Xtremesoft

# AppMetrics Solutions

Improving the performance of your Application, by giving you unique visibility into the core of your Application

www.xtremesoft.com

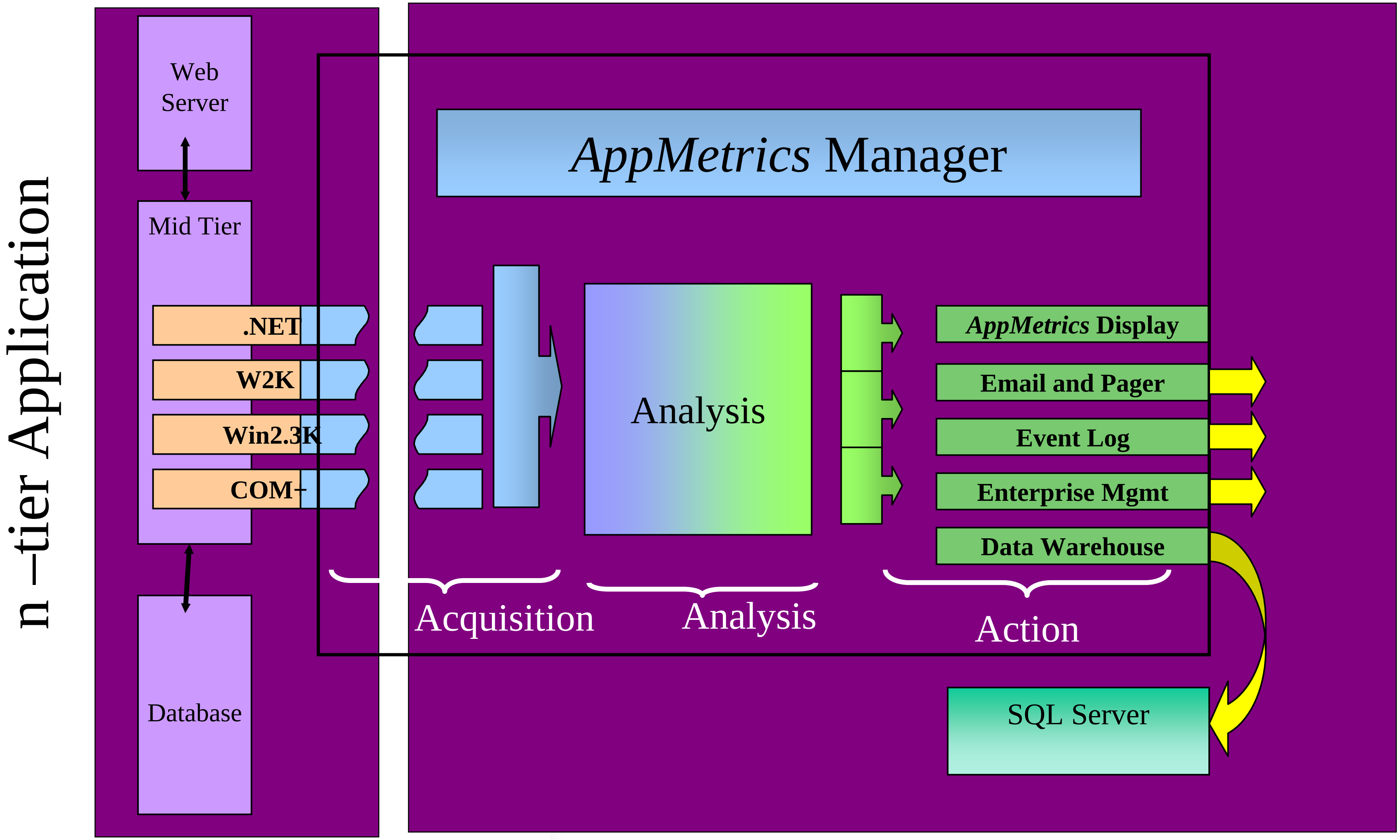
# Typical problems we solve

- Which component is hung?
- Which component and method is slowing down the business process?
- Where should I invest development/engineering resources?
- Where are the leaks?
- Where did performance change?

# Architecture

- The AppMetrics Agent runs on the AppServer application, where it collects and forwards application events to the AppMetrics Manager
- The AppMetrics Manager, usually running on its own machine correlates the events, and generates unique application metrics and persists them to a database for historical reporting
- The Manager monitors the Application metrics in real-time, compares them to benchmarks and alerts based on predefined alert criteria

# Architecture



# AppMetrics Usage Scenarios

- Business Activities Take Too Long?
- Where to Optimize?
- Inconsistent Performance?
- Hung Components?
- Memory Leaks?
- Too Many Finger-pointing Meetings?
- Need for Early Warnings?

# Business Activities Take Too Long?

Problem: **Transactions are running slowly.**

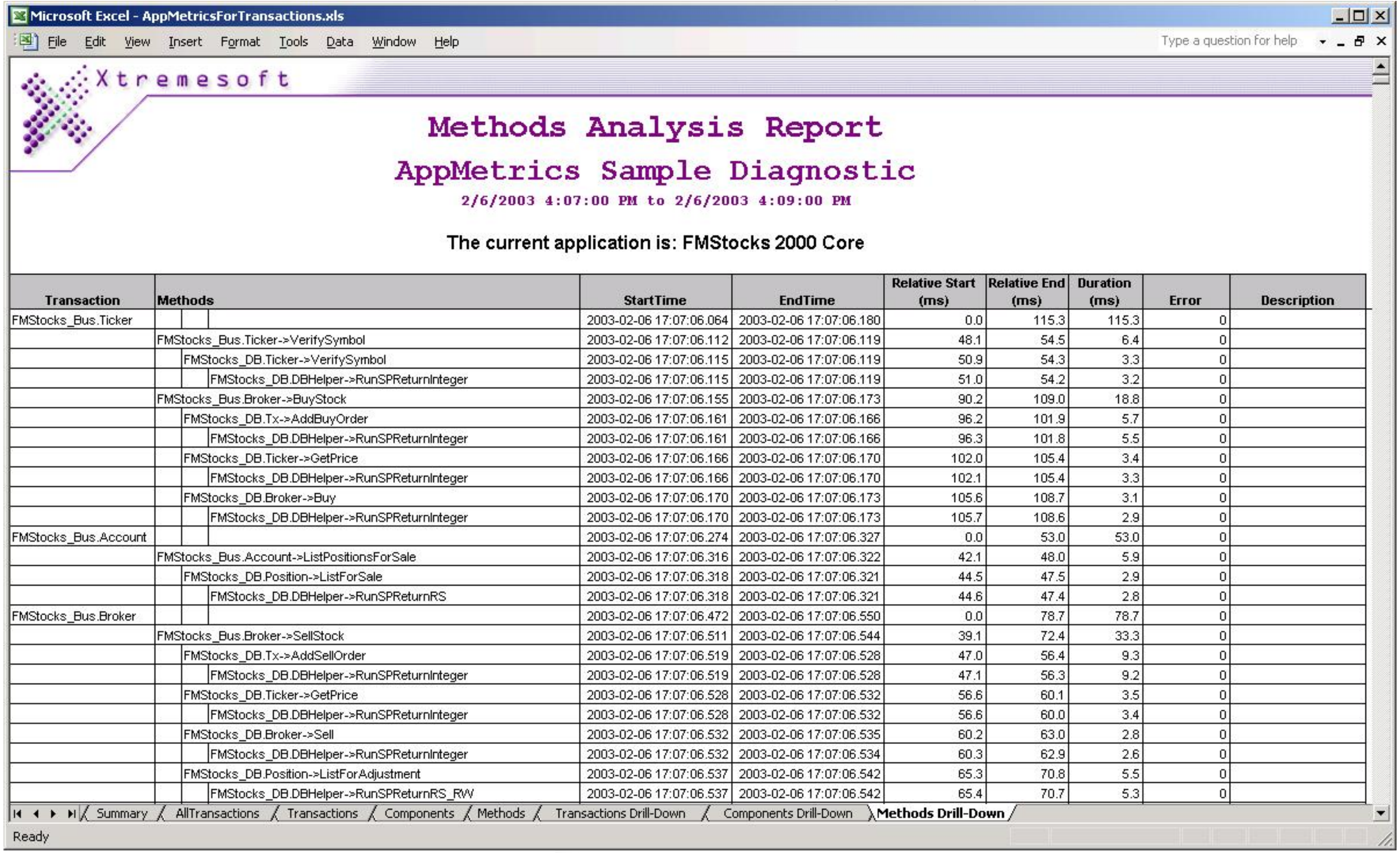
How can I tell which method(s) in each transaction is causing the problem?

- Transactions involve multiple method calls. You may remember the list of method calls involved in a transaction, but which method calls which?
- The relationships between method calls can be easily forgotten. How can you tell which method in the call chain is causing the problem?

# Business Activities Take Too Long?

## Solution: AppMetrics' Method Analysis Report

Each COM+ Activity (transactional or not) is broken down by method call, displaying overall durations as well as the durations of each of the method calls that make up the activity / transaction.



Methods Analysis Report

AppMetrics Sample Diagnostic

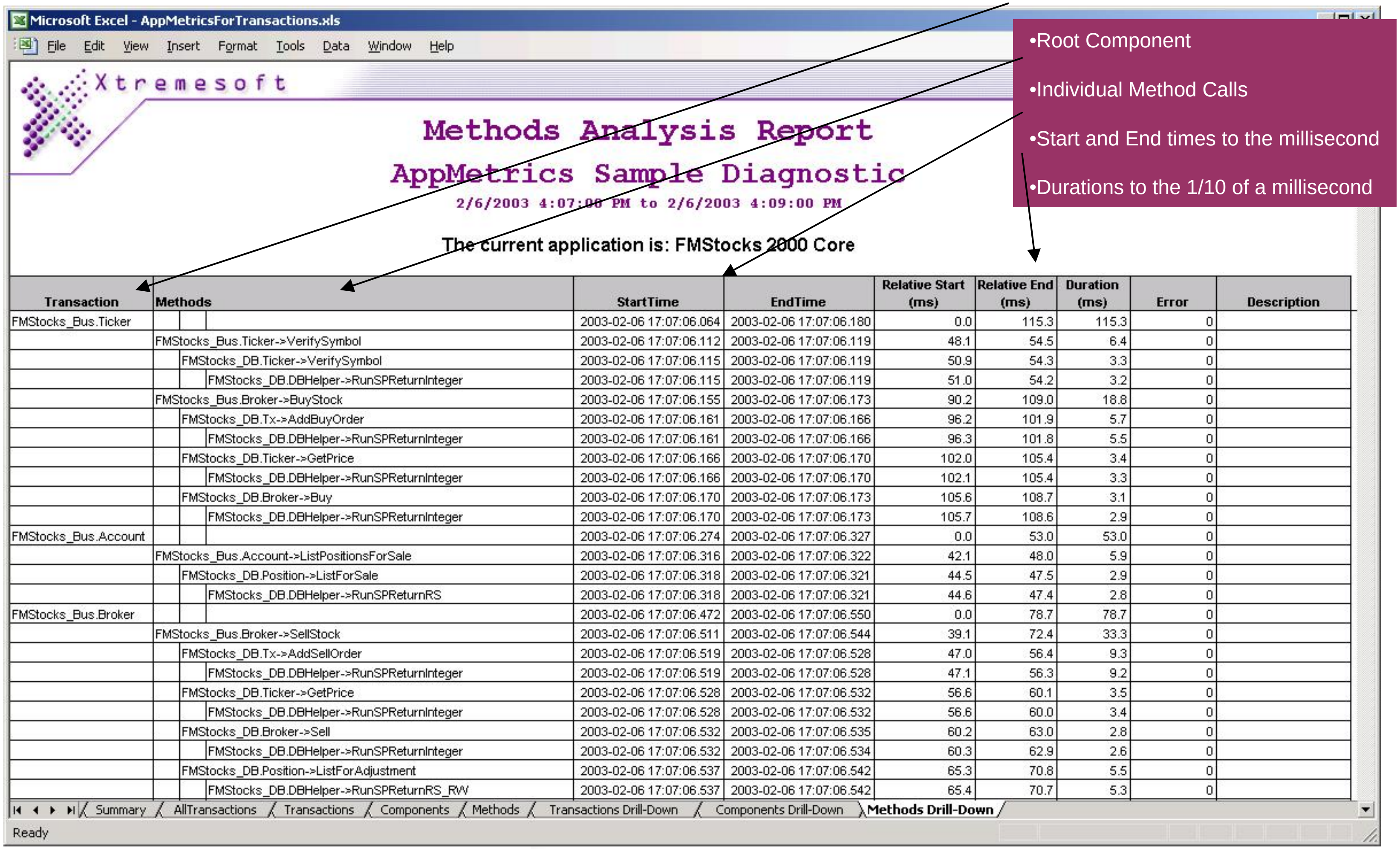
2/6/2003 4:07:00 PM to 2/6/2003 4:09:00 PM

