Appendix F

Sample Reports Overview

AppMetrics for Transactions includes two sample databases intended to demonstrate the various out-of-box reports generated by AppMetrics. The data sets illustrate some of the different scenarios where AppMetrics reporting can quickly add value to your Development or Operations toolbox.

Customers use AppMetrics reports for:

- Bottleneck detection
- Problem diagnosis
- Capacity planning
- Demonstrating service level agreements (SLAs)

This document is provided a complement to the AppMetrics product, although independently it does provide additional value to the benefits of the product.

As you review the reports, consider your *own* application. With AppMetrics, such in-depth data could be easily collected and used to gain a whole new perspective on the internals of your COM+ middle tier.

To minimize disk space, the collection time was limited to only a few hours. Keep in mind that in production you can store much larger periods of data and AppMetrics can optionally be configured to trim old data – so you only store what you need.

Also note that this document describes a subset of the reports you will see when using the sample databases. This document is intended primarily to accelerate your ability to use the reporting system, and secondarily to describe some of the reports that come with the product.

Installation

AppMetrics for Transactions installs and registers two sample databases during AppMetrics Manager Installation:

- **AppMetrics_Sample_Production**
- **AppMetrics_Sample_Diagnostics**

No additional installation or configuration should be required. If you experience difficulties accessing these samples please contact Xtremesoft Customer Support using the information provided at the end of this document.

Exploring the Sample Production Reports

With production monitoring in AppMetrics data is collected from MTS/COM+ in a stream of events, processed, and then recorded at a default interval of sixty seconds.

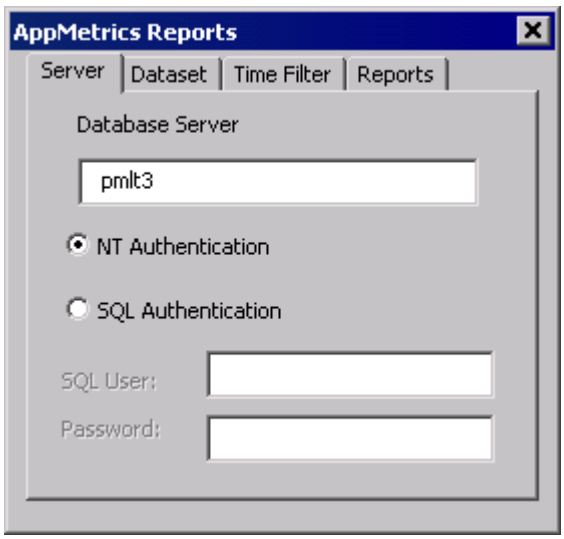
Data collected for day-to-day production monitoring is less granular than data collected for diagnostics monitoring, but the impact on the system is more suited for production environments.

The Sample Production database reports include four hours of collected data. The number of simulated users driving load was increased during the latter portion of the monitored period.

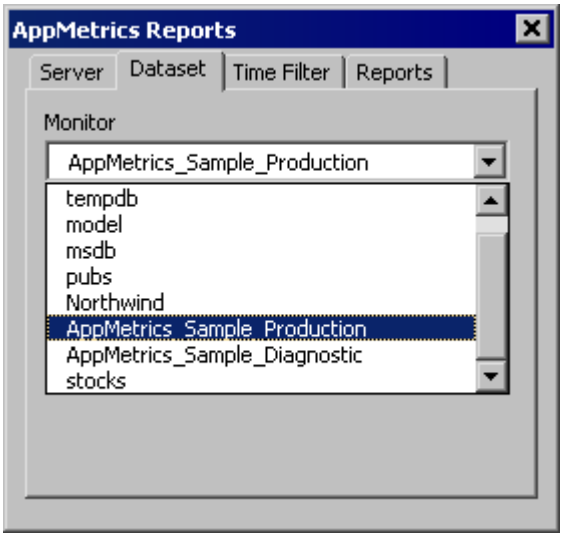
To view the Sample Production database reports:

From the **Start** menu, select **Programs ->Xtremesoft ->AppMetrics for Transactions ->AppMetrics Reports**.

