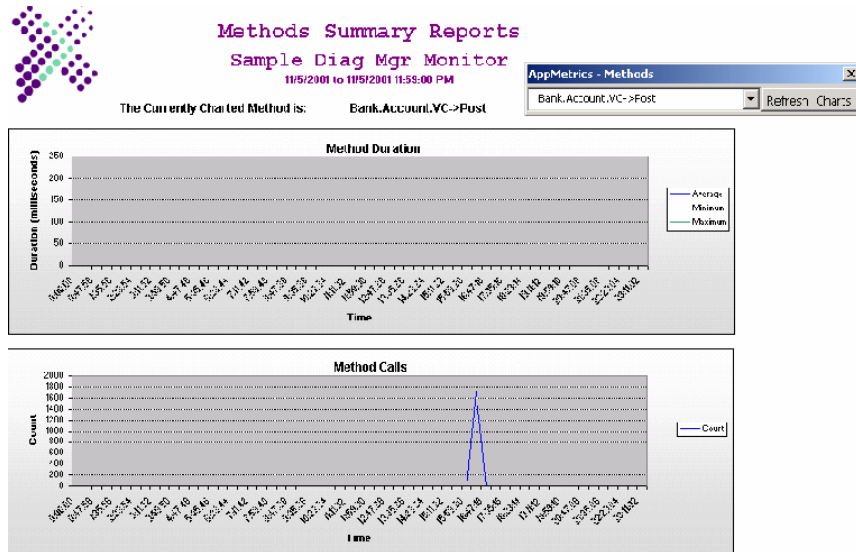


Figure 21

Comparing Component & Method level detail to Component-only level detail in AppMetrics Reports

Component-Only. Changing the detail level of a Production Monitor in AppMetrics changes the nature of the data that AppMetrics stores in its log files and database, and ultimately the data you see in the reports. If you have selected Component-only level detail then the transactions charted and listed in the table below the charts of the Transactions worksheet will reflect that detail level.

Look at the snapshot below of a Production Monitor transactions worksheet. You will note that the transactions are named by their root component, in this instance ‘**Sample Bank | Bank Account.VC**’. This is an example of Component-only level detail in an AppMetrics Production report.

Transactions	Average	Active Minimum Maximum	Total (select Started Compl)
IFMStocks 2000 Core FMStocks_Bus.Account	0.004484	0 2	14027 1.
IFMStocks 2000 Core FMStocks_Bus.Broker	0	0 1	3502 :
IFMStocks 2000 Core FMStocks_Bus.Ticker	0	0 1	3505 :
Sample Bank Bank.Account.VC	1.22072	0 5	128979 12
Sample Bank Bank.MoveMoney.VC	1.252252	0 5	128078 12

Look at the snapshot below of a Production Monitor transactions worksheet. You will note that the transactions are named by their root method, in this instance **‘Sample Bank | Bank Account.VC | Post’**. This is an example of Component& Method level detail in an AppMetrics Production report.

Transactions	Active	
	Average	Minimum
/fmstocks/accountssummary.asp FMStocks 2000 Core FMStocks_Bus.Account	0	0
/fmstocks/buystock.asp FMStocks 2000 Core FMStocks_Bus.Ticker	0	0
/fmstocks/default.asp FMStocks 2000 Core FMStocks_Bus.Account	0	0
/fmstocks/portfolio.asp FMStocks 2000 Core FMStocks_Bus.Account	0	0
/fmstocks/sellstock.asp FMStocks 2000 Core FMStocks_Bus.Account	0	0
/fmstocks/sellstockaction.asp FMStocks 2000 Core FMStocks_Bus.Broker	0	0
Sample Bank Bank.Account.VC Post	1.146396	0
Sample Bank Bank.MoveMoney.VC Perform	1.182432	0



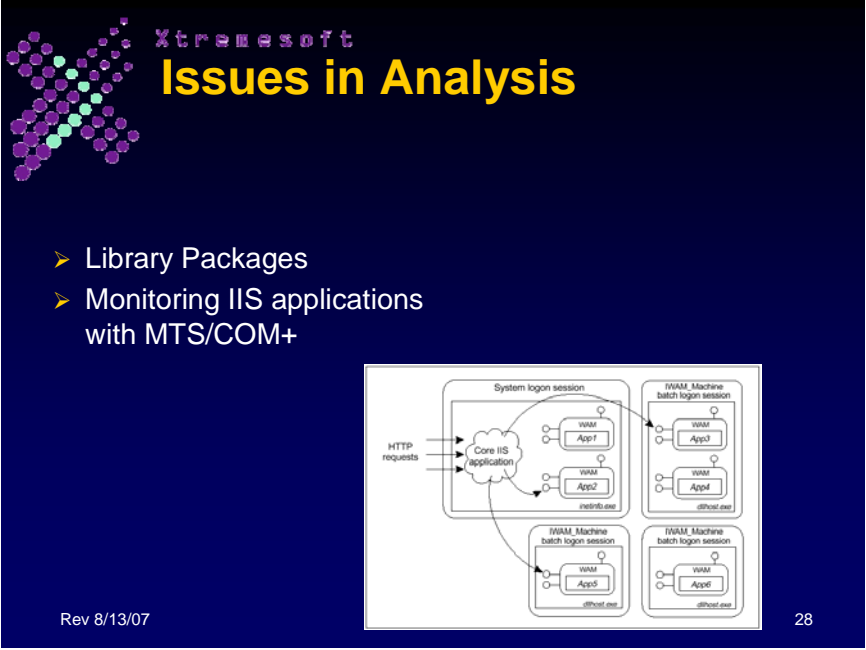
Note: When deciding to initially configure or change a Production Monitor’s detail level, keep in mind who will be reading the reports and what level of detail they will need to analyze the data. Remember that an AppMetrics report only reflects the level of detail written to the database.



Review Questions

1. Where in a report would you look for averages of the data charted?
2. Why do you suppose that the Production Monitor Reports has no Methods report, and the Diagnostics Monitor Reports has no Packages/Applications report?
3. Does rotational frequency determine the size of an AppMetrics Monitor database?
4. Where in a Diagnostics Monitor would you look for specific metrics on named transactions?

Section 14: Issues in Analysis



The slide features the Xtremesoft logo (a starburst of purple and green dots) and the title "Issues in Analysis" in yellow. Below the title, there are two bullet points:

- Library Packages
- Monitoring IIS applications with MTS/COM+

At the bottom left, it says "Rev 8/13/07" and at the bottom right, "28".

The diagram shows a "Core IIS application" receiving "HTTP requests". It contains two sub-applications: "App1" (with file "inetrb.asp") and "App2" (with file "diffstat.asp"). "App1" is associated with a "System logon session" and "App2" with an "IWM Machine batch logon session". Both "App1" and "App2" are also associated with "IWM Machine batch logon session" boxes, which contain "App2" and "App1" respectively, with file "diffstat.asp".

Library Packages

In the Applications Configuration Panel, AppMetrics only shows server applications. However, AppMetrics can monitor library applications, but only if they run within the context of a monitored server application. In particular, AppMetrics detects and monitors the components of library applications if they were called by a server application.

Why? Library applications run within the process space of the applications that call them, thus they are instantiated only when they are called by a monitored server application. Once a library application is visible, AppMetrics displays this component information within the Components Runtime Panel and in the Components Configuration Panel.

The Components Runtime Panel enables the user to select the metrics to be displayed by component. The Component drop-down list contains the all the components that were active during the prior session within the selected server application. The Component drop-down will also contain components of library applications if they were called within the context of the selected server application.

The Components Configuration panel enables the user to set benchmarks and thresholds for components of interest. Here also, the Component drop-down will contain components of library applications if they were called within the context of the selected server application.

When listing library components, AppMetrics lists the name of the called component followed by the name of the library application (in parentheses). For example, if the component is "FMStore_Events.ShoppingCart", and the library application is "FMStocks 2000 Events", then AppMetrics would list the component as "FMStore_Events.ShoppingCart (FMStocks 2000 Events)".

Monitoring IIS applications with MTS/COM+

Because IIS applications (including ASP) are implemented as COM registered objects in IIS4 and IIS5, AppMetrics for Transactions can provide direct performance monitoring of your IIS processes. Before you can monitor the IIS application you must understand and know what kind of IIS process isolation your package has implemented.