The current application is: FMStocks 2000 Core

| Transaction | Methods | StartTime | EndTime | Relative Start (ms) | Relative End (ms) | Duration (ms) | Error | Description |
|---|---|---|---|---|---|---|---|---|
| FMStocks_Bus.Ticker | | 2003-02-06 17:07:06.064 | 2003-02-06 17:07:06.180 | 0.0 | 115.3 | 115.3 | 0 | |
| | FMStocks_Bus.Ticker->VerifySymbol | 2003-02-06 17:07:06.112 | 2003-02-06 17:07:06.119 | 48.1 | 54.5 | 6.4 | 0 | |
| | FMStocks_DB.Ticker->VerifySymbol | 2003-02-06 17:07:06.115 | 2003-02-06 17:07:06.119 | 50.9 | 54.3 | 3.3 | 0 | |
| | FMStocks_DB.DBHelper->RunSPReturnInteger | 2003-02-06 17:07:06.115 | 2003-02-06 17:07:06.119 | 51.0 | 54.2 | 3.2 | 0 | |
| | FMStocks_Bus.Broker->BuyStock | 2003-02-06 17:07:06.155 | 2003-02-06 17:07:06.173 | 90.2 | 109.0 | 18.8 | 0 | |
| | FMStocks_DB.Tx->AddBuyOrder | 2003-02-06 17:07:06.161 | 2003-02-06 17:07:06.166 | 96.2 | 101.9 | 5.7 | 0 | |
| | FMStocks_DB.DBHelper->RunSPReturnInteger | 2003-02-06 17:07:06.161 | 2003-02-06 17:07:06.166 | 96.3 | 101.8 | 5.5 | 0 | |
| | FMStocks_DB.Ticker->GetPrice | 2003-02-06 17:07:06.166 | 2003-02-06 17:07:06.170 | 102.0 | 105.4 | 3.4 | 0 | |
| | FMStocks_DB.DBHelper->RunSPReturnInteger | 2003-02-06 17:07:06.166 | 2003-02-06 17:07:06.170 | 102.1 | 105.4 | 3.3 | 0 | |
| | FMStocks_DB.Broker->Buy | 2003-02-06 17:07:06.170 | 2003-02-06 17:07:06.173 | 105.6 | 108.7 | 3.1 | 0 | |
| | FMStocks_DB.DBHelper->RunSPReturnInteger | 2003-02-06 17:07:06.170 | 2003-02-06 17:07:06.173 | 105.7 | 108.6 | 2.9 | 0 | |
| FMStocks_Bus.Account | | 2003-02-06 17:07:06.274 | 2003-02-06 17:07:06.327 | 0.0 | 53.0 | 53.0 | 0 | |
| | FMStocks_Bus.Account->ListPositionsForSale | 2003-02-06 17:07:06.316 | 2003-02-06 17:07:06.322 | 42.1 | 48.0 | 5.9 | 0 | |
| | FMStocks_DB.Position->ListForSale | 2003-02-06 17:07:06.318 | 2003-02-06 17:07:06.321 | 44.5 | 47.5 | 2.9 | 0 | |
| | FMStocks_DB.DBHelper->RunSPReturnRS | 2003-02-06 17:07:06.318 | 2003-02-06 17:07:06.321 | 44.6 | 47.4 | 2.8 | 0 | |
| FMStocks_Bus.Broker | | 2003-02-06 17:07:06.472 | 2003-02-06 17:07:06.550 | 0.0 | 78.7 | 78.7 | 0 | |
| | FMStocks_Bus.Broker->SellStock | 2003-02-06 17:07:06.511 | 2003-02-06 17:07:06.544 | 39.1 | 72.4 | 33.3 | 0 | |
| | FMStocks_DB.Tx->AddSellOrder | 2003-02-06 17:07:06.519 | 2003-02-06 17:07:06.528 | 47.0 | 56.4 | 9.3 | 0 | |
| | FMStocks_DB.DBHelper->RunSPReturnInteger | 2003-02-06 17:07:06.519 | 2003-02-06 17:07:06.528 | 47.1 | 56.3 | 9.2 | 0 | |
| | FMStocks_DB.Ticker->GetPrice | 2003-02-06 17:07:06.528 | 2003-02-06 17:07:06.532 | 56.6 | 60.1 | 3.5 | 0 | |
| | FMStocks_DB.DBHelper->RunSPReturnInteger | 2003-02-06 17:07:06.528 | 2003-02-06 17:07:06.532 | 56.6 | 60.0 | 3.4 | 0 | |
| | FMStocks_DB.Broker->Sell | 2003-02-06 17:07:06.532 | 2003-02-06 17:07:06.535 | 60.2 | 63.0 | 2.8 | 0 | |
| | FMStocks_DB.DBHelper->RunSPReturnInteger | 2003-02-06 17:07:06.532 | 2003-02-06 17:07:06.534 | 60.3 | 62.9 | 2.6 | 0 | |
| | FMStocks_DB.Position->ListForAdjustment | 2003-02-06 17:07:06.537 | 2003-02-06 17:07:06.542 | 65.3 | 70.8 | 5.5 | 0 | |
| | FMStocks_DB.DBHelper->RunSPReturnRS_RW | 2003-02-06 17:07:06.537 | 2003-02-06 17:07:06.542 | 65.4 | 70.7 | 5.3 | 0 | |

# Business Activities Take Too Long?

## Solution: AppMetrics' Method Analysis Report



- •Root Component
- •Individual Method Calls
- •Start and End times to the millisecond
- •Durations to the 1/10 of a millisecond

Methods Analysis Report

AppMetrics Sample Diagnostic

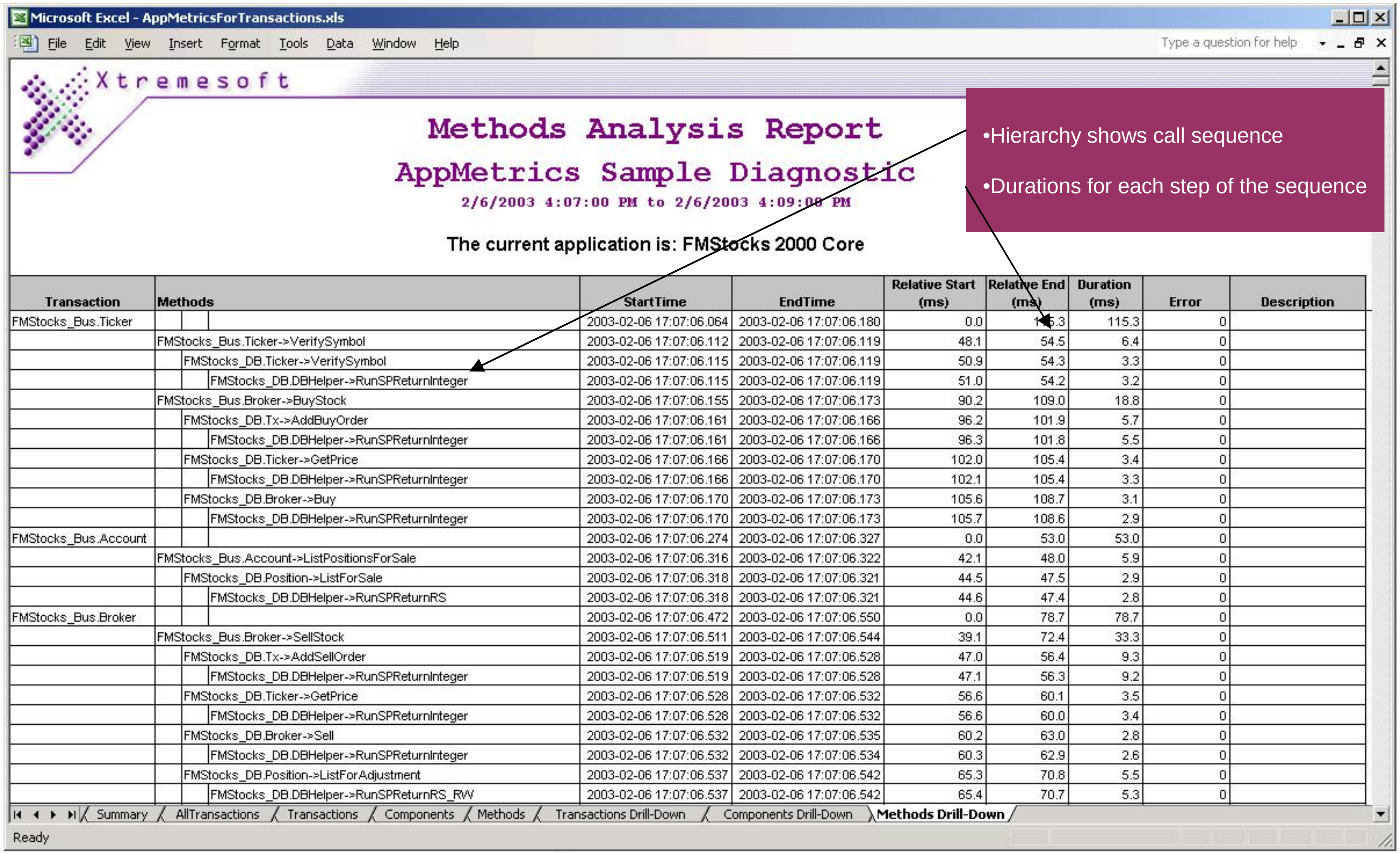
2/6/2003 4:07:00 PM to 2/6/2003 4:09:00 PM