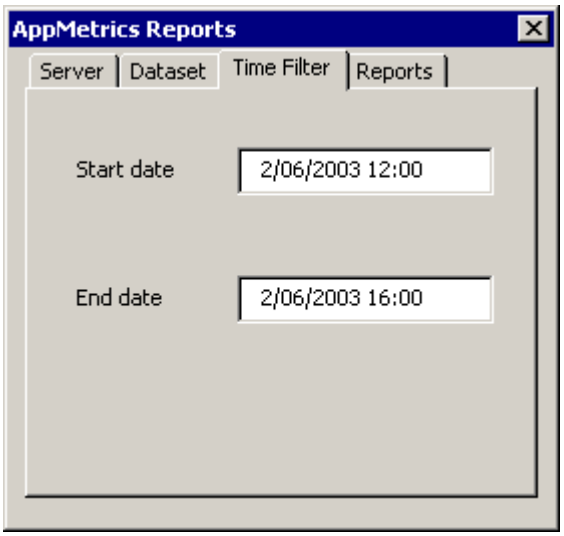
Click any prompt to enable macros. Within the Excel window, the AppMetrics Reports dialog opens.



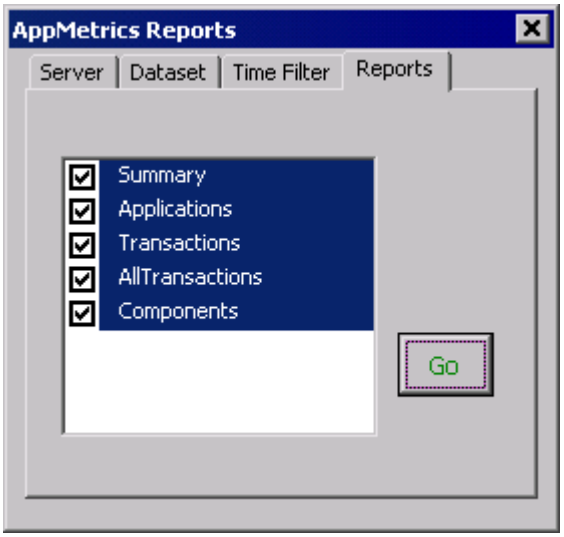
1. **Dataset** and select the **Sample Production** database.



2. Click **Time Filter**. Accept the default **Start date** and **End date**. These correspond to the range of data contained within the database. (Please note: the *only* data in the sample database corresponds to the date and time displayed here. Selecting other timeframes will result in a series of messages about the lack of data.)