When ISAPI and ASP were first introduced, they were based on in-process execution like a standard library package - a single faulty ISAPI DLL was capable of bringing down all Web sites hosted on the server. IIS 4.0 introduced a new architecture based on a new component called the Web Application Manager (WAM). You can think of the WAM as a simple COM wrapper around existing ISAPI functionality. The WAM is implemented as a standard MTS-registered COM object that encapsulates all the logic required to locate and load an ISAPI DLL and process the ISAPI request. Each IIS application has an associated WAM object responsible for its ISAPI functionality.

In IIS4, the WAM is a standard MTS component, it can take advantage of MTS provided process isolation. Furthermore, because process isolation is a declarative attribute of an MTS package, you can configure the WAM to be part of either a library package ('in process' - it will run in inetinfo.exe) or as part of a server package ('out of process' - it will run isolated in the MTS-provided surrogate process mtx.exe).

A new option in IIS 5.0 occurs between out-of-process and in-process applications: All IIS applications configured as *pooled* out-of-process will run under one server package/application created under MTS/COM+. Below are the different IIS process isolation levels.

IIS In-Process Applications. If an IIS application is configured to run as in-process (low application protection in IIS 5.0), it runs in the Web server's (inetinfo) process space. Running in-process offers a performance advantage, but the drawback is that a single malfunctioning application could bring down the entire Web server. All applications are run as in-process in IIS 4.0 by default

IIS Out-of-Process Applications. Running an application out-of-process prevents a single application error from bringing down the inetinfo process. This kind of server protection is invaluable for managing complex production Web sites and hosting multiple

applications. If an application malfunctions, its process is automatically terminated without affecting the inetinfo process.

Another advantage of running an application out-of-process is that you can unload the application without having to restart IIS. The main disadvantage of out-of-process applications is that they each have their own process space and overhead that affects performance. Also, configuring a number of applications to run out-of-process in a single server box can be quite resource-intensive and affect the performance of the server.

IIS Pooled Out-of-Process Applications (IIS 5.0 Only). A new option in IIS 5.0 occurs between out-of-process and in-process applications: All IIS applications configured as *pooled* out-of-process will run under one server package/application created under MTS/COM+.

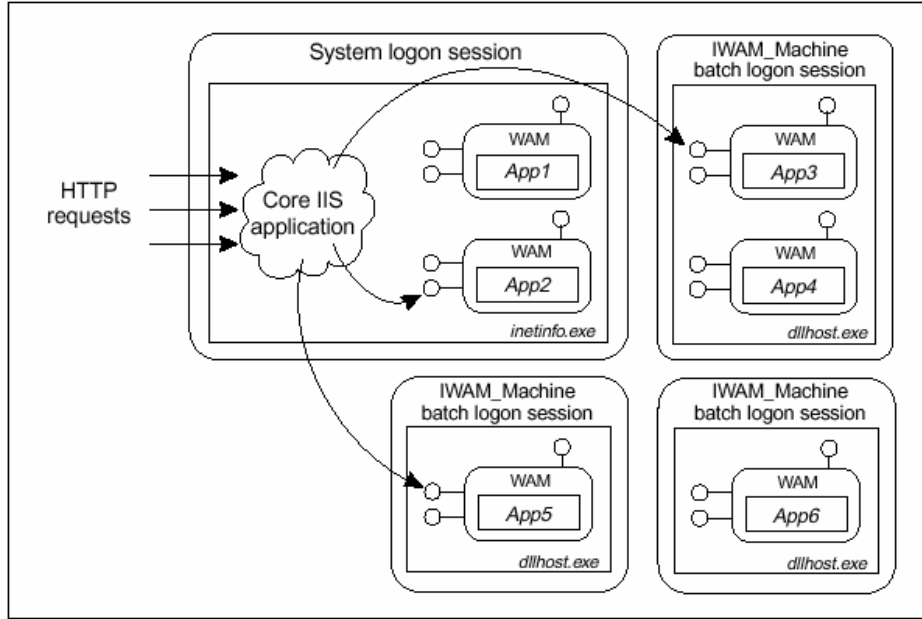
This option shares the out-of-process advantage, because pooled apps have their own process space, and a failure would not affect the availability of the Web server. In addition, since all the pooled applications share the same process space, this option is not as resource-intensive as the out-of-process option. The disadvantage is that since all the pooled applications share the same process space, a failure in one application would bring down all the applications that share the pool.

In IIS 5.0, applications run as pooled out-of-process by default.

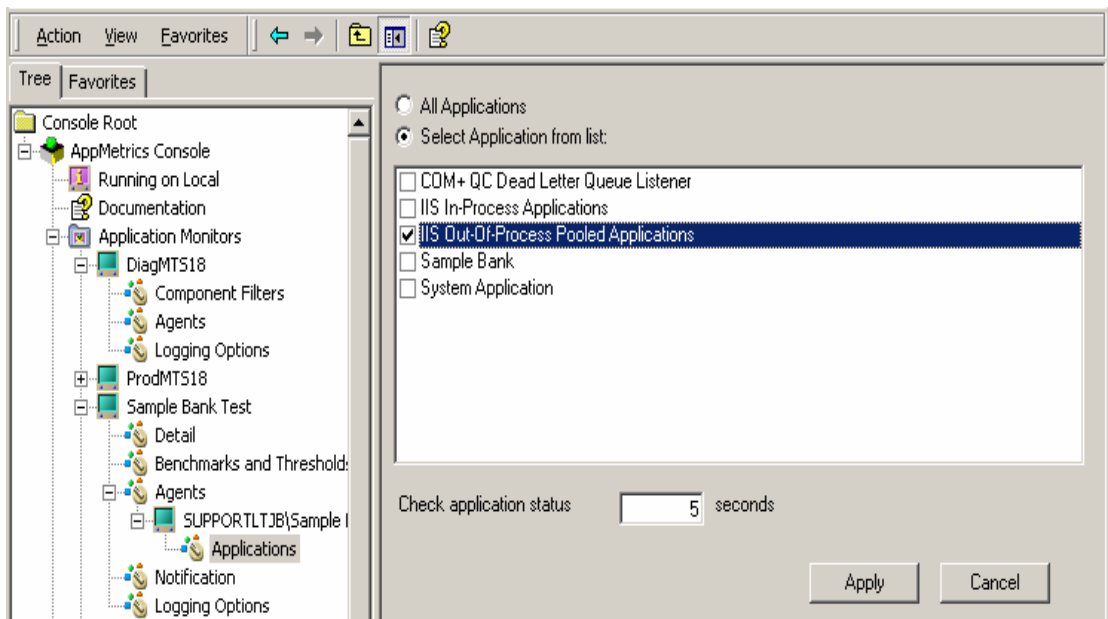
To demonstrate the differences between these levels of process isolation, imagine creating six virtual directories, App1 through App6, and configuring them as follows:

- App1: Low (IIS Process)
- App2: Low (IIS Process)
- App3: Medium (Pooled)
- App4: Medium (Pooled)
- App5: High (Isolated)
- App6: High (Isolated)

The illustration below shows the results if all these applications are in use simultaneously.



Once you understand where the IIS application will be monitored (IIS In-Process Applications, IIS Out-of-Process Applications, or IIS Pooled Out-of-Process Applications) you can select the correct package/application type for monitoring (from the Package/Applications list in the Agent node of your Production or Diagnostics Monitor – see illustration below).



Functionally, IIS In-Process and IIS Out-of-Process Pooled applications look like server applications to AppMetrics.

Section 15: Managing Applications



Xtremesoft

Managing Applications

- Establishing Accurate Thresholds
- Diagnostics Monitors in-depth
- Analyzing Transaction Trends

Rev 8/13/07 Copyrighted material 29

Establishing Accurate Thresholds

One way to determine accurate warning and notification thresholds is through discovering your site's "normal" levels of activity. You can discover this information by performing the following tasks:

1. Create a Production monitor for your packages/applications.
2. Turn off notifications.
3. Run the monitor for a week.
4. Use the AppMetrics Reports to view the activity levels for your packages/applications during the week.