The current application is: FMStocks 2000 Core

| Transaction | Methods | StartTime | EndTime | Relative Start (ms) | Relative End (ms) | Duration (ms) | Error | Description |
|---|---|---|---|---|---|---|---|---|
| FMStocks_Bus.Ticker | | 2003-02-06 17:07:06.064 | 2003-02-06 17:07:06.180 | 0.0 | 115.3 | 115.3 | 0 | |
| | FMStocks_Bus.Ticker->VerifySymbol | 2003-02-06 17:07:06.112 | 2003-02-06 17:07:06.119 | 48.1 | 54.5 | 6.4 | 0 | |
| | FMStocks_DB.Ticker->VerifySymbol | 2003-02-06 17:07:06.115 | 2003-02-06 17:07:06.119 | 50.9 | 54.3 | 3.3 | 0 | |
| | FMStocks_DB.DBHelper->RunSPReturnInteger | 2003-02-06 17:07:06.115 | 2003-02-06 17:07:06.119 | 51.0 | 54.2 | 3.2 | 0 | |
| | FMStocks_Bus.Broker->BuyStock | 2003-02-06 17:07:06.155 | 2003-02-06 17:07:06.173 | 90.2 | 109.0 | 18.8 | 0 | |
| | FMStocks_DB.Tx->AddBuyOrder | 2003-02-06 17:07:06.161 | 2003-02-06 17:07:06.166 | 96.2 | 101.9 | 5.7 | 0 | |
| | FMStocks_DB.DBHelper->RunSPReturnInteger | 2003-02-06 17:07:06.161 | 2003-02-06 17:07:06.166 | 96.3 | 101.8 | 5.5 | 0 | |
| | FMStocks_DB.Ticker->GetPrice | 2003-02-06 17:07:06.166 | 2003-02-06 17:07:06.170 | 102.0 | 105.4 | 3.4 | 0 | |
| | FMStocks_DB.DBHelper->RunSPReturnInteger | 2003-02-06 17:07:06.166 | 2003-02-06 17:07:06.170 | 102.1 | 105.4 | 3.3 | 0 | |
| | FMStocks_DB.Broker->Buy | 2003-02-06 17:07:06.170 | 2003-02-06 17:07:06.173 | 105.6 | 108.7 | 3.1 | 0 | |
| | FMStocks_DB.DBHelper->RunSPReturnInteger | 2003-02-06 17:07:06.170 | 2003-02-06 17:07:06.173 | 105.7 | 108.6 | 2.9 | 0 | |
| FMStocks_Bus.Account | | 2003-02-06 17:07:06.274 | 2003-02-06 17:07:06.327 | 0.0 | 53.0 | 53.0 | 0 | |
| | FMStocks_Bus.Account->ListPositionsForSale | 2003-02-06 17:07:06.316 | 2003-02-06 17:07:06.322 | 42.1 | 48.0 | 5.9 | 0 | |
| | FMStocks_DB.Position->ListForSale | 2003-02-06 17:07:06.318 | 2003-02-06 17:07:06.321 | 44.5 | 47.5 | 2.9 | 0 | |
| | FMStocks_DB.DBHelper->RunSPReturnRS | 2003-02-06 17:07:06.318 | 2003-02-06 17:07:06.321 | 44.6 | 47.4 | 2.8 | 0 | |
| FMStocks_Bus.Broker | | 2003-02-06 17:07:06.472 | 2003-02-06 17:07:06.550 | 0.0 | 78.7 | 78.7 | 0 | |
| | FMStocks_Bus.Broker->SellStock | 2003-02-06 17:07:06.511 | 2003-02-06 17:07:06.544 | 39.1 | 72.4 | 33.3 | 0 | |
| | FMStocks_DB.Tx->AddSellOrder | 2003-02-06 17:07:06.519 | 2003-02-06 17:07:06.528 | 47.0 | 56.4 | 9.3 | 0 | |
| | FMStocks_DB.DBHelper->RunSPReturnInteger | 2003-02-06 17:07:06.519 | 2003-02-06 17:07:06.528 | 47.1 | 56.3 | 9.2 | 0 | |
| | FMStocks_DB.Ticker->GetPrice | 2003-02-06 17:07:06.528 | 2003-02-06 17:07:06.532 | 56.6 | 60.1 | 3.5 | 0 | |
| | FMStocks_DB.DBHelper->RunSPReturnInteger | 2003-02-06 17:07:06.528 | 2003-02-06 17:07:06.532 | 56.6 | 60.0 | 3.4 | 0 | |
| | FMStocks_DB.Broker->Sell | 2003-02-06 17:07:06.532 | 2003-02-06 17:07:06.535 | 60.2 | 63.0 | 2.8 | 0 | |
| | FMStocks_DB.DBHelper->RunSPReturnInteger | 2003-02-06 17:07:06.532 | 2003-02-06 17:07:06.534 | 60.3 | 62.9 | 2.6 | 0 | |
| | FMStocks_DB.Position->ListForAdjustment | 2003-02-06 17:07:06.537 | 2003-02-06 17:07:06.542 | 65.3 | 70.8 | 5.5 | 0 | |
| | FMStocks_DB.DBHelper->RunSPReturnRS_RW | 2003-02-06 17:07:06.537 | 2003-02-06 17:07:06.542 | 65.4 | 70.7 | 5.3 | 0 | |

# Business Activities Take Too Long?

## Solution: AppMetrics’ Method Analysis Report



Methods Analysis Report

AppMetrics Sample Diagnostic

2/6/2003 4:07:00 PM to 2/6/2003 4:09:00 PM

The current application is: FMStocks 2000 Core

| Transaction | Methods | | | StartTime | EndTime | Relative Start (ms) | Relative End (ms) | Duration (ms) | Error | Description |
|---|---|---|---|---|---|---|---|---|---|---|
| FMStocks_Bus.Ticker | | | | 2003-02-06 17:07:06.064 | 2003-02-06 17:07:06.180 | 0.0 | 115.3 | 115.3 | 0 | |
| | FMStocks_Bus.Ticker->VerifySymbol | | | 2003-02-06 17:07:06.112 | 2003-02-06 17:07:06.119 | 48.1 | 54.5 | 6.4 | 0 | |
| | | FMStocks_DB.Ticker->VerifySymbol | | 2003-02-06 17:07:06.115 | 2003-02-06 17:07:06.119 | 50.9 | 54.3 | 3.3 | 0 | |
| | | | FMStocks_DB.DBHelper->RunSPReturnInteger | 2003-02-06 17:07:06.115 | 2003-02-06 17:07:06.119 | 51.0 | 54.2 | 3.2 | 0 | |
| | FMStocks_Bus.Broker->BuyStock | | | 2003-02-06 17:07:06.155 | 2003-02-06 17:07:06.173 | 90.2 | 109.0 | 18.8 | 0 | |
| | | FMStocks_DB.Tx->AddBuyOrder | | 2003-02-06 17:07:06.161 | 2003-02-06 17:07:06.166 | 96.2 | 101.9 | 5.7 | 0 | |
| | | | FMStocks_DB.DBHelper->RunSPReturnInteger | 2003-02-06 17:07:06.161 | 2003-02-06 17:07:06.166 | 96.3 | 101.8 | 5.5 | 0 | |
| | | FMStocks_DB.Ticker->GetPrice | | 2003-02-06 17:07:06.166 | 2003-02-06 17:07:06.170 | 102.0 | 105.4 | 3.4 | 0 | |
| | | | FMStocks_DB.DBHelper->RunSPReturnInteger | 2003-02-06 17:07:06.166 | 2003-02-06 17:07:06.170 | 102.1 | 105.4 | 3.3 | 0 | |
| | | FMStocks_DB.Broker->Buy | | 2003-02-06 17:07:06.170 | 2003-02-06 17:07:06.173 | 105.6 | 108.7 | 3.1 | 0 | |
| | | | FMStocks_DB.DBHelper->RunSPReturnInteger | 2003-02-06 17:07:06.170 | 2003-02-06 17:07:06.173 | 105.7 | 108.6 | 2.9 | 0 | |
| FMStocks_Bus.Account | | | | 2003-02-06 17:07:06.274 | 2003-02-06 17:07:06.327 | 0.0 | 53.0 | 53.0 | 0 | |
| | FMStocks_Bus.Account->ListPositionsForSale | | | 2003-02-06 17:07:06.316 | 2003-02-06 17:07:06.322 | 42.1 | 48.0 | 5.9 | 0 | |
| | | FMStocks_DB.Position->ListForSale | | 2003-02-06 17:07:06.318 | 2003-02-06 17:07:06.321 | 44.5 | 47.5 | 2.9 | 0 | |
| | | | FMStocks_DB.DBHelper->RunSPReturnRS | 2003-02-06 17:07:06.318 | 2003-02-06 17:07:06.321 | 44.6 | 47.4 | 2.8 | 0 | |
| FMStocks_Bus.Broker | | | | 2003-02-06 17:07:06.472 | 2003-02-06 17:07:06.550 | 0.0 | 78.7 | 78.7 | 0 | |
| | FMStocks_Bus.Broker->SellStock | | | 2003-02-06 17:07:06.511 | 2003-02-06 17:07:06.544 | 39.1 | 72.4 | 33.3 | 0 | |
| | | FMStocks_DB.Tx->AddSellOrder | | 2003-02-06 17:07:06.519 | 2003-02-06 17:07:06.528 | 47.0 | 56.4 | 9.3 | 0 | |
| | | | FMStocks_DB.DBHelper->RunSPReturnInteger | 2003-02-06 17:07:06.519 | 2003-02-06 17:07:06.528 | 47.1 | 56.3 | 9.2 | 0 | |
| | | FMStocks_DB.Ticker->GetPrice | | 2003-02-06 17:07:06.528 | 2003-02-06 17:07:06.532 | 56.6 | 60.1 | 3.5 | 0 | |
| | | | FMStocks_DB.DBHelper->RunSPReturnInteger | 2003-02-06 17:07:06.528 | 2003-02-06 17:07:06.532 | 56.6 | 60.0 | 3.4 | 0 | |
| | | FMStocks_DB.Broker->Sell | | 2003-02-06 17:07:06.532 | 2003-02-06 17:07:06.535 | 60.2 | 63.0 | 2.8 | 0 | |
| | | | FMStocks_DB.DBHelper->RunSPReturnInteger | 2003-02-06 17:07:06.532 | 2003-02-06 17:07:06.534 | 60.3 | 62.9 | 2.6 | 0 | |
| | | FMStocks_DB.Position->ListForAdjustment | | 2003-02-06 17:07:06.537 | 2003-02-06 17:07:06.542 | 65.3 | 70.8 | 5.5 | 0 | |
| | | | FMStocks_DB.DBHelper->RunSPReturnRS_RW | 2003-02-06 17:07:06.537 | 2003-02-06 17:07:06.542 | 65.4 | 70.7 | 5.3 | 0 | |

- Hierarchy shows call sequence
- Durations for each step of the sequence

# Business Activities Take Too Long?

Our Customers say:

*“Xtremesoft is the only company that translates application logic metrics into business performance information."*

# Where to Optimize?

Problem:

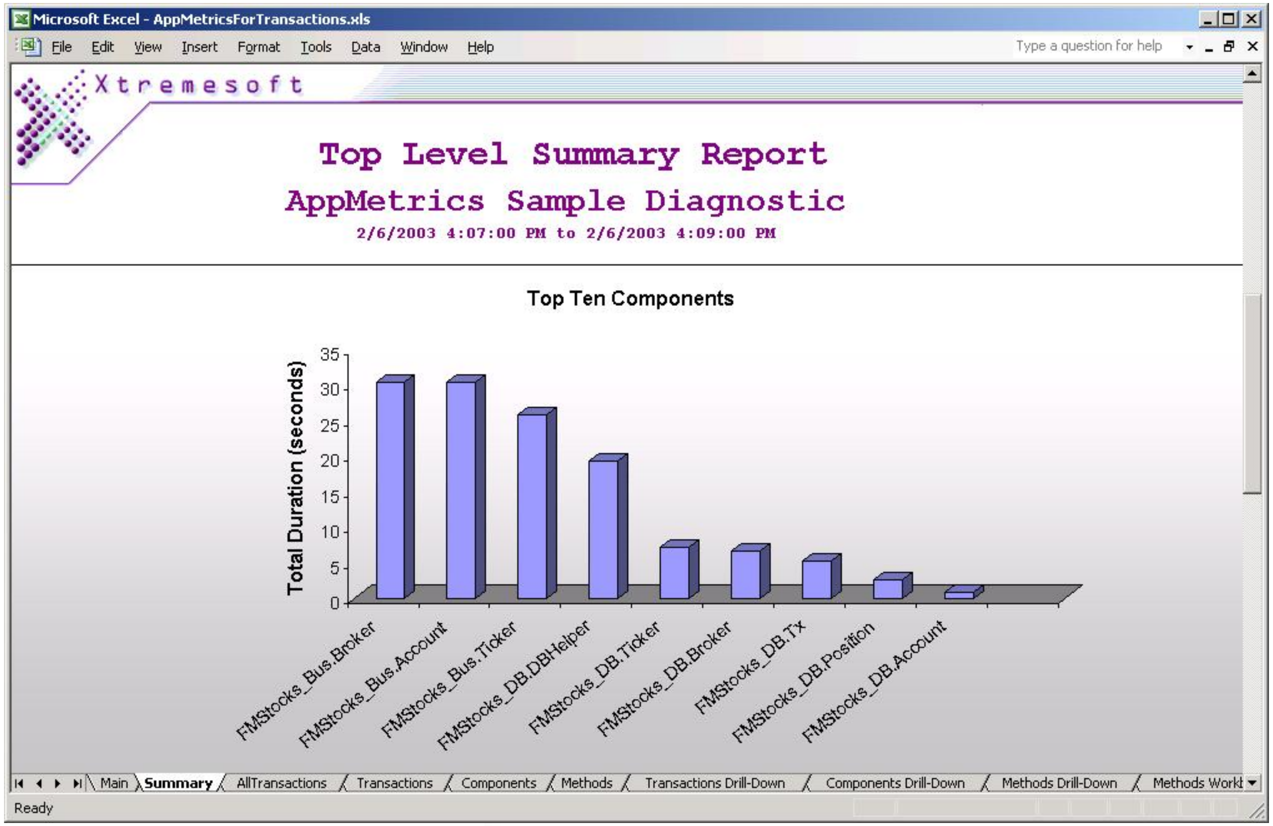
**I need to know my ‘most expensive’ components.**

When told the application is too slow and the application has tens, often hundreds of components, where do you start looking for possible candidates for optimization?