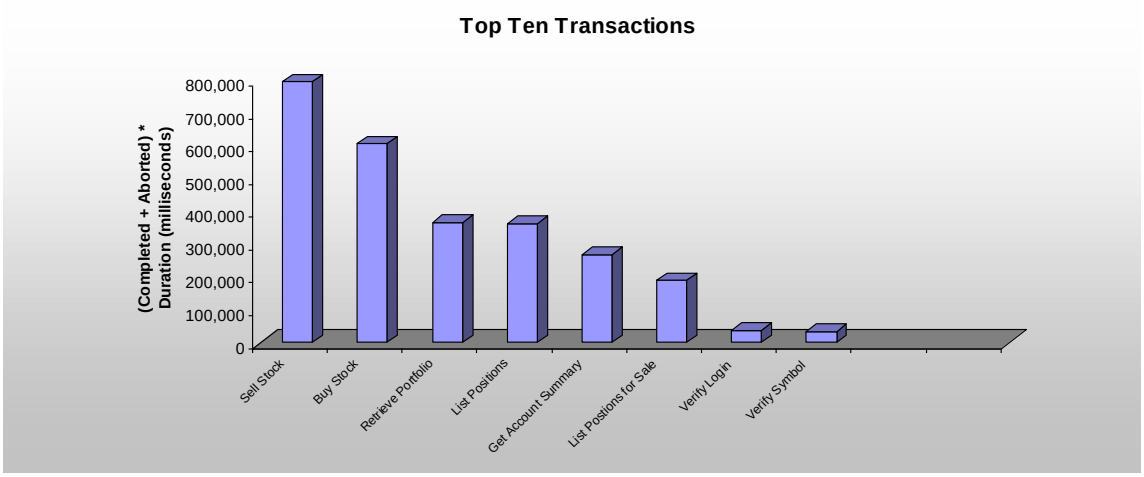


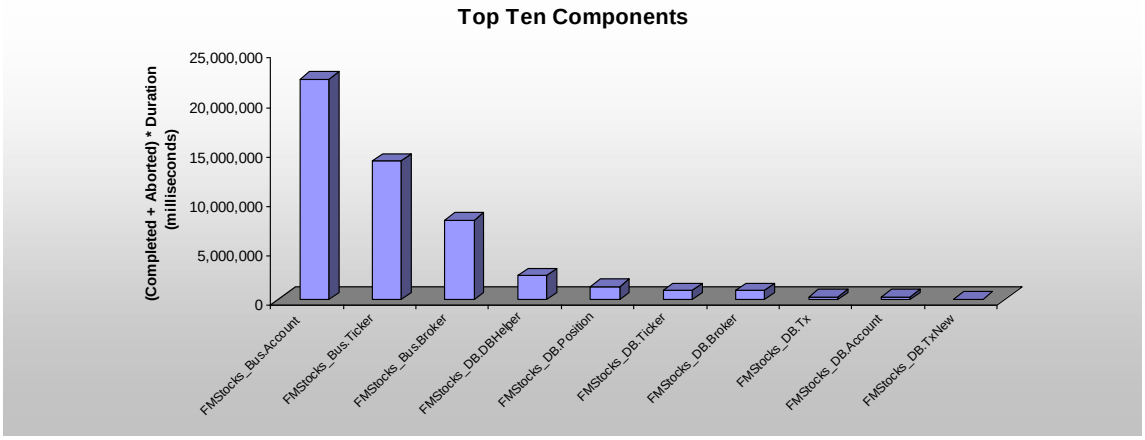
3. Click **Reports** and click **Go**.



Sample Production Reports

You will be presented with a pair of Charts that look like this:





The two charts share much in common. Each presents you with insight concerning throughput within the COM+ tier of your application server.

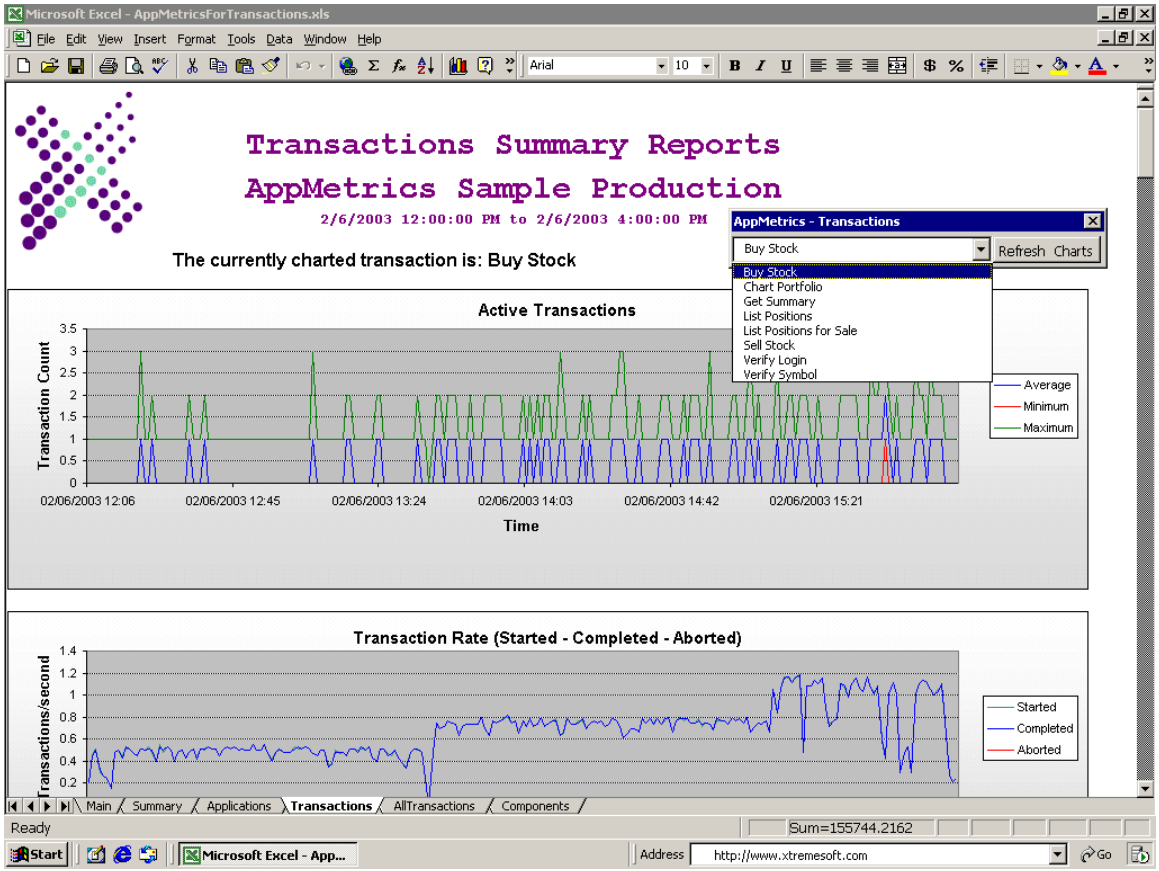
The first chart gives you insight into which transactions are busiest on your servers. The names of the transactions displayed here are what AppMetrics designates 'Named Transactions'. The names of the transactions are supplied by the user, if desired; otherwise AppMetrics uses the URL of the calling ASP, in the form of **/website/active server page|component|method**.