Given the weeklong report, you should be able to see your system's normal levels of activity. You can then apply your organization's service level agreements, if any, to these normal activity levels. They can serve as the basis for your thresholds.

Diagnostics Monitors in-depth

The primary purpose of AppMetrics Diagnostics monitors is to record every event produced by MTS or COM+ with regards to the application under test, and to record those events into sets of log files. They are optional for monitoring your packages/applications. You may use them when you detect problems in your packages/applications through the Production monitors. Your instructor may discuss this topic in more depth.

Analyzing Trends

You can use AppMetrics Reports to discern trends over time. Although the trends for your packages/applications will be unique to your site, one potentially common trend is when the memory usage of your packages/applications rises over time. This indicates a potential memory leak. Other trends to notice are spikes in activity.

Your instructor may provide sample reports to show instances of trends for specific types of packages/applications.



Section 16: Lab 6 – Analyzing your Applications with AppMetrics Reports

Objectives

- Learn to discover packages/applications that may be using excessive resources
- Learn to recognize applications with potential memory leaks
- Learn to uncover failing packages/applications and components
- Learn to view the individual method durations in a transaction

Prerequisites

- A DCOM network connection to the Instructor’s computer for access to the ProdMTS18 database
- A DCOM network connection to the Instructor’s computer for access to the ProdMTS18 database

Preparation

Students should perform this Lab individually on an AppMetrics Console/Reports client.



Students should ensure that the Macro security level setting in Excel on the AppMetrics Manager machine - and any AppMetrics Console/Reports clients you are using – is set to **Medium** or **Low**.

Check this setting by selecting **Macro -> Security** from the **Tools** menu in Excel.

Part 1: Discover applications that may be using excessive resources



Scenario: Your organization has deployed an enterprise management solution, and your team is given daily reports of resource utilization on your production servers (CPU, Memory, Bytes In/Out, etc.). In the last week, 4/18 to 4/19 you have noted exceptionally high memory utilization on server ‘SrvMTS18’. SrvMTS18 is an MTS server running many different applications, some ASP based. Your job is to review the AppMetrics data for this server and determine if any package is exhibiting excessive memory and CPU utilization.

Your objective in this section of the lab is to review the Production Monitor data in database ProdMTS18 and locate the package that may be 'hogging the processor'.

1. AppMetrics for Transactions provides the AppMetrics Reports tool for use in viewing and analyzing your Monitor's data.

AppMetrics Reports are viewed and managed via a macro-enabled Excel workbook installed with any AppMetrics Manager and Console/Reports client installation.

2. Start the AppMetrics Reports by selecting **AppMetrics for Transactions -> AppMetrics Reports** from the **Xtremesoft** program group in your Windows Programs menu.

If prompted select the **Enable Macros** option.

3. The AppMetrics Reports dialog appears when you start AppMetrics Reports. The AppMetrics Reports dialog enables you to create a specific set of reports. It contains the tabs where you choose the Manager (machine name), Dataset (Monitor name), Time Filter (time period of report), and report set (specific report components). Once you enter the information in this dialog, you can then create the reports.

Note: You should complete each dialog tab in consecutive order, starting with the Manager tab before proceeding to the next. The following four subsections describe each tab.

The AppMetrics Reports dialog disappears after you create a report. To reopen the dialog at any time, right-click any non-chart cell, and then choose **AppMetrics** in the popup menu.

4. First input the SQL Server instance name of your Instructor's AppMetrics Manager in the **AppMetrics Manager Machine** field of the **Manager** tab (see Figure 1).

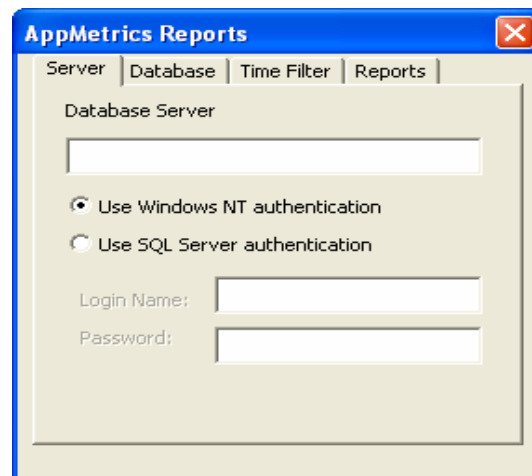


Figure 1

5. Next, click on the **Dataset** tab.

Select the name of your Production Monitor from the drop-down list (see Figure 2).

For this portion of the lab, you should select Dataset 'ProdMTS18'.

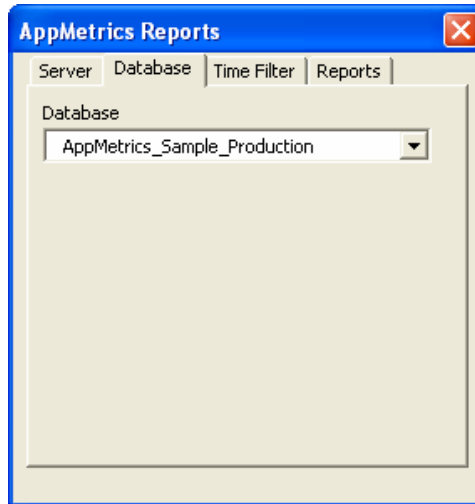


Figure 2

6. Click on the **Time Filter** tab to select the time period of the data to be viewed. This tab provides dates and times that are available (i.e. data was generated and uploaded to the database) for the monitor chosen in the Dataset tab (see Figure 3).

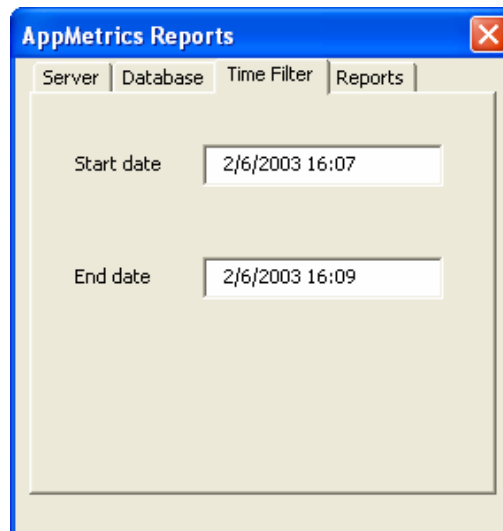


Figure 3



For this lab, our data spans an approximate 2 day period. As in a real world situation, you **do not** know which date and time period the data you seek resides in. So you should choose the maximum time period to report against.

Note: Choosing a large time period to report against may result in long processing waits while SQL processes the query and returns the results.

7. Click on the **Reports** tab, and select the specific report components you wish to see generated (see Figure 4).

Each report results in its own worksheet in the Excel window. By default, all the reports are selected. If desired, you can deselect the ones you do not want to generate. The list of available reports in this tab depends on the monitor type chosen in the Dataset tab.

In this lab we are predominantly concerned with package process data. So we should select the 'Applicatoins' report check-box only.

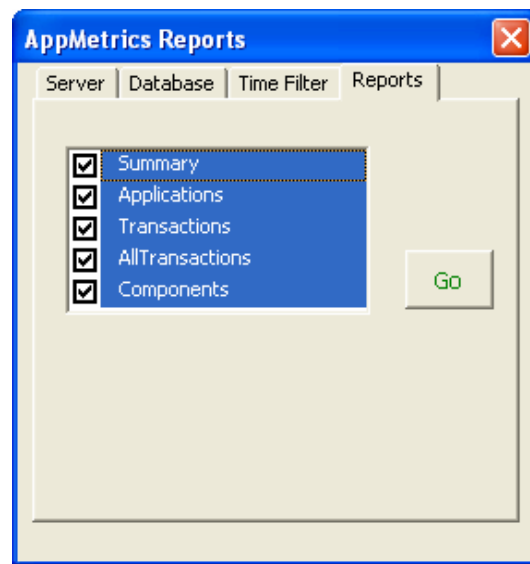


Figure 4

Click the **Go** button.

8. The **Packages/Applications Report** will be generated

The Packages/Applications Report has four charts:



- **Application Percent CPU** – Shows the portion of the CPU used by the application during each interval. It can help determine if the currently charted application is using an excessive amount of CPU.