# Where to Optimize?

## Solution: AppMetrics' Top Ten Component Report

This report reveals which components are spending the most time running on the machine. The total duration of all component instances of each component type is calculated, and then the component types are sorted by total duration. This view helps you to choose which components are likely performance problems, because this algorithm will, for example rate a component that runs 100 times for an average of 1 second each time higher than a component that only runs 1 time, but for 50 seconds. The second component may need work, but it is less likely to be the cause of the problem.



Top Level Summary Report

AppMetrics Sample Diagnostic

2/6/2003 4:07:00 PM to 2/6/2003 4:09:00 PM

# Where to Optimize?

Our Customers say:

> *"Each operation originally was taking 5,420,3 ms. After we adjusted the code (following the tips from AppMetrics and DevPartner); each operation took 781,5 ms.* ***It's an amazing result*** *- the original code was about 700% slower compared to the final one!"*

# Inconsistent Performance?

Problem:

**I don’t know my ‘typical’ method durations.**

You have to know,

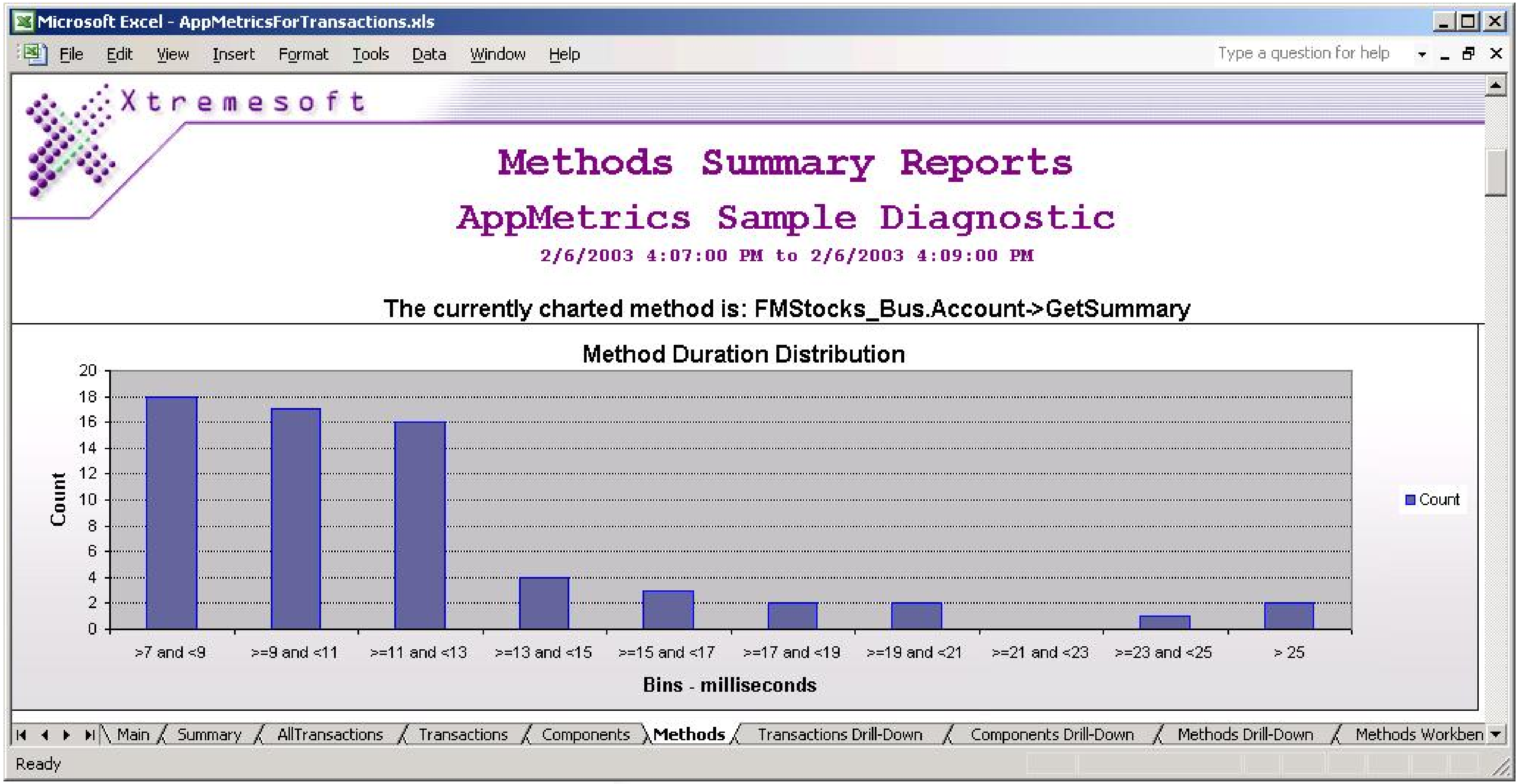
- what is if the average duration metric
- Is it representative of the typical duration, or if the average is being skewed by ‘outlier’ method instances.

If you are to identify the possible performance bottlenecks

# Inconsistent Performance?

## Solution: Method Duration Distribution Report

The Method Duration Distribution Report creates 10 evenly-sized ‘bins’, and displays the count of method instances that fell into each bin during the selected time window.  This provides a view of the typical durations of methods, with an indication of the quantity of outliers -



Methods Summary Reports

AppMetrics Sample Diagnostic

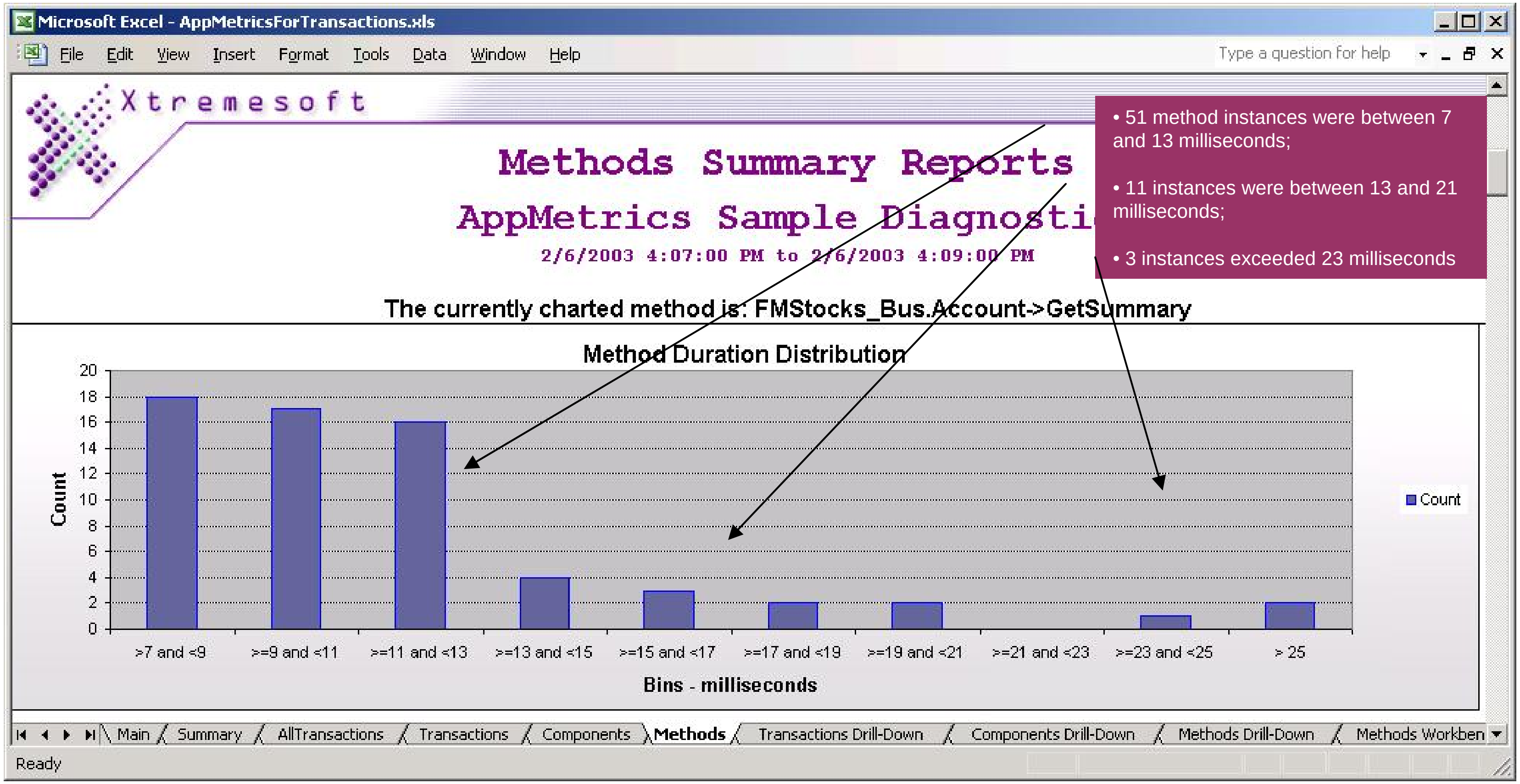
2/6/2003 4:07:00 PM to 2/6/2003 4:09:00 PM

The currently charted method is: FMStocks_Bus.Account->GetSummary

# Inconsistent Performance?

## Solution: Method Duration Distribution Report

The Method Duration Distribution Report creates 10 evenly-sized ‘bins’, and displays the count of method instances that fell into each bin during the selected time window.  This provides a view of the typical method durations of methods in production, with an indication of the quantity of outliers -


Microsoft Excel - AppMetricsForTransactions.xls
File Edit View Insert Format Tools Data Window Help
Type a question for help
Xtremesoft

Methods Summary Reports

AppMetrics Sample Diagnosti

2/6/2003 4:07:00 PM to 2/6/2003 4:09:00 PM

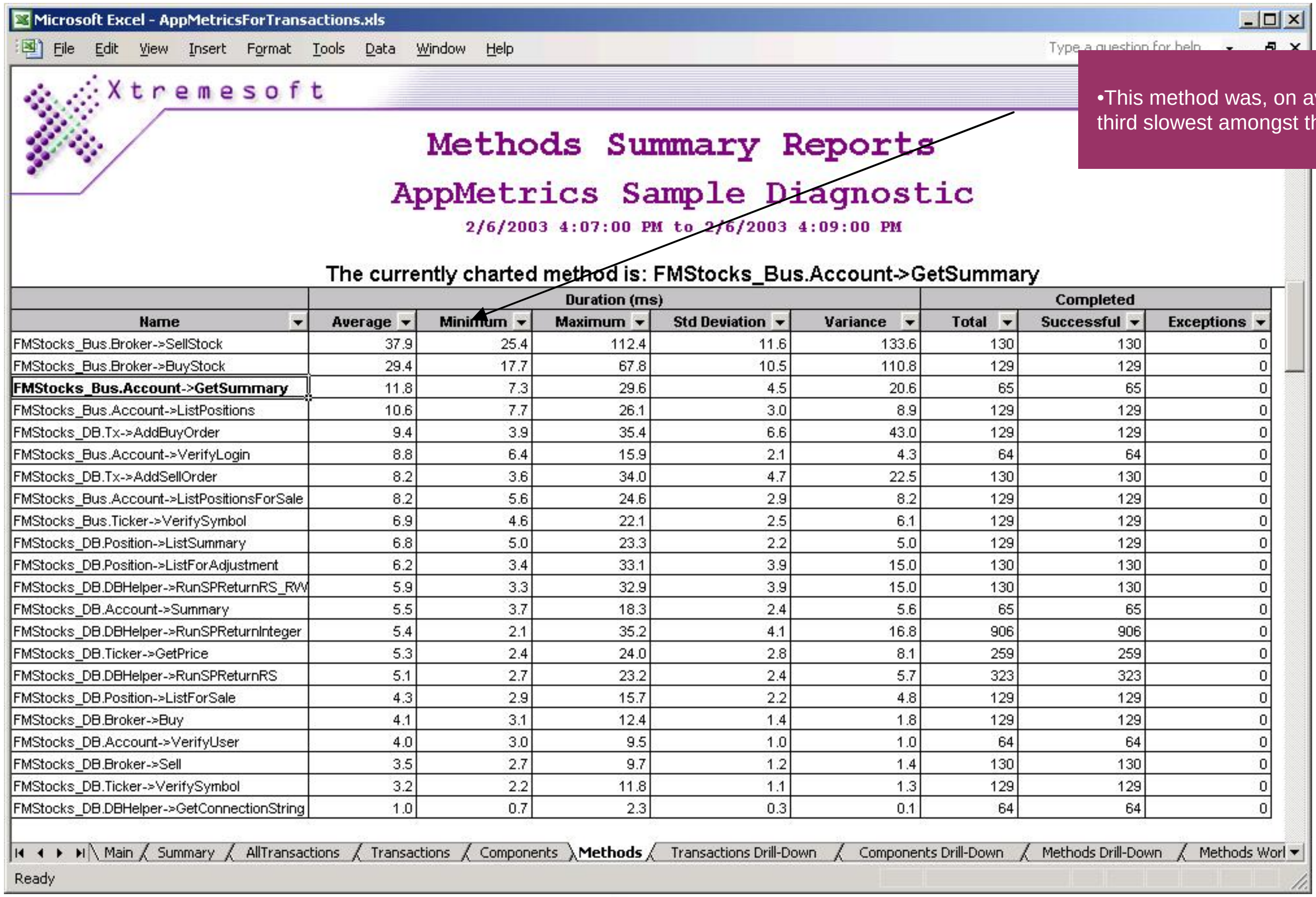
The currently charted method is: FMStocks_Bus.Account->GetSummary