AppMetrics observes the number of instances of each transaction type on your server during the reporting period, and multiplies that by average duration for each of the transaction instances. The product of the count of the instances times the duration of the instances is used to sort the transaction types. The second chart charts the same type of information, but for each of the components on your server.

Armed with this information you make better decisions about where to put your application optimization resources. Let's take a look at one of the left-most transactions, BuyStock.

Transaction Metrics

Once you identify a transaction that you want to know more about, you should select the Transactions tab at the bottom of the reports. Using the drop down in the Transactions picker, select the transaction of interest, and click Refresh charts.

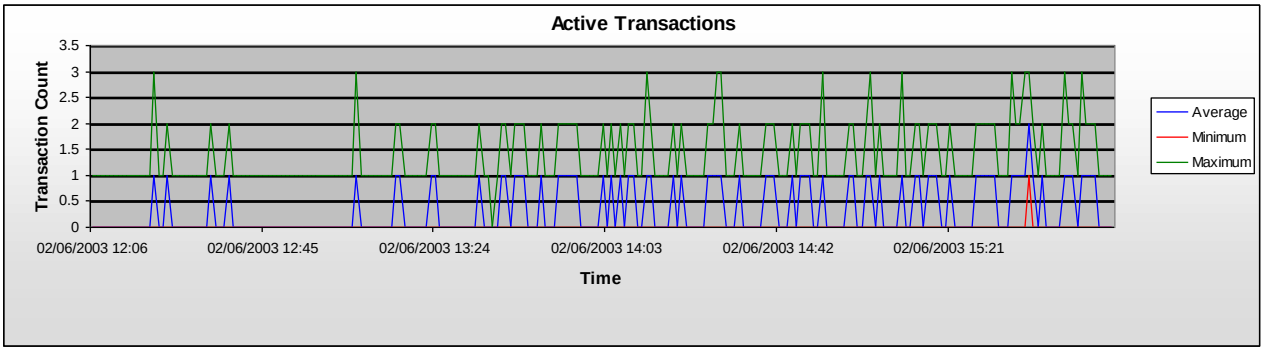


The charts will update to reflect that transaction, in this case Buy Stock. The charts are chronologically aligned across the same reporting period that was chosen on the Date tab, providing insight into relationships between the metrics.

Below you will see a subset of the charts for Buy Stock that are provided by AppMetrics.