- **Application Memory** – Shows how much memory was being used by the application during each successive interval. A rising chart may indicate the application has a memory leak.
- **Application Page faults per Second** – Shows the rate of page faults. An excessive rate may result from low amounts of RAM on the server.
- **Application Threads** – Shows how many threads were used by the application during each interval.

Beneath the last chart, the report also includes a table that lists the individual applications measured by the monitor. For each individual application measured during the time period of the report, the table provides the average, minimum and maximum values.

Note: Any value below one millisecond is displayed as a zero value in the table.



9. Look at the Package/Application report.

Where does it identify memory utilization of a package?

Package memory graph and the spread sheet at the bottom of the graphs.

Would it make sense to chart each package individually to see which was using the most memory resources?

You can find that at the spread sheet

- (a) Which package(s) have been ‘hogging the processor’? If more than one, which package was the highest?

System Package

- (b) What was the maximum point of non-system CPU utilization?

37%

- (c) At what date and time did this package reach its maximum point of CPU utilization? How did you find this information?

4/18/02 at 18:15 found by charting the system package

- (d) What was the maximum point of memory utilization?

4112384

- (e) At what date and time did this package reach its maximum point of memory utilization? How did you find this information?

4/18/02 at 18:16 Package memory for system package

Part 2: Recognize Applications with potential memory leaks



Scenario: Systems monitoring shows an MTX.EXE process that is ‘chewing up memory’. Find the package with high memory growth and identify if there is a potential memory leak or high memory utilization that is valid and acceptable under current load and conditions.

1. Start the AppMetrics Reports by selecting **AppMetrics for Transactions -> AppMetrics Reports** from the Xtremesoft program group in your Windows Programs menu.

If prompted select the **Enable Macros** option.

2. First input the machine name of your Instructor’s AppMetrics Manager in the **AppMetrics Manager Machine** field of the **Manager** tab (see Figure 1).

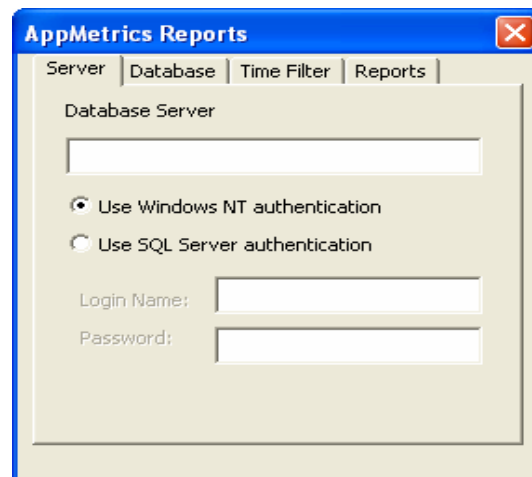


Figure 1

3. Next, click on the **Dataset** tab.

Select the name of your Production Monitor from the drop-down list (see Figure 2).

For this portion of the lab, you should select Dataset ‘ProdMTS18’.

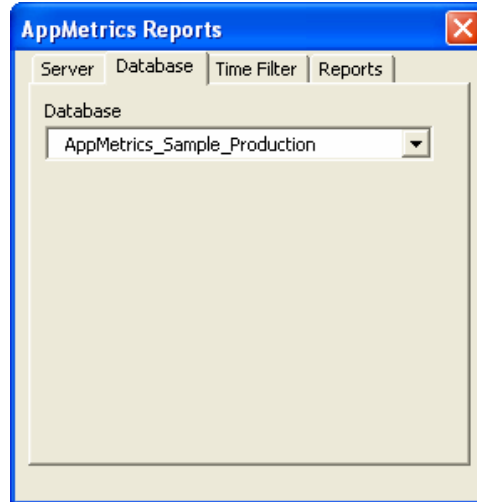


Figure 2

4. Click on the **Time Filter** tab to select the time period of the data to be viewed. This tab provides dates and times that are available (i.e. data was generated and uploaded to the database) for the monitor chosen in the Dataset tab (see Figure 3).

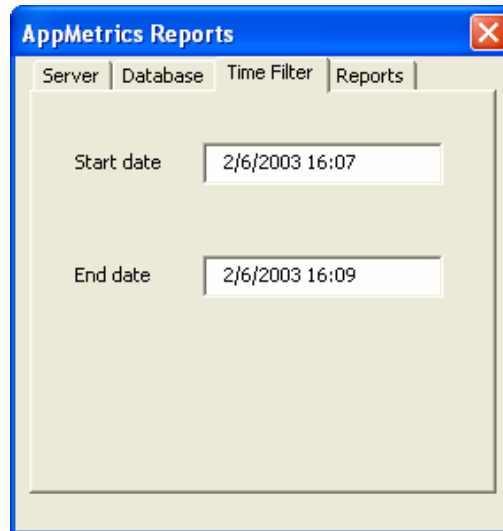


Figure 3



For this lab, our data spans an approximate 2 day period. As in a real world situation, you **do not** know which date and time period the data you seek resides in. So you should choose the maximum time period to report against.

Note: Choosing a large time period to report against may result in long processing waits while SQL processes the query and returns the results.

5. Click on the **Reports** tab, and select the specific report components you wish to see generated (see Figure 4).

Each report results in its own worksheet in the Excel window. By default, all the reports are selected. If desired, you can deselect the ones you do not want to generate. The list of available reports in this tab depends on the monitor type chosen in the Dataset tab.

In this lab we are predominantly concerned with package process and transaction data. So we should select the 'Applications' and 'Transactions' report check-boxes only.

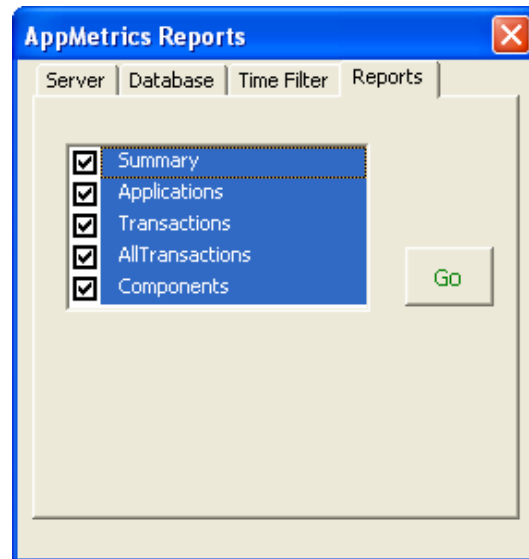


Figure 4

Click the **Go** button.

6. The **Packages/Applications Report** and the **Transactions Report** will be generated
7. Identify which packages/applications were the large memory users.
8. Chart those applications one by one.
9. Identify those that show a steady linear memory growth
10. Find the most likely suspect.
11. Next we should look at the package's transactions to see if the memory utilization can be attributed to other causes.

Due to the large number of applications monitored it may be difficult to view package transaction data with respect to its column values. So first we should freeze the column headings so that we can continue to see them when we scroll through the transaction data.

- a. Select the first cell of the first row of transaction data
- b. Pull down to Freeze Panes from the Window menu in Excel

Transaction names can be quite large for non-friendly named transactions. To make it easier to see which transactions you are actually reviewing, we should expand the column to fit the transaction names.

- c. Select the first cell (transaction name) for all the rows of transaction data belonging to the suspect package
- d. Pull down to Column -> Autofit Selection from the Format menu in Excel

12. One possible explanation of high memory utilization is high transaction aborts (some transaction exceptions do not clean up allocated memory after themselves). Do you see many exceptions on the transactions of this suspect package? **NO**

13. Chart the transactions for this package one by one. Do you see a linearly growing transaction rate that would partly account for linear growth of memory? **NO**

14. Conclusions.



- a. Which package(s) exhibit the behavior of a memory leak?

FMSTOCKS

- b. Calculate memory growth over time. Identify the date and time at which the memory growth started, and the memory utilization at that point.

Identify the date and time at which memory growth stopped, and the memory utilization at that point.

Approximately what percentage growth did you see over what period of time. 15% over 12 Hours

15. Possible Next Steps. You have completed this part of the lab exercise, and have identified the likely package exhibiting a memory leak. Is that as far as you can go with AppMetrics for Transactions? The answer is no. Locating memory leaks is difficult, and finding the leaking package is phase I of a two phase process. The next phase would involve locating the actual leaking component.