- 51 method instances were between 7 and 13 milliseconds;
- 11 instances were between 13 and 21 milliseconds;
- 3 instances exceeded 23 milliseconds

# Inconsistent Performance?

## Solution: Method Aggregate Report

A variety of aggregate metrics are supplied for each of the observed methods during the test window.  Here the report is sorted by Average Duration.



Methods Summary Reports

AppMetrics Sample Diagnostic

2/6/2003 4:07:00 PM to 2/6/2003 4:09:00 PM

The currently charted method is: FMStocks_Bus.Account->GetSummary

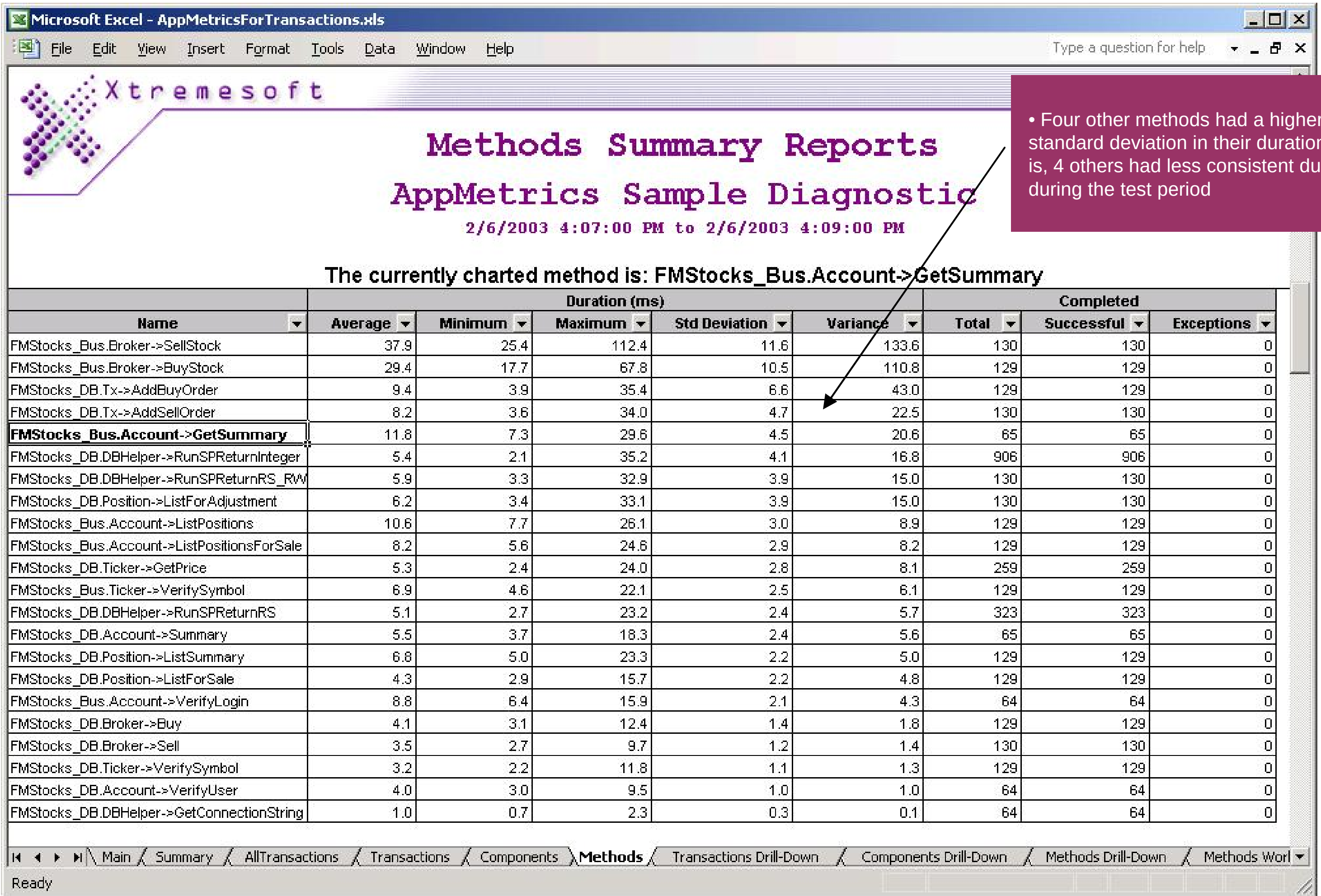
| Name | Duration (ms) | | | | | Completed | | |
|---|---|---|---|---|---|---|---|---|
| | Average | Minimum | Maximum | Std Deviation | Variance | Total | Successful | Exceptions |
| FMStocks_Bus.Broker->SellStock | 37.9 | 25.4 | 112.4 | 11.6 | 133.6 | 130 | 130 | 0 |
| FMStocks_Bus.Broker->BuyStock | 29.4 | 17.7 | 67.8 | 10.5 | 110.8 | 129 | 129 | 0 |
| **FMStocks_Bus.Account->GetSummary** | 11.8 | 7.3 | 29.6 | 4.5 | 20.6 | 65 | 65 | 0 |
| FMStocks_Bus.Account->ListPositions | 10.6 | 7.7 | 26.1 | 3.0 | 8.9 | 129 | 129 | 0 |
| FMStocks_DB.Tx->AddBuyOrder | 9.4 | 3.9 | 35.4 | 6.6 | 43.0 | 129 | 129 | 0 |
| FMStocks_Bus.Account->VerifyLogin | 8.8 | 6.4 | 15.9 | 2.1 | 4.3 | 64 | 64 | 0 |
| FMStocks_DB.Tx->AddSellOrder | 8.2 | 3.6 | 34.0 | 4.7 | 22.5 | 130 | 130 | 0 |
| FMStocks_Bus.Account->ListPositionsForSale | 8.2 | 5.6 | 24.6 | 2.9 | 8.2 | 129 | 129 | 0 |
| FMStocks_Bus.Ticker->VerifySymbol | 6.9 | 4.6 | 22.1 | 2.5 | 6.1 | 129 | 129 | 0 |
| FMStocks_DB.Position->ListSummary | 6.8 | 5.0 | 23.3 | 2.2 | 5.0 | 129 | 129 | 0 |
| FMStocks_DB.Position->ListForAdjustment | 6.2 | 3.4 | 33.1 | 3.9 | 15.0 | 130 | 130 | 0 |
| FMStocks_DB.DBHelper->RunSPReturnRS_RW | 5.9 | 3.3 | 32.9 | 3.9 | 15.0 | 130 | 130 | 0 |
| FMStocks_DB.Account->Summary | 5.5 | 3.7 | 18.3 | 2.4 | 5.6 | 65 | 65 | 0 |
| FMStocks_DB.DBHelper->RunSPReturnInteger | 5.4 | 2.1 | 35.2 | 4.1 | 16.8 | 906 | 906 | 0 |
| FMStocks_DB.Ticker->GetPrice | 5.3 | 2.4 | 24.0 | 2.8 | 8.1 | 259 | 259 | 0 |
| FMStocks_DB.DBHelper->RunSPReturnRS | 5.1 | 2.7 | 23.2 | 2.4 | 5.7 | 323 | 323 | 0 |
| FMStocks_DB.Position->ListForSale | 4.3 | 2.9 | 15.7 | 2.2 | 4.8 | 129 | 129 | 0 |
| FMStocks_DB.Broker->Buy | 4.1 | 3.1 | 12.4 | 1.4 | 1.8 | 129 | 129 | 0 |
| FMStocks_DB.Account->VerifyUser | 4.0 | 3.0 | 9.5 | 1.0 | 1.0 | 64 | 64 | 0 |
| FMStocks_DB.Broker->Sell | 3.5 | 2.7 | 9.7 | 1.2 | 1.4 | 130 | 130 | 0 |
| FMStocks_DB.Ticker->VerifySymbol | 3.2 | 2.2 | 11.8 | 1.1 | 1.3 | 129 | 129 | 0 |
| FMStocks_DB.DBHelper->GetConnectionString | 1.0 | 0.7 | 2.3 | 0.3 | 0.1 | 64 | 64 | 0 |

- This method was, on average, the third slowest amongst those observed

# Inconsistent Performance?

## Solution: Method Aggregate Report

A variety of aggregate metrics are supplied for each of the observed methods during the test window. Here the report is sorted by Standard Deviation



Methods Summary Reports

AppMetrics Sample Diagnostic

2/6/2003 4:07:00 PM to 2/6/2003 4:09:00 PM

The currently charted method is: FMStocks_Bus.Account->GetSummary

| | Duration (ms) | | | | | Completed | | |
|---|---|---|---|---|---|---|---|---|
| Name | Average | Minimum | Maximum | Std Deviation | Variance | Total | Successful | Exceptions |
| FMStocks_Bus.Broker->SellStock | 37.9 | 25.4 | 112.4 | 11.6 | 133.6 | 130 | 130 | 0 |
| FMStocks_Bus.Broker->BuyStock | 29.4 | 17.7 | 67.8 | 10.5 | 110.8 | 129 | 129 | 0 |
| FMStocks_DB.Tx->AddBuyOrder | 9.4 | 3.9 | 35.4 | 6.6 | 43.0 | 129 | 129 | 0 |
| FMStocks_DB.Tx->AddSellOrder | 8.2 | 3.6 | 34.0 | 4.7 | 22.5 | 130 | 130 | 0 |
| **FMStocks_Bus.Account->GetSummary** | 11.8 | 7.3 | 29.6 | 4.5 | 20.6 | 65 | 65 | 0 |
| FMStocks_DB.DBHelper->RunSPReturnInteger | 5.4 | 2.1 | 35.2 | 4.1 | 16.8 | 906 | 906 | 0 |
| FMStocks_DB.DBHelper->RunSPReturnRS_RW | 5.9 | 3.3 | 32.9 | 3.9 | 15.0 | 130 | 130 | 0 |
| FMStocks_DB.Position->ListForAdjustment | 6.2 | 3.4 | 33.1 | 3.9 | 15.0 | 130 | 130 | 0 |
| FMStocks_Bus.Account->ListPositions | 10.6 | 7.7 | 26.1 | 3.0 | 8.9 | 129 | 129 | 0 |
| FMStocks_Bus.Account->ListPositionsForSale | 8.2 | 5.6 | 24.6 | 2.9 | 8.2 | 129 | 129 | 0 |
| FMStocks_DB.Ticker->GetPrice | 5.3 | 2.4 | 24.0 | 2.8 | 8.1 | 259 | 259 | 0 |
| FMStocks_Bus.Ticker->VerifySymbol | 6.9 | 4.6 | 22.1 | 2.5 | 6.1 | 129 | 129 | 0 |
| FMStocks_DB.DBHelper->RunSPReturnRS | 5.1 | 2.7 | 23.2 | 2.4 | 5.7 | 323 | 323 | 0 |
| FMStocks_DB.Account->Summary | 5.5 | 3.7 | 18.3 | 2.4 | 5.6 | 65 | 65 | 0 |
| FMStocks_DB.Position->ListSummary | 6.8 | 5.0 | 23.3 | 2.2 | 5.0 | 129 | 129 | 0 |
| FMStocks_DB.Position->ListForSale | 4.3 | 2.9 | 15.7 | 2.2 | 4.8 | 129 | 129 | 0 |
| FMStocks_Bus.Account->VerifyLogin | 8.8 | 6.4 | 15.9 | 2.1 | 4.3 | 64 | 64 | 0 |
| FMStocks_DB.Broker->Buy | 4.1 | 3.1 | 12.4 | 1.4 | 1.8 | 129 | 129 | 0 |
| FMStocks_DB.Broker->Sell | 3.5 | 2.7 | 9.7 | 1.2 | 1.4 | 130 | 130 | 0 |
| FMStocks_DB.Ticker->VerifySymbol | 3.2 | 2.2 | 11.8 | 1.1 | 1.3 | 129 | 129 | 0 |
| FMStocks_DB.Account->VerifyUser | 4.0 | 3.0 | 9.5 | 1.0 | 1.0 | 64 | 64 | 0 |
| FMStocks_DB.DBHelper->GetConnectionString | 1.0 | 0.7 | 2.3 | 0.3 | 0.1 | 64 | 64 | 0 |