Examples of Transaction Metrics

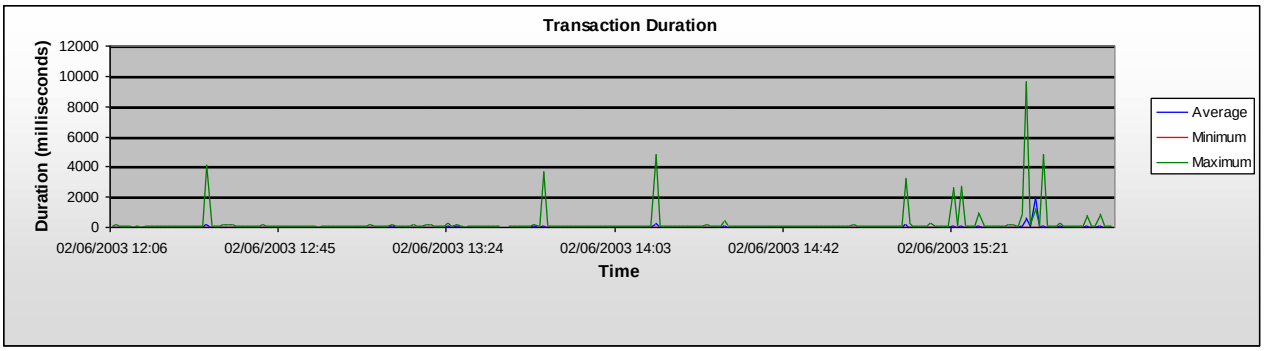
Active Components – Buy Stock



This report provides the number of transaction instances that AppMetrics observed over the reporting period. Activity is clearly going up over the time period.

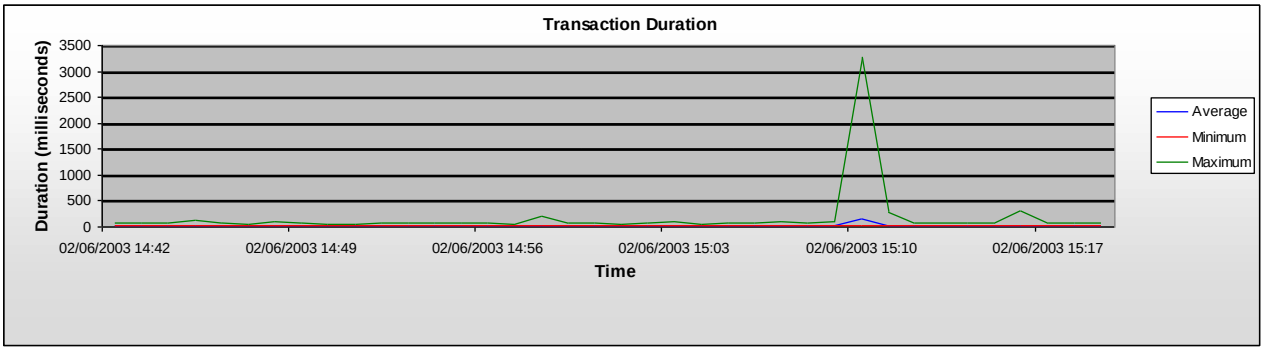
Knowing the number of Active Instances is important for capacity planning purposes, whether you want to observe trends over time, or want to identify particular times when demand is different from what's typical.

Transaction Duration – Buy Stock



This chart shows a great deal of variability in the duration of transaction instances. Comparing it to the above Active Instances, there is little correlation between the instance count and the duration, except perhaps on the right hand side of the chart. This might indicate that there is occasional database locking going on, or perhaps some blocking calls on shared threads.

Transaction Duration – Buy Stock



This chart is otherwise the same as the prior one, with this exception: here AppMetrics has zeroed-in on a period between 14:42 and 15:21 showing most transactions to be quite short-lived.

For a comparison of all the transactions, the following table is provided:

Transaction Comparison

	Active			Total (selected period)			Duration (ms)			Rate (per second)		
Transactions	**Avg**	**Min**	**Max**	**Started**	**Completed**	**Aborted**	**Avg**	**Min**	**Max**	**Started**	**Completed**	**Aborted**
Buy Stock	0.321	0	3	9,377	9,375	2	51.656	17	9,624	0.667	0.667	0
Chart Portfolio	0.030	0	2	4,734	4,734	0	13.816	7	4,209	0.336	0.336	0
Get Summary	0.017	0	2	4,717	4,717	0	14.159	7	4,684	0.335	0.335	0
List Positions	0.030	0	2	4,738	4,738	0	13.116	7	4,392	0.337	0.337	0
List Positions for Sale	0.060	0	3	9,463	9,463	0	11.408	6	4,958	0.673	0.673	0
Sell Stock	0.338	0	4	9,477	9,456	21	60.081	24	9,764	0.674	0.673	0.001
Verify Login	0.137	0	4	4,733	4,732	0	15.823	6	4,432	0.336	0.336	0
Verify Symbol	0.085	0	2	9,377	9,377	0	7.088	5	153	0.667	0.667	0

Some of these transactions run much longer than others. This may be as intended – but could be due to bottlenecks from database locks, from blocking calls on shared threads. Even when the transactions themselves are well behaved, long durations can be the "canary in the coal mine" that reveals a previously unseen problem: perhaps that transaction has a connection to an external resource which is in fact the source of the overall performance problem.