The process of locating a leaking component is outside of the scope of this Lab, but is listed here for your review.

- a. Create an MTS/COM+ Server application *for each* of the components currently residing in the package exhibiting the leak.
- b. Move each component to one of these separate MTS/COM+ applications. In other words, each component lives in its own MTS/COM+ Server application.

- c. Create a Production Monitor in AppMetrics for Transactions and configure it to monitor all of these new MTS/COM+ applications.
- d. Start the Production Monitor.
- e. Run load against the business application for a fixed period of sufficient length to allow a linear leak to exhibit itself.
- f. Rotate the log files in the AppMetrics Production Monitor.
- g. Generate a packages/applications worksheet for this Production Monitor in the AppMetrics Reports.
- h. Observe the memory columns in the tabular data set at the bottom of the report.
- i. Chart the individual packages/applications in the AppMetrics packages/applications reports worksheet.
- j. The chart for a package/application (i.e. component) that is leaking should display a linear rise.

Part 3: Recognize failing packages/applications and components



Scenario: Inventory Managers at a shipping hub are reporting that their bill-to/ship-to transactions are failing. They have noted this behavior on the afternoon of 4/18 or 4/19. The Network Operations Center is blaming the operating system. The System Admins are blaming the network. The Developers are blaming the middleware layer. Management wants you to end the finger-pointing and discover which applications and components are failing. Use AppMetrics to determine if the failures are occurring within MTS/COM+, and if so, which application is failing.

1. Start the AppMetrics Reports by selecting **AppMetrics for Transactions -> AppMetrics Reports** from the Xtremesoft program group in your Windows Programs menu.
If prompted select the **Enable Macros** option.
2. First input the machine name of your Instructor's AppMetrics Manager in the **AppMetrics Manager Machine** field of the **Manager** tab (see Figure 1).

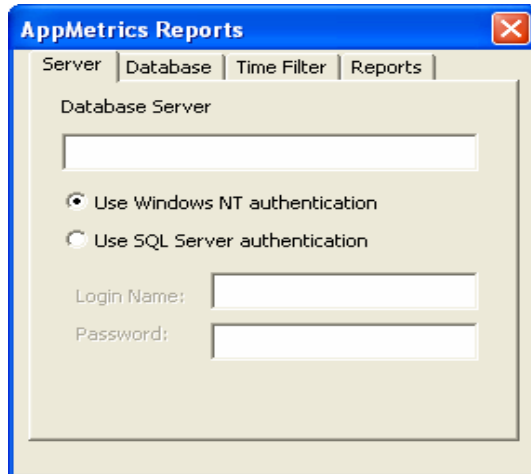


Figure 1

3. Next, click on the **Dataset** tab.

Select the name of your Production Monitor from the drop-down list (see Figure 2).

For this portion of the lab, you should select Dataset 'ProdMTS18'.

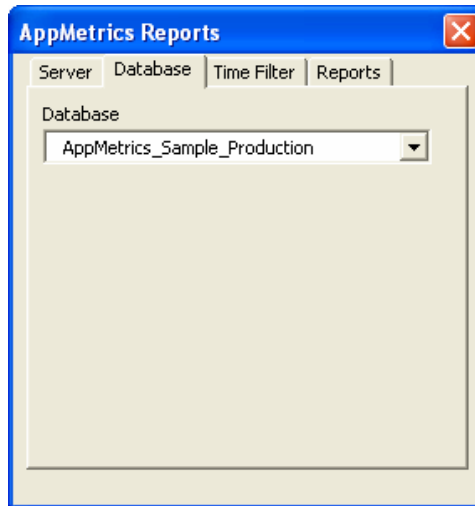


Figure 2

4. Click on the **Time Filter** tab to select the time period of the data to be viewed. This tab provides dates and times that are available (i.e. data was generated and uploaded to the database) for the monitor chosen in the Dataset tab (see Figure 3).

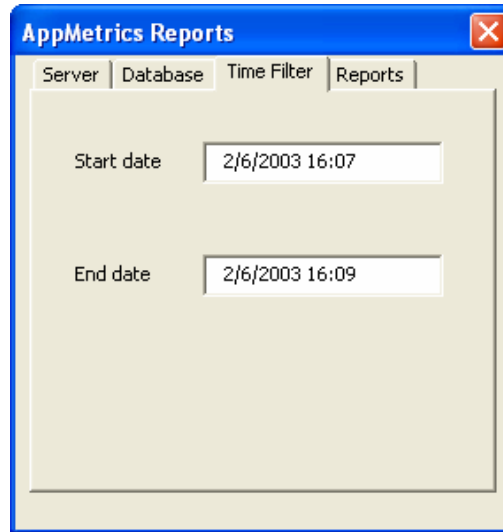


Figure 3



For this part of the lab, our data spans approximately 2 days. You *do* know which date and time period the data you seek resides in (based on the inventory manager's complaint). So you should choose the appropriate time period to report against.

Note: Choosing a large time period to report against may result in long processing waits while SQL processes the query and returns the results.

5. Click on the **Reports** tab, and select the specific report components you wish to see generated (see Figure 4).

Each report results in its own worksheet in the Excel window. By default, all the reports are selected. If desired, you can deselect the ones you do not want to generate. The list of available reports in this tab depends on the monitor type chosen in the Dataset tab.

In this lab we are predominantly concerned with package process and transaction data. So we should select the 'Applications', 'Transactions' and 'Components' report check-boxes only.

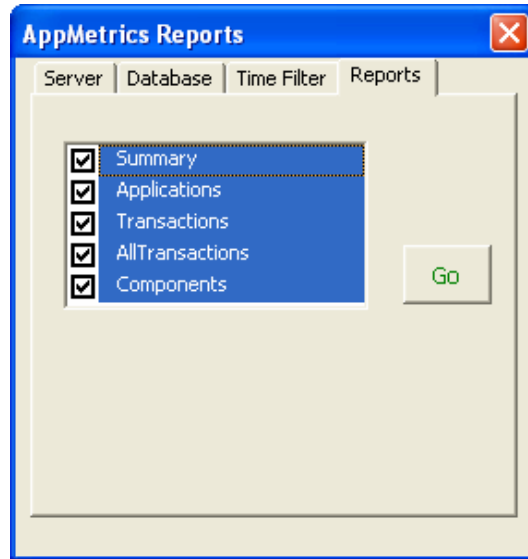


Figure 4

Click the **Go** button.

6. The **Transactions Report** and the **Components Report** will be generated
7. View the transactions report table below the charts. Can you identify a package with frequently failing transactions?
8. To identify a package with frequently failing transactions, we should look at the transactions report table to see which transactions are exhibiting the highest rates of abort.

Due to the large number of applications monitored it may be difficult to view transaction data with respect to its column values. So first we should freeze the column headings so that we can continue to see them when we scroll through the transaction data.

- a. Select the first cell of the first row of transaction data
- b. Pull down to Freeze Panes from the Window menu in Excel
- c. When ready to return to the charts, you should pull down to UnFreeze Panes from the Window menu in Excel

Transaction names can be quite large for non-friendly named transactions. To make it easier to see which transactions you are actually reviewing, we should expand the column to fit the transaction names.

- d. Select the first cell (transaction name) for a package whose rows of transaction data are showing frequent aborts
- e. Pull down to Column -> Autofit Selection from the Format menu in Excel

9. Chart the suspect transactions one by one. Do you see any periods of high transaction abort rates that match the timeframe reported by your Inventory Managers?

10. Next we should look at the components report to see which package components are exhibiting the highest rates of abort.

Due to the large number of applications monitored it may be difficult to view component data with respect to its column values. So first we should freeze the column headings so that we can continue to see them when we scroll through the component data.

- a. Select the first cell of the first row of component data
- b. Pull down to Freeze Panes from the Window menu in Excel

Component names can be quite large. To make it easier to see which components you are actually reviewing, we should expand the column to fit the component names.