- Four other methods had a higher standard deviation in their durations, that is, 4 others had less consistent durations during the test period

# Inconsistent Performance?

Our Customers tell us that:

> *"We accomplished a BIG performance gain - the final code runs 7 times faster (about 14%) than the original one."*

# Hung Components?

Problem: **A COM+ Application is ‘hung’**

A customer calls or emails (usually at night) to complain that the web site is ‘frozen’. It’s time to restart the COM+ Application….again!

*Wouldn’t you like an automated component monitor and recovery solution?*

# Hung Components?

**Real Time Detection**

- Xtremesoft AppMetrics monitors component instances in real-time
- Component instance durations are compared to user-defined benchmarks
- AppMetrics detects running instances whose durations exceed their benchmarks, and creates Windows Event Viewer, SMTP or SNMP alerts
- Appmetrics can run a predefined script to recover the application automatically Custom Component – enables *automated* responses to events

# Hung Components?

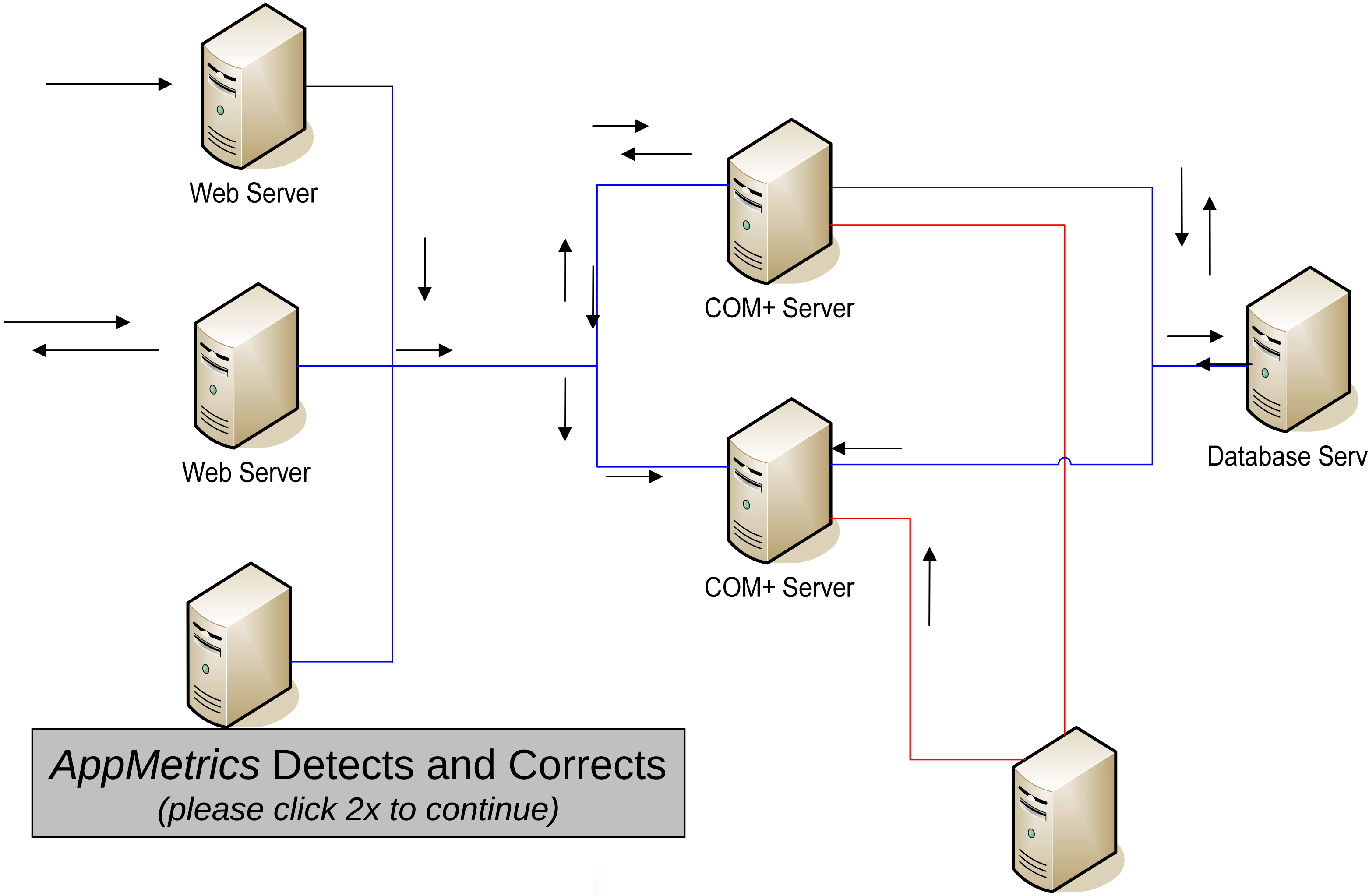
- Custom Component

Receives all the information concerning the AppMetrics Event, including:

- The name of the server with the hung COM+ component
- The name of the COM+ Application that contains the hung component
- Enabling code such as this script:

```
set oCat = createObject("COMAdmin.COMAdminCatalog")
oCat.connect appServerName 'server where the COM+ component is hung
oCat.ShutdownApplication appName 'the COM+ Application
```

# Automated Application Shutdown



*AppMetrics* Detects and Corrects

*(please click 2x to continue)*

# Let AppMetrics Do It

- AppMetrics can tell when a component is hung
- The AppMetrics Custom Component event handler can restart your application automatically – minimizing application down time

**Our Customers tell us that:**

*"I wanted to let everyone know the AppMetrics is working wonderfully. It is monitoring our systems for hung components 7x24 and gives us automatic recovery. These hangs have been costing us real dollars."*

# Leaky Memory?

Problem:

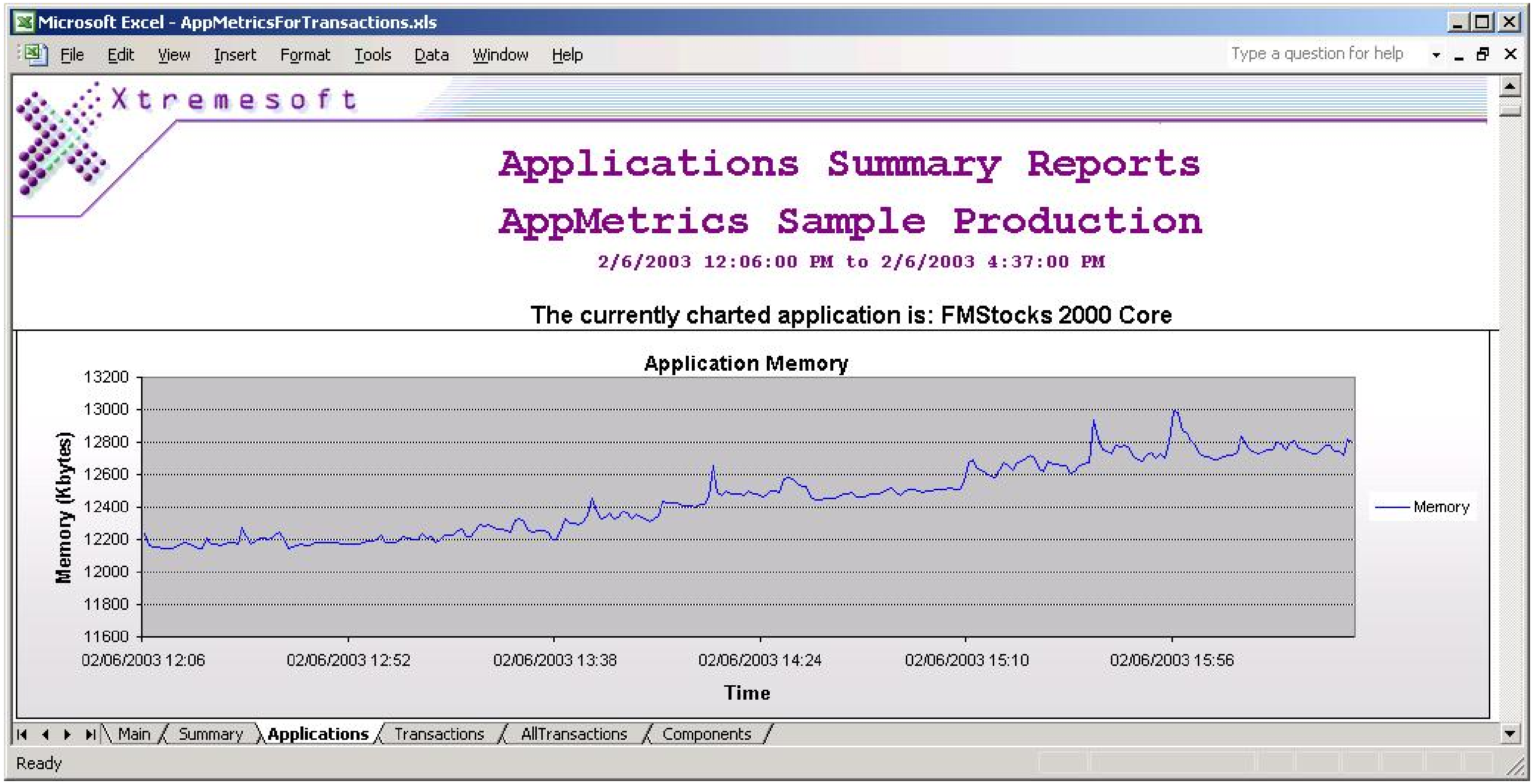
I can’t tell which of my COM+ applications are leaking memory**.**

- Although the Windows Task Manager and Performance Monitor can show memory usage based on dllhost.exe, memory and other counters are not displayed using the application name you gave your COM+ application.  This is a problem when you have dozens or hundreds of dllhost.exe to monitor.  Furthermore, it requires some effort to collect and report on the information over time.

# Leaky Memory?

### Solution: AppMetrics' Application Summary Report

This report includes all the resource metrics associated with COM+ Applications, over the time period requested. In particular, the memory that was consumed by the application during the selected period. Below you can see a steadily increasing amount of memory being consumed by the Core application. Included on the same page of this report are %CPU, Threads, and Page Faults for the selected COM+ application.



Microsoft Excel - AppMetricsForTransactions.xls

Applications Summary Reports

AppMetrics Sample Production

2/6/2003 12:06:00 PM to 2/6/2003 4:37:00 PM

The currently charted application is: FMStocks 2000 Core

# Leaky Memory?

What our Customers are saying:

> *"By using AppMetrics, the availability of the applicatio has increased by several percentage points. Considering for every one percentage point of additional availability 1000 man-hours of down-time are saved--this has resulted in millions of dollars in improved efficiency throughout our organization."*

# Finger–pointing Meetings?

Problem:

A multi-tier application slows down. The Team Leader calls a meeting. “What’s causing the slowdown?” Each attendee points to the person on the right!