This chart highlights total transactions over the reporting period. If you know the business (dollar) value of these transactions, then knowing the number of transactions per day can be helpful in determining the ROI of a deployed applications –

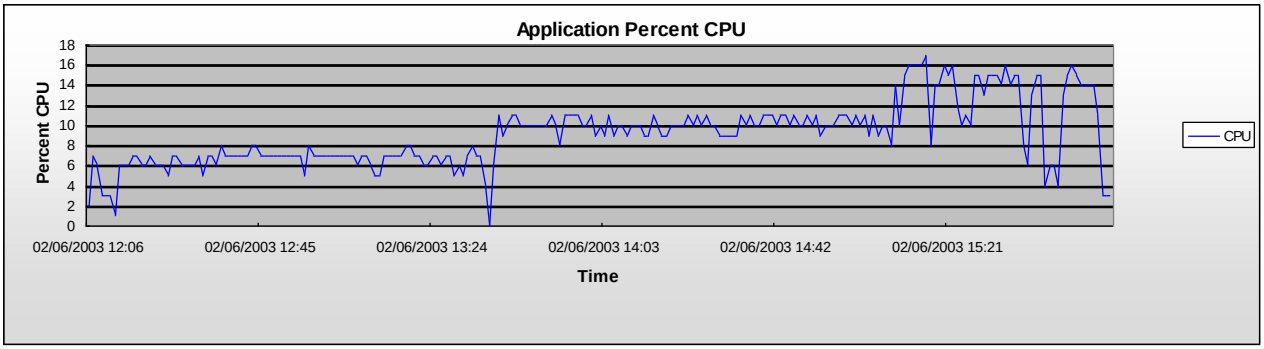
Knowing how many transactions of each type are aborting is important – if transactions are aborting, the business can be failing.

Application Resources

Let's review now the resources consumed by COM+ Applications. Click on the Applications tab, and you will see an Applications picker like this:

The next set of charts will all be for FMStocks 2000 Core.

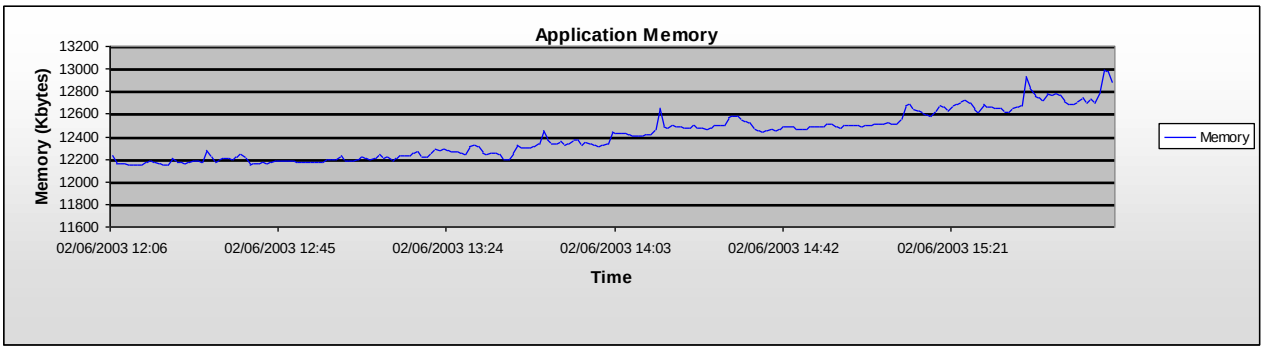
CPU – FMStocks 2000 Core



CPU consumption of FMStocks Core rises over the reporting period. Note that this is not %CPU for the entire machine, but the %CPU being consumed by FMStocks Core COM+ application process specifically.

In the case of.NET framework based applications, if process pooling is being utilized, AppMetrics collects metrics from each process instance and aggregates them into a single metric.