- c. Select the first cell (component name) for a package whose rows of component data are showing frequent aborts
- d. Pull down to Column -> Autofit Selection from the Format menu in Excel

11. Chart the suspect components for this package one by one. Do you see any periods of high transaction abort rates that match the timeframe reported by your Inventory Managers? Are the rates for component aborts evenly spread out through the charted period or clustered into certain peaks? Clustered aborts that track closely to started component rates may demonstrate a component that is failing frequently under load.

Conclusions.



a. Which package(s) exhibit the most frequently failing components?

Sample Bank

b. Which package transaction has the highest abort rate?

|Sample Bank|Bank.MoveMoney.VC|Perform

c. Which package component has the highest abort rate?

Bank.MoveMoney.VC

d. What other conclusions can we draw about this package's transaction and component failures? **Component failures track closely to component started rates. In this case the component failure rate tracks too closely to the started rate to draw any conclusion that this application is failing under**

higher load. Nor can we draw any specific conclusions about these failures based on the time of abortions.

- e. Why don't we use the Diagnostic Reports for this analysis?
Diagnostic reports will not chart failures against started and completed rates.

Part 4: Locating the longest running method in a long duration transaction



Scenario: AppMetrics notifications have alerted the Performance Management team that a business critical transaction, Bank.Account has exceed its threshold for transactional duration (3000 ms.). Locate this transaction and determine which component and method are taking the longest to complete.

1. Start the AppMetrics Reports by selecting **AppMetrics for Transactions -> AppMetrics Reports** from the **Xtremesoft** program group in your Windows Programs menu.

If prompted select the **Enable Macros** option.

2. First input the machine name of your Instructor's AppMetrics Manager in the **AppMetrics Manager Machine** field of the **Manager** tab (see Figure 1).

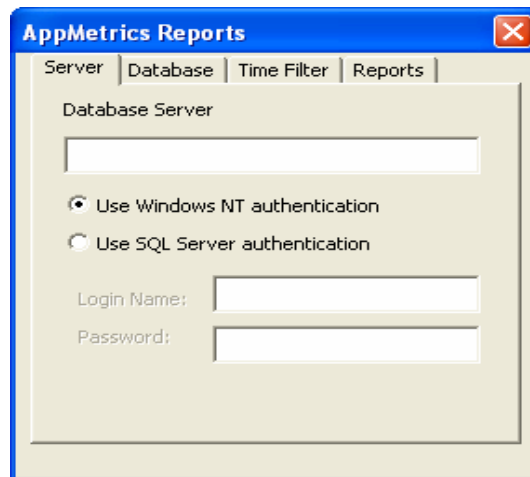


Figure 1

3. Next, click on the **Dataset** tab.

Select the name of your Diagnostic Monitor from the drop-down list (see Figure 2).

For this portion of the lab, you should select Dataset 'DiagMTS18'.

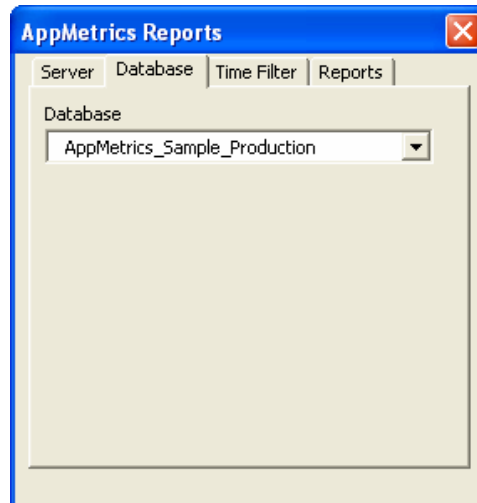


Figure 2

4. Click on the **Time Filter** tab to select the time period of the data to be viewed. This tab provides dates and times that are available (i.e. data was generated and uploaded to the database) for the monitor chosen in the Dataset tab (see Figure 3).

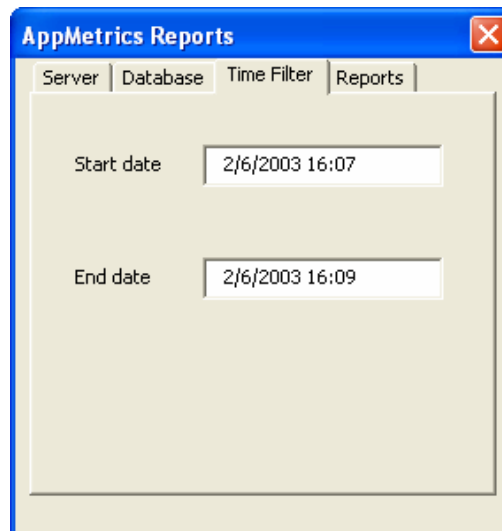


Figure 3

For this part of the lab, our data spans approximately 8 hours. You **do not** know which date and time period the data you seek resides in, so you should choose the entire time period to report against.

Note: Choosing a large time period to report against may result in long processing waits while SQL processes the query and returns the results.

5. Click on the **Reports** tab, and select the specific report components you wish to see generated (see Figure 4).



Each report results in its own worksheet in the Excel window. By default, all the reports are selected. If desired, you can deselect the ones you do not want to generate. The list of available reports in this tab depends on the monitor type chosen in the Dataset tab.

In this lab we are predominantly concerned with transactional data. So we should select the ‘Transactions’, ‘Components’ and ‘Methods’ report check-boxes only.

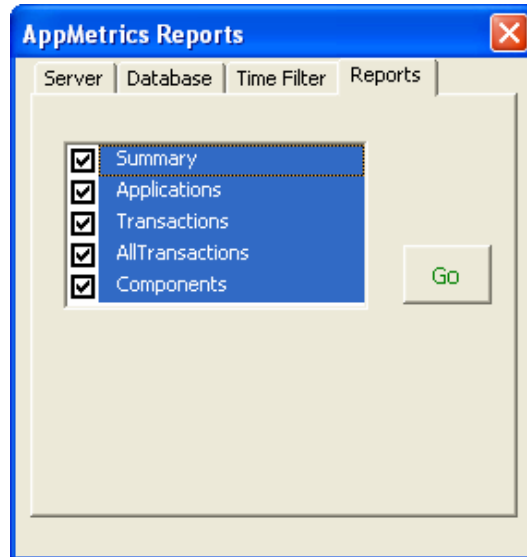


Figure 4

Click the **Go** button.

6. The **Transactions Report**, **Components Report**, and **Methods Report** will be generated
7. View the transactions report table below the charts. Can you locate the long transaction you were notified of?
8. Position your cursor in a blank cell to the right of the charts. Now cursor over to the right until you see the actual transactional data in tabular format. This is the raw data that the report is charting.
9. Locate the data row that corresponds to the long transaction you were notified of. Note the time stamp of the interval for use later.
10. Next we should look at the components worksheet to see which transaction component exhibited the same long duration during that interval. Chart that component.
11. Position your cursor in a blank cell to the right of the charts. Now cursor over to the right until you see the actual component data in tabular format. This is the raw data that the report is charting.

12. Locate the data row that corresponds to the long transaction component you were notified of. Note the time stamp of the interval. It should match the interval noted above.
13. Next we should look at the method worksheet to see which component method(s) had a maximum duration that exceeded the 3000 ms.
14. If there is more than one method, then you will need to chart the methods and view the actual raw data (to the right of the charts) to see which method had the maximum duration in that same interval.

Conclusions.



Which Component likely triggered this notification?

Bank.Account

Which Method was the longest method in this transaction?

Bank.Account->Post

Which interval did the triggering component and method occur in?

11:35:31 of 4/18



Section 17: Lab 7 – Querying AppMetrics Databases

Objectives

- **Learn to use SQL Query Analyzer to access valuable AppMetrics data not in the reports**
- **Querying a Production Monitor**
- **Querying a Diagnostic Monitor**
- **Example: Locate crashing components and methods**

Prerequisites

- A DCOM network connection to the Instructor's computer for access to the ProdMTS18 database
- A DCOM network connection to the Instructor's computer for access to the ProdMTS18 database
- SQL Server Query Analyzer installed on student workstation

Preparation

Students should perform this Lab individually via a SQL Server connection to the instructor's workstation to query databases.