- How do you stop the finger-pointing?
- How do you eliminate these meetings?
- How do you identify the root cause?

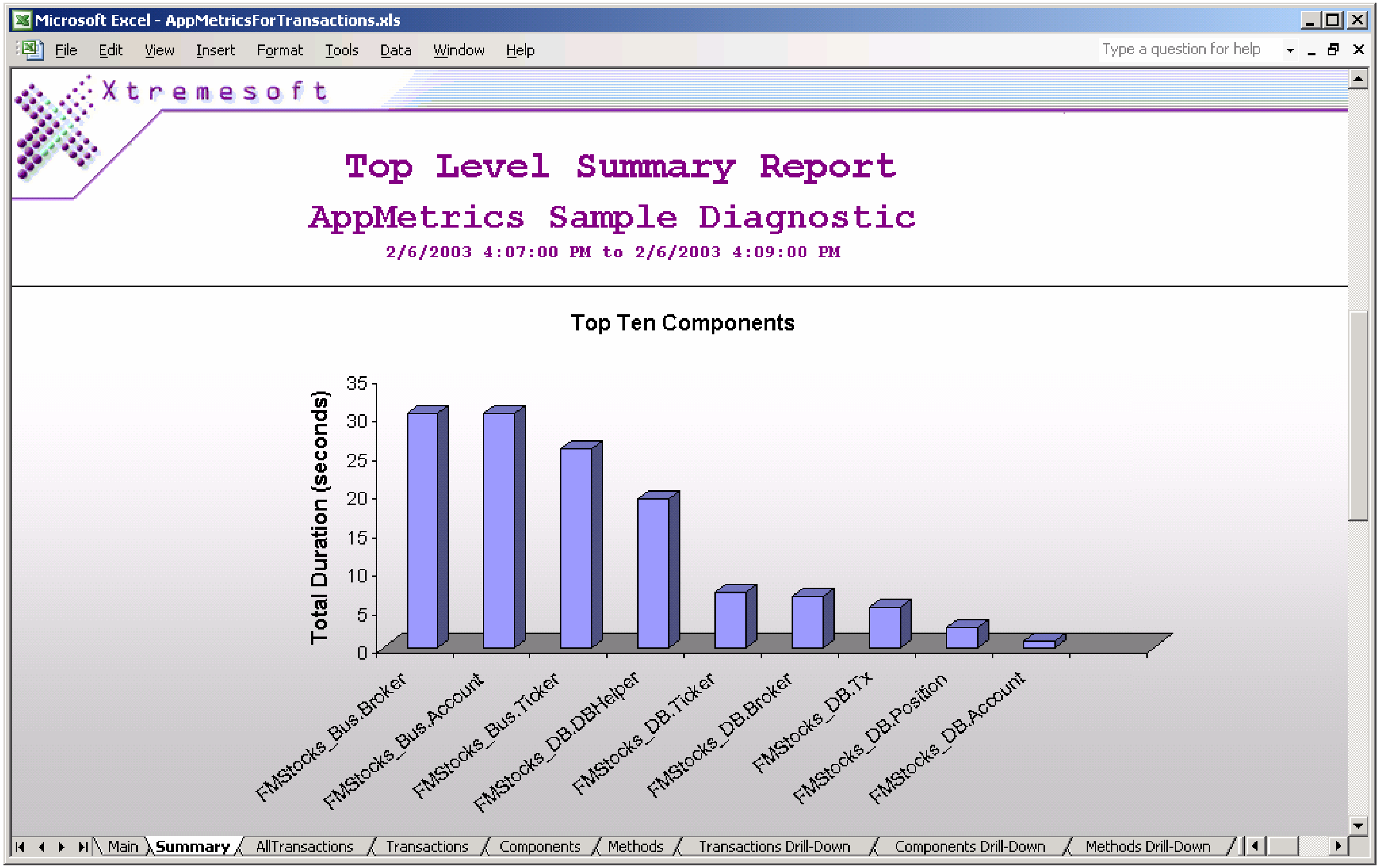
# Break the Cycle

- The Web team blames the mid-tier
- The mid-tier team blames the database
- The database team blames the network
- The Network team blames the web server
- Repeat

# AppMetrics Historical Metrics

- You can identify the responsible module…
- You can compare:
  - same component, different backend
  - same load, different software version
  - same database, new web page
  - And so forth…
- …and eliminate those modules (and team members) whose performance has not changed…

# If it looked like this last time…



Microsoft Excel - AppMetricsForTransactions.xls

Xtremesoft

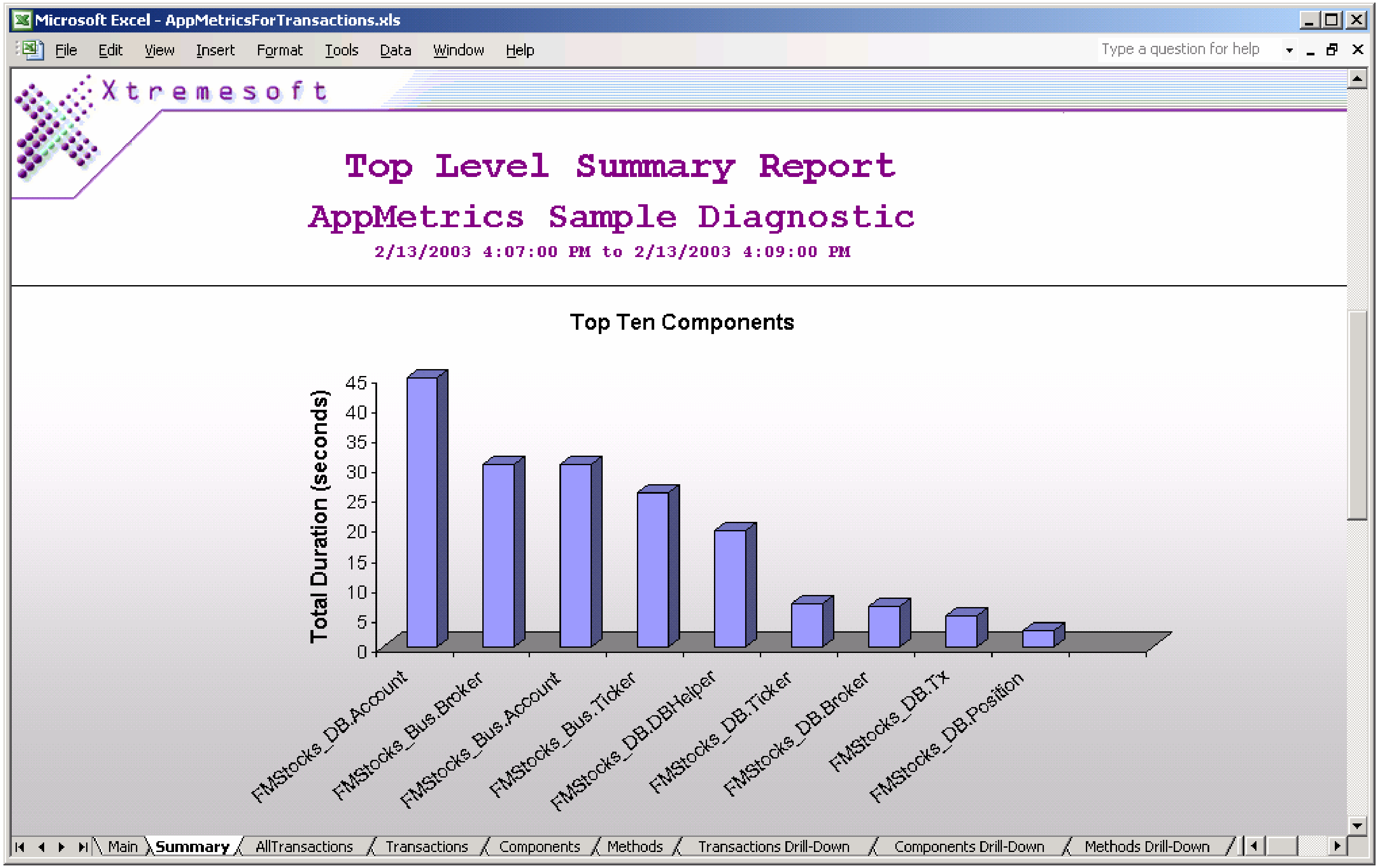
Top Level Summary Report

AppMetrics Sample Diagnostic

2/6/2003 4:07:00 PM to 2/6/2003 4:09:00 PM

Top Ten Components

# why does it look like this now?



## Top Level Summary Report

## AppMetrics Sample Diagnostic

2/13/2003 4:07:00 PM to 2/13/2003 4:09:00 PM

### Top Ten Components

# Eliminate Unnecessary Participants

- By identifying where performance has changed, you can focus on the likely root causes.
- This is true in all software phases:
  - Development, testing before Check-in
  - Quality Assurance, verifying before Release
  - Operations, resolving production problems

# Finger-pointing Meetings?

What our Customers are saying:

*"You guys are awesome! In less than 5 minutes, these drilldown reports traced the exact component and method that were giving me trouble - we saved weeks."*

# Need Early Warning?

Problem:

- Your website is slow.
- Customers Complain.
- Marketing calls Operations.
- Operations Calls you.

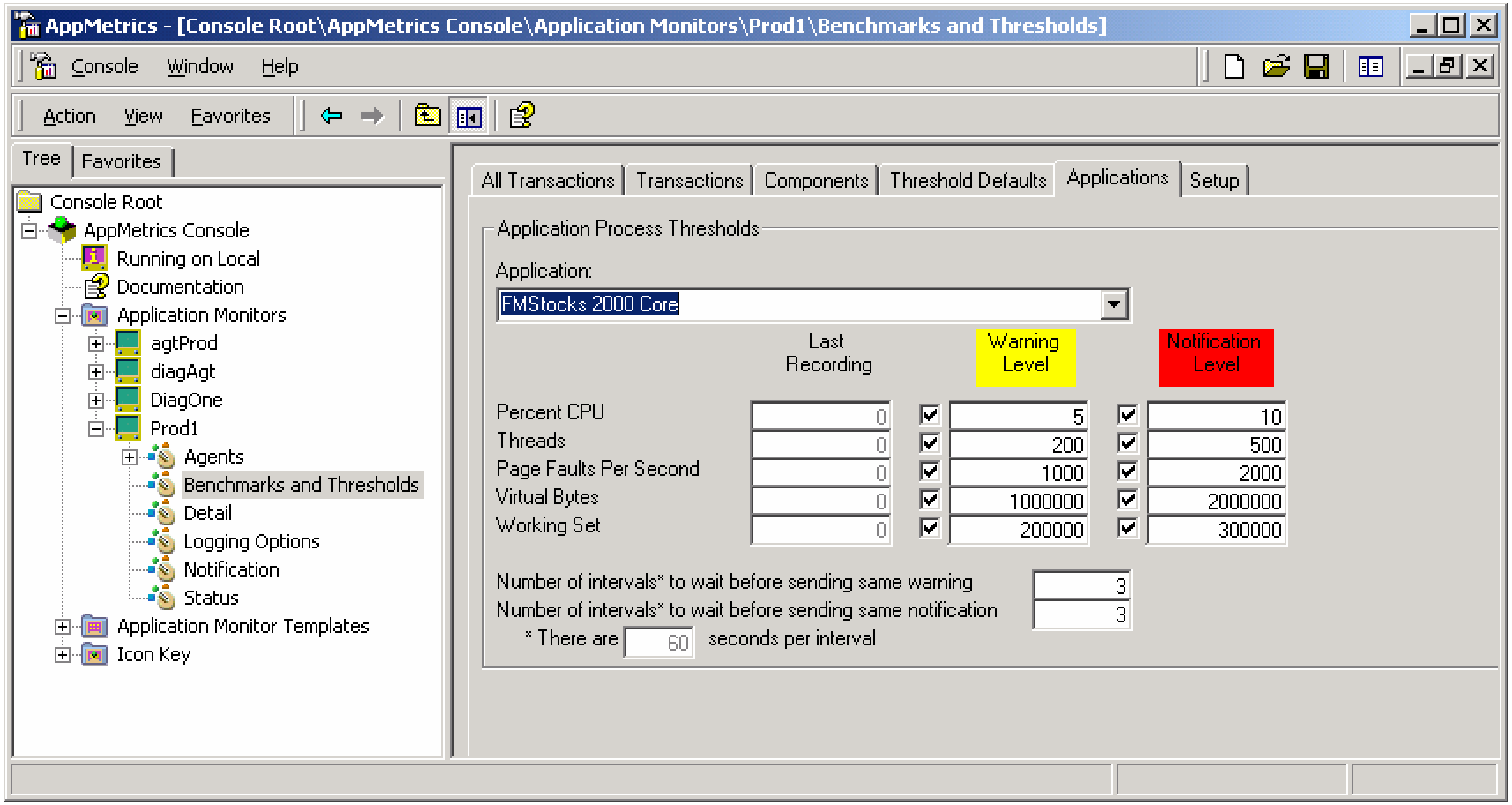
Wouldn’t you like to be the first to know when there’s a problem?