Application Memory – FMStocks 2000 Core



FMStocks Core can be seen using more and more memory. This could indicate a memory leak. Given our knowledge from the Component Instances chart (a growing number of instances) we understand that the memory consumption is being driven by growing application usage. When there is no application usage increase while Application Memory grows, a memory leak should be suspected.

Application Resource Comparison

	Percent CPU			Memory		Page Faults/Second			Application Threads		
Applications	**Avg**	**Min**	**Max**	**Min**	**Max**	**Avg**	**Min**	**Max**	**Avg**	**Min**	**Max**
FMStocks 2000 Core	9.103	0	17	12,144,298	12,992,512	58.833	0	116	27.154	26	29
IIS Out-Of-Process Pooled Applications	6.111	0	11	0	12,262,478	27.624	0	49	33.919	31	40
System Application	1.932	0	5	0	10,720,597	0	0	0	14.966	14	16

From this table you can compare and contrast the resource consumption of your Active COM+ applications.

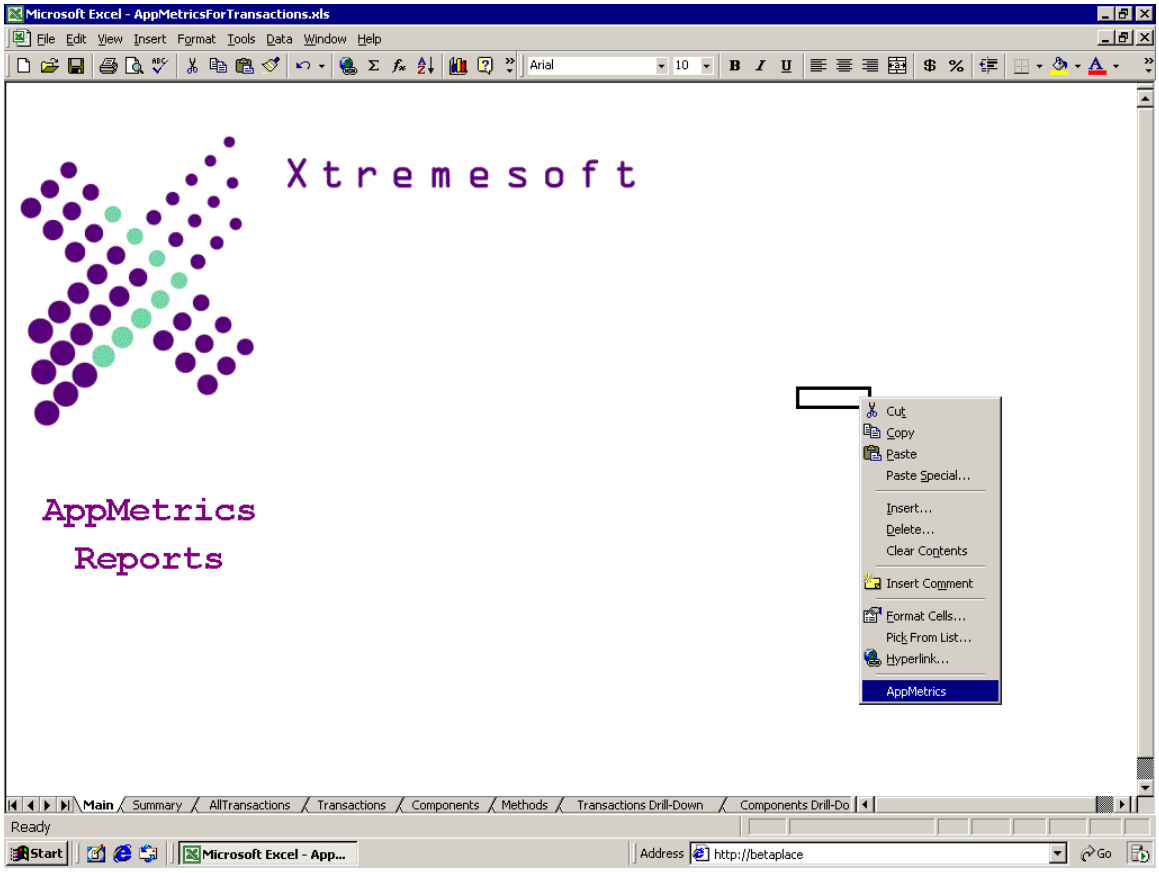
This table appears at the bottom of each Application Summary report. Using these values can be quite helpful in capacity planning exercises – whether you are anticipating future server capacity, or if you performing server consolidation, and wish to see which applications can best coexist on a given hardware configuration.