Part 1: Using SQL Query Analyzer



AppMetrics Monitors store their data in Microsoft SQL Server. Some of this data is reported directly in the AppMetrics Reports (e.g. Package Name). Some of the data is used to calculate results that are reported directly in the AppMetrics Report (e.g. Transaction Duration). There is still quite a bit of valuable data that is stored in the AppMetrics Monitor database tables, but is not directly available via the AppMetrics Reports. To take full advantage of AppMetrics, it may be necessary to query the database directly for certain data.

There are a number of third party tools that can query a SQL Server database, however Xtremesoft recommends the use of Microsoft's Query Analyzer, which is installed by default in every Microsoft SQL Server installation (Note that MSDE database installations do *not* include SQL Enterprise Manager or Query Analyzer).

All of the queries in this lab are simple, and *do not* involve the use of joins or multiple table references. It is both encouraged and recommended that you experiment with SQL to get your queries optimized. *All* of the queries performed below could have been accomplished with a single Transact-SQL statement.

In this lab we will use Query Analyzer to report directly against the AppMetrics database tables. Figure 1 below shows the important elements of Query Analyzer.

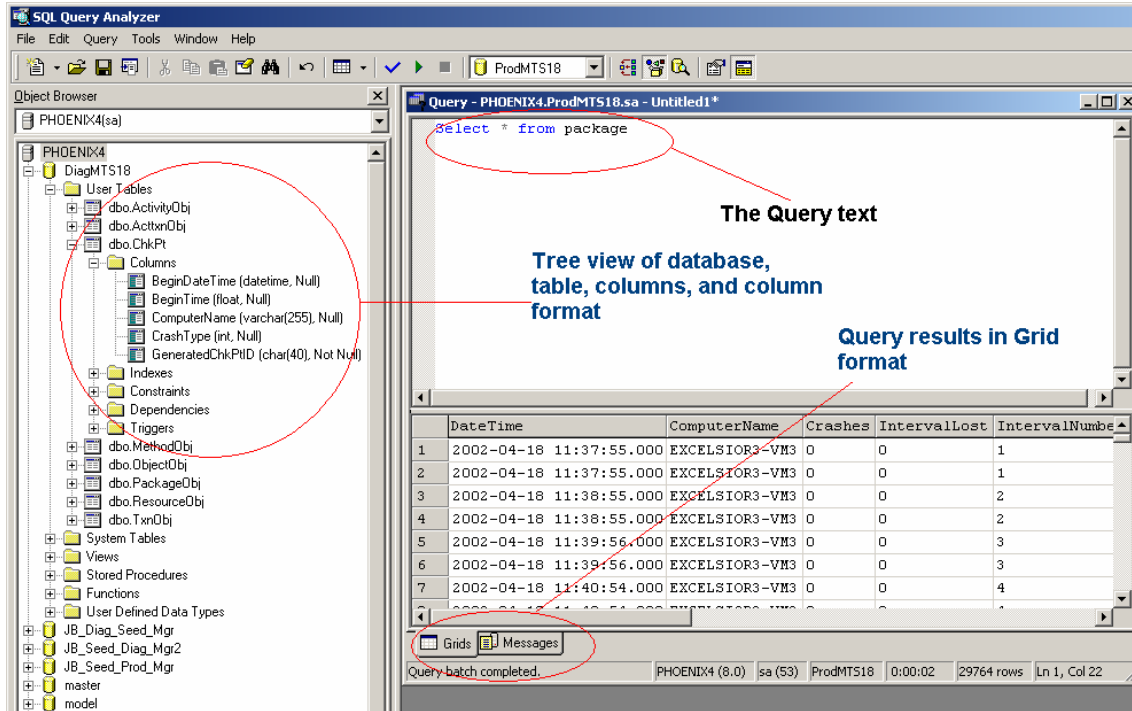


Figure 1

The Query Analyzer interface allows us to code a query, check its syntax, view the record layouts of the tables in our database, and view the results of our query.

Start Query Analyzer from Start -> Program Files -> Microsoft SQL Server -> Query Analyzer.

Part 2: Querying a Production Monitor

The Database Schema of an AppMetrics Production Monitor is available in .PDF format in Appendix B of the AppMetrics documentation.



As the Schema indicates, the tables of a Production Monitor can be joined for queries by either IntervalNumber (integer), PackageName (Varchar(255)), SinkName (Varchar(255)), or DateTime (SQL datetime format).

Most of the valuable data in a Production Monitor is already represented in the AppMetrics Reports, and there is less to discover in a query of a Production Monitor. It is important to remember that the data in a Production Monitor is all interval-based. Further, more fine-grained data should be derived from a Diagnostic Monitor. However the data can be queried to validate the calculated results presented by the Reports, and there are times when you might want to query the Production Report to determine how and when to run a Diagnostic monitor.



Scenario: Query a Production Monitor to determine the last date and time the package Sample Bank was shutdown.

1. Looking in the Schema, we see that the Package log table stores an incrementing count of shutdowns on the package since the monitor was started in the field 'shutdowns'. This count is relative to a specific monitor instance. Along with this shutdown count, the 'datetime' field stores the date and time when the interval ended, and the 'PackageName' field records the name of the package for which the record applies.
2. We can thus query the Package table to show date & time of the interval in which we have a non-zero shutdown count. The date and time of that interval will tell us approximately when the last shutdown occurred (i.e. the last time the shutdown count was incremented).
3. Start Query Analyzer.
4. Select 'ProdMTS18' from the Database pull-down list in the toolbar header.



5. In the Query Analyzer text window input the following query:

```
Select datetime, shutdowns from Package
where PackageName like '%bank%'
and shutdowns <> 0
```

An alternative query would be:

```
Select * from Package
where PackageName like '%bank%'
and crashes <> 0
```

6. You will see the following result set (presented in tabular format below):

2002-04-18 16:42:00.000	1
2002-04-18 16:43:00.000	2
2002-04-18 16:43:59.000	2
2002-04-18 16:45:00.000	2
2002-04-18 16:45:59.000	2
2002-04-18 16:47:00.000	2
2002-04-18 16:47:58.000	2
2002-04-18 16:48:58.000	2
2002-04-18 16:49:59.000	2
2002-04-18 16:51:00.000	2
2002-04-18 16:51:58.000	2
2002-04-18 16:52:59.000	2
2002-04-18 16:53:58.000	2
2002-04-18 16:54:59.000	2
2002-04-18 16:56:00.000	2
2002-04-18 16:56:58.000	3
2002-04-18 16:58:00.000	3
2002-04-18 16:58:58.000	3
2002-04-18 16:59:59.000	3
2002-04-18 17:01:00.000	3
2002-04-18 17:01:58.000	3
2002-04-18 17:02:59.000	3
2002-04-18 17:04:00.000	3
2002-04-18 17:04:58.000	3
2002-04-18 17:05:59.000	3
2002-04-18 17:07:01.000	3
2002-04-18 17:07:59.000	3
2002-04-18 17:54:05.000	4
2002-04-18 17:55:06.000	4
2002-04-18 17:56:04.000	4
2002-04-18 19:00:28.000	4
2002-04-18 19:01:29.000	4
2002-04-18 19:02:27.000	4
2002-04-18 19:03:28.000	4
2002-04-18 19:04:30.000	4

2002-04-18 19:05:28.000	4
2002-04-18 19:06:27.000	4
2002-04-18 19:07:28.000	4
2002-04-18 19:08:30.000	4
2002-04-18 19:09:27.000	4
2002-04-18 19:10:27.000	4
2002-04-18 19:11:30.000	4
2002-04-18 19:12:30.000	4
2002-04-18 19:13:30.000	4
2002-04-18 19:14:28.000	4
2002-04-18 19:15:29.000	4
2002-04-18 19:16:28.000	4
2002-04-18 19:17:29.000	4
2002-04-18 19:18:30.000	4
2002-04-18 19:19:28.000	4
2002-04-18 19:20:30.000	4
2002-04-18 19:21:28.000	4

7. Since you know that the field shutdowns is a ‘counter’, you can infer that the last time a shutdown occurred was the last time the counter was incremented. Shutdowns was last incremented in interval 2002-04-18 17:54:05.000 so the last time that package Sample Bank was shutdown was in the interval ending on 5:54 pm.
8. In a similar manner, simply changing the field ‘shutdowns’ in your query to ‘crashes’ you can find out in which interval your package last crashed.