# Realtime Problem Detection

Xtremesoft AppMetrics® monitors your application in real-time, and compares it to your established Benchmarks:

- COM+ Application Resources
- Transaction Metrics
- Component Metrics

# Application Resource Benchmarks



Application Process Thresholds

Application: FMStocks 2000 Core

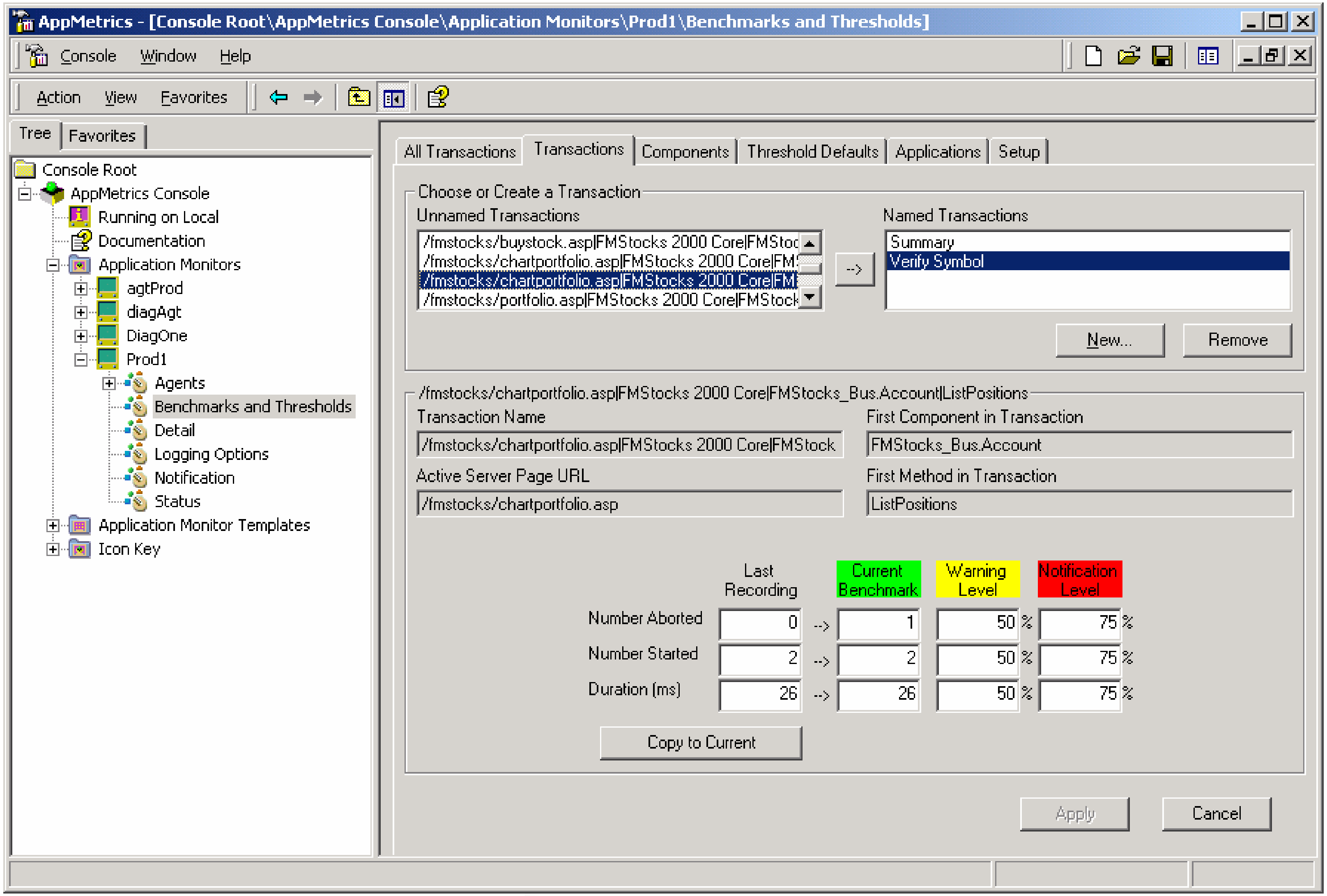
| | Last Recording | | Warning Level | | Notification Level |
|---|---|---|---|---|---|
| Percent CPU | 0 | ☑ | 5 | ☑ | 10 |
| Threads | 0 | ☑ | 200 | ☑ | 500 |
| Page Faults Per Second | 0 | ☑ | 1000 | ☑ | 2000 |
| Virtual Bytes | 0 | ☑ | 1000000 | ☑ | 2000000 |
| Working Set | 0 | ☑ | 200000 | ☑ | 300000 |

Number of intervals* to wait before sending same warning: 3

Number of intervals* to wait before sending same notification: 3

* There are 60 seconds per interval

# Transaction Benchmarks



Choose or Create a Transaction

Unnamed Transactions

/fmstocks/buystock.asp|FMStocks 2000 Core|FMStoc
/fmstocks/chartportfolio.asp|FMStocks 2000 Core|FM
/fmstocks/chartportfolio.asp|FMStocks 2000 Core|FM
/fmstocks/portfolio.asp|FMStocks 2000 Core|FMStock

Named Transactions

Summary
Verify Symbol

/fmstocks/chartportfolio.asp|FMStocks 2000 Core|FMStocks_Bus.Account|ListPositions

Transaction Name: /fmstocks/chartportfolio.asp|FMStocks 2000 Core|FMStock

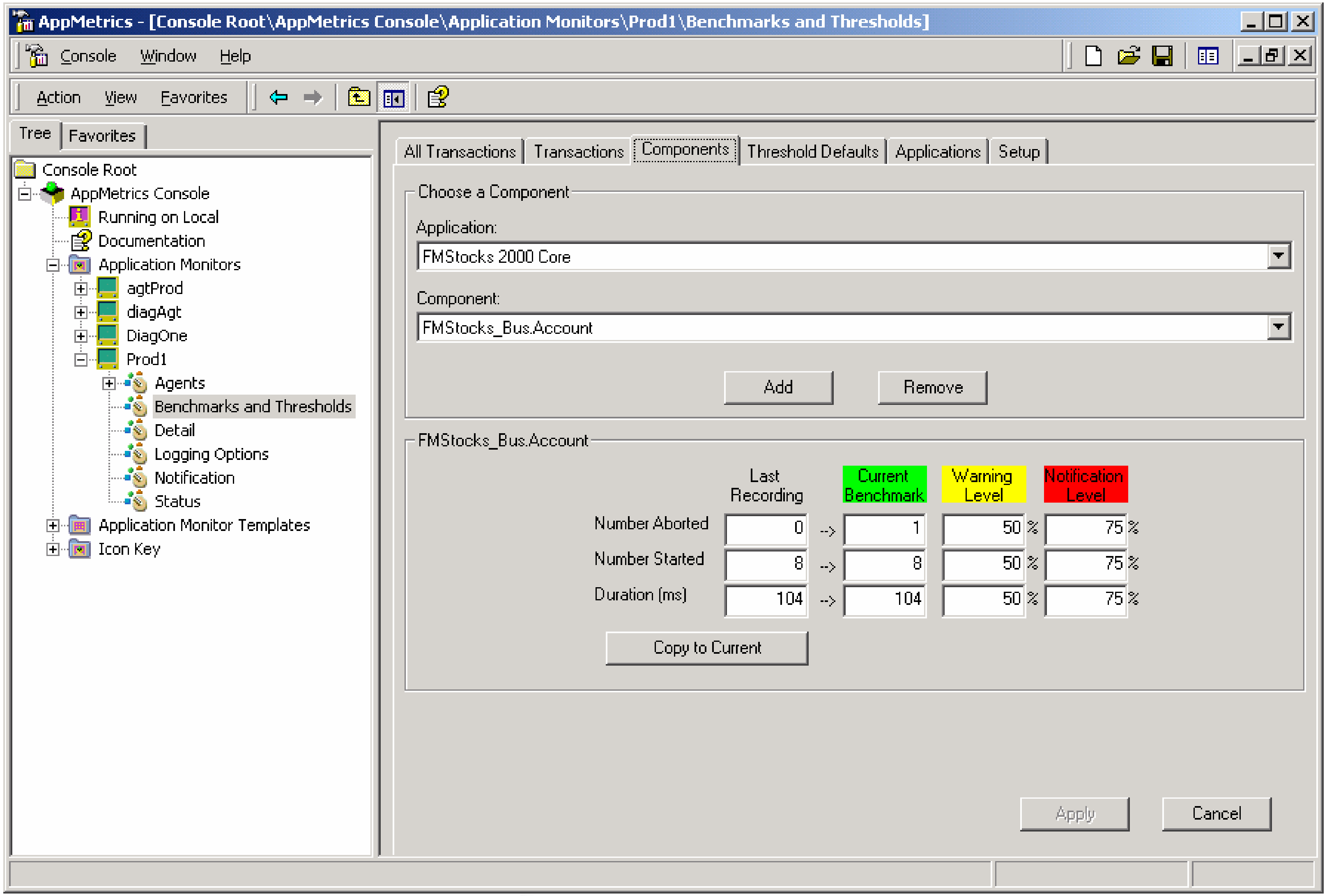
First Component in Transaction: FMStocks_Bus.Account

Active Server Page URL: /fmstocks/chartportfolio.asp

First Method in Transaction: ListPositions

| | Last Recording | | Current Benchmark | Warning Level | Notification Level |
|---|---|---|---|---|---|
| Number Aborted | 0 | --> | 1 | 50 % | 75 % |
| Number Started | 2 | --> | 2 | 50 % | 75 % |
| Duration (ms) | 26 | --> | 26 | 50 % | 75 % |

# Component Benchmarks



AppMetrics - [Console Root\AppMetrics Console\Application Monitors\Prod1\Benchmarks and Thresholds]

Choose a Component

Application: FMStocks 2000 Core

Component: FMStocks_Bus.Account

FMStocks_Bus.Account

| | Last Recording | | Current Benchmark | Warning Level | Notification Level |
|---|---|---|---|---|---|
| Number Aborted | 0 | --> | 1 | 50 % | 75 % |
| Number Started | 8 | --> | 8 | 50 % | 75 % |
| Duration (ms) | 104 | --> | 104 | 50 % | 75 % |

# If Metric > Benchmark then Alert

Alerts are published via Notifications:

- Windows Event Viewer
- SMTP (e-mail, pager)
- SNMP (Traditional Management Frameworks)
- Custom Component (automated response)

AppMetrics detects the problem notifies you by

- e-mail or pager
- Your Management Console, via Windows Event Viewer or SNMP

- AppMetrics can also publish alerts to your custom component, enabling automated ‘detect and correct’ scenarios.

# Need Early Warning?

Our Customers tell us that:

> *"By using AppMetrics, the availability of the application has increased by several percentage points. Considering for every one percentage point of additional availability 1000 man-hours of down-time are saved--this has resulted in millions of dollars in improved efficiency throughout our organization."*

# Conclusion

- A variety of common COM+ scenarios are quickly and easily addressed with AppMetrics
- Increased Application uptime increases return on investment in that application
- Staff Productivity increases because the time to resolve problems is greatly reduced.