Part 3: Querying a Diagnostics Monitor



The Database Schema of an AppMetrics Diagnostics Monitor is available in .PDF format in Appendix A of the AppMetrics documentation.

As the Schema indicates, the tables of a Diagnostics Monitor can be joined for queries by many values, depending on what data you wish to extract. Of particular note is the UniqueGUID field. The UniqueGUID is a unique identifier to link instances of activities (ActtxnObj table) , transactions (TxnObj table and ActtxnObj table), objects (ObjectObj table) and methods (MethodObj table). Using the UniqueGUID field it is possible to traverse all of the tables in the Diagnostics Monitor within the context of a single transaction.

Table	Fields
ChkPt	BeginDateTime BeginTime ComputerName CrashType GeneratedChkPtID
TxnObj	ActivityGuid BeginDateTime BeginTime ChkPtRecordID ComputerName EndDateTime EndTime GeneratedChkPtID ObjectID PackageGuid PackageName TsidProp TxnGuidProp TxnStateProp UniqueGuid
PackageObj	BeginDateTime BeginTime ChkPtRecordID ComputerName EndDateTime EndTime GeneratedChkPtID PackageGuid PackageName ProcessID ShutdownReason
ActivityObj	Aborted ActivityGuid BeginDateTime BeginTime ComputerName EndDateTime EndTime FirstID FirstMethod FirstObject NumActivities PackageGuid PackageName ThreadID UserName
ObjectObj	Aborted ActivityGuid BeginDateTime BeginTime ClientIP clsid Completed ComputerName ContextID CreatedTsidObj EndDateTime EndTime ErrID ErrMethod Errstatus ErrstID FirstMethod MethodExcept NumResCreated NumResUsed ObjectID PackageGuid PackageName ServerIP TotalResourceUsageTime TsidProp TxnGuidProp UniqueGuid URL
MethodObj	ApplicationName BeginDateTime BeginTime ComputerName EndDateTime EndTime Errstatus MethodID MethodIID MethodName ObjectID UniqueGuid Valid
ActxnObj	ActivityGuid BeginDateTime BeginTime ChkPtRecordID ComputerName EndDateTime EndTime FirstMethod FirstObject ObjectID PackageGuid PackageName StartTxnPoint TsidProp TxnGuidProp UniqueGuid UsesDTC



Scenario: Query a Diagnostics Monitor to determine the duration of the last method called in a transaction.

1. The best starting place for querying a Diagnostics Monitor in regard to transactions is the ActivityObj table. This table stores the data related to activities, defined as ‘a collection of MTS/COM+ objects that have a single logical thread of execution’. The ActivityObj table identifies the start and end of the transaction, the name of the package, and the activity GUID.

Let’s say that the transaction we are interested in occurred at 3:24:54 on package Sample Bank. We will query all of the activities that occurred at this time on this package.

2. Start Query Analyzer.
3. Select ‘DiagMTS18’ from the Database pull-down list in the toolbar header.
4. In the Query Analyzer text window input the following query:

```
select begintatetime, begintime, enddatetime,
endtime, activityguid, firstobject, packagename
from activityobj
where packagename like '%bank%'
and begintatetime = '2002-04-18 15:24:54.000'
```

5. You will see the following result set (presented in grid format):

begintatetime	begintime	enddatetime	endtime	firstobject	packagename
2002-04-18 15:24:54.000				12663631494792.355	
2002-04-18 15:24:54.000				12663631494472.955	
{184E05F3-5301-11D6-A9B0-005056428685}	Bank.MoveMoney				Sample Bank
2002-04-18 15:24:54.000				12663631494820.768	
2002-04-18 15:24:54.000				12663631494504.982	
{184E05F0-5301-11D6-A9B0-005056428685}	Bank.MoveMoney				Sample Bank

6. This identifies for us that there are two transactions on package Sample Bank at that time. Each transaction started at a different time, however SQL's datetime field is not atomic enough to drill down into thousandths of a millisecond. So AppMetrics uses the begintime field, which is a float variable, and can record very small increments of time. Based on the output of step 5 above, we can see that the transaction with the endtime of 12663631494504.982 is the real first transaction, and this is the one we will use.
7. Next we will use the ActivityGUID field of that first activity ('{184E05F0-5301-11D6-A9B0-005056428685}') to locate the specific transaction that this activity correlates to. In the Query Analyzer text window input the following query:

```
select activityguid, begintatetime, generatedchkptid,
uniqueguid from txnobj
where activityguid = '{184E05F0-5301-11D6-A9B0-005056428685}'
```

8. You will see the following result set (presented in grid format):

activityguid	begintatetime	generatedchkptid	uniqueguid
{184E05F0-5301-11D6-A9B0-005056428685}	2002-04-18 15:24:54.000		
NULL	{1B4BDB13-E0A6-4856-80C1-510306563D20}		

9. One interesting thing to note is that the GeneratedChkPtID is NULL. This means that there was *no* exception (error or abort) during this transaction. However the main reason we queried TxnObj was to look up the UniqueGUID of this transaction.
10. Armed with this UniqueGUID we can now query the MethodObj table, and sort the returned records by the date and time the method was called. The record with the last date and time is the 'last method' that is the goal of this query scenario.

11. In the Query Analyzer text window input the following query:

```
select begindatetetime, begintime, methodname from
methodobj

where uniqueguid = '{1B4BDB13-E0A6-4856-80C1-
510306563D20}'
```

12. You will see the following result set (presented in grid format):

```
begindatetetime begintime methodname
2002-04-18 15:24:54.000 12663631494492.934
Bank.Account->Post
2002-04-18 15:24:54.000 12663631494501.225
Bank.GetReceipt->GetNextReceipt
```



13. Looking at the results, which methodname do you see as the last method in this transaction? **Bank.GetReceipt->GetNextReceipt**

Part 4: Locate Crashing Components and Methods



Now its time to put all of your newfound querying experience to a very practical application. AppMetrics users frequently want to know why a particular package crashed. AppMetrics records the specific exceptions of any method invoked during a transaction. If a package crashes, AppMetrics will have recorded its MTS/COM+ object invocations, and the HRESULTS (exception numbers) of the methods that were running at the time of the crash. Armed with the exception number related to a package crash, Developers can review their method code and discover the cause for the crash.

Scenario: You have been informed by Performance Management that your application NuGroup Server generated an exception at 1pm on 4/18. Let's find out why. We will query the Diagnostics Monitor database and discover the exception related to the crash.

1. First we will query the ObjectObj table to locate any exception (HRESULT) records related to our application NuGroup Server. Hint: An HRESULT of zero means no exception occurred.
2. Start Query Analyzer.
3. Select 'DiagMTS18' from the Database pull-down list in the toolbar header.
4. In the Query Analyzer text window how would you query against table ObjectObj to discover which NuGroup Server objects returned an exception?



```
select aborted, begindatetetime, begintime, errid,
errmethod, errstatus, packagename, uniqueguid
from objectobj where errstatus <> 0
```

and packagename like '%nugroup%'

5. The result set should have 39 records in it. Note which object record id with an exception occurred closest to 1pm on 4/18/02.

```
aborted begintime errid errmethod errstatus packagename  
uniqueguid
```

```
0 2002-04-18 13:00:02.000 12663622802693.322 {00020400-0000-0000-C000-  
000000000046} 5 -2147352570 NuGroup Server {A3E911A6-E950-4DFE-A8E2-  
419816655C14}
```

6. Did this transaction abort? How can you tell?

Transaction did not abort, because Aborted field for this transaction is 0, and zero means did not abort.

7. How would you query against table MethodObj to discover which method was executing at the time of the crash and the exception it returned?

```
select applicationname, begintime, begintime,  
errstatus, methodname from methodobj  
  
where uniqueguid = '{A3E911A6-E950-4DFE-A8E2-  
419816655C14}'
```

8. What is the exception value returned by the crashing transaction?

-2147352570 ('unknown name' error code).

9. Sometimes an exception value is a standard Windows environment exception value (as in this case), and sometimes it is generated by the application itself. When you query and receive an exception value, you can look it up in a tool such as Visual Studio 6.0's 'Error Lookup' tool, or contact the component developer for a definition of the exception.