Exploring the Sample Diagnostics Reports

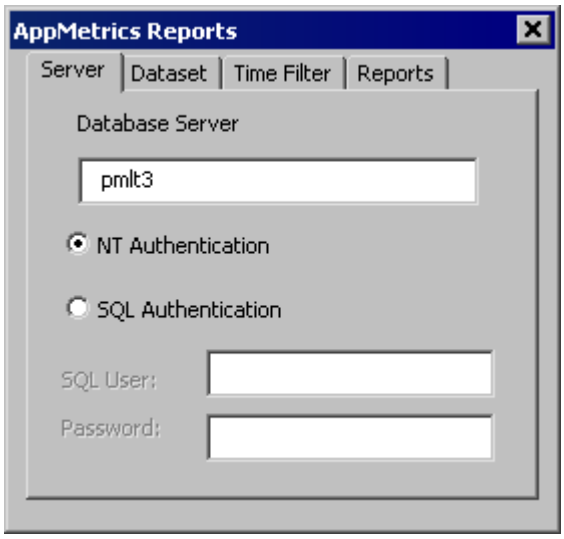
While the Production monitors sample the data and aggregate data over intervals (typically 60 seconds), the Diagnostics monitors record metrics as they happen. Many of the Diagnostic Reports have the same names as the Production Reports, but with much more granular detail.

These sample databases have a large dataset, but over a relatively short period of time. Note that the reports have several mechanisms for filtering the data, enabling you to control what you collect and store, and the impact that it has on the system

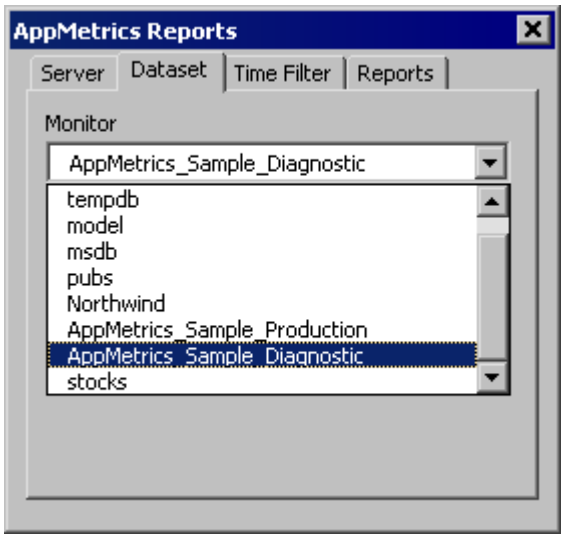
Right click in the spreadsheet, and select the AppMetrics menu at the bottom of the list.



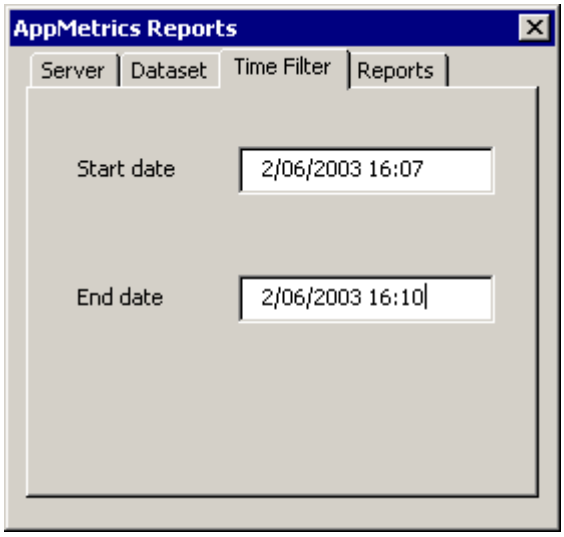
1. **Enter** the name of your **SQLServer** machine, and the appropriate credentials.



2. Click **Dataset** and select the **Sample Diagnostics** database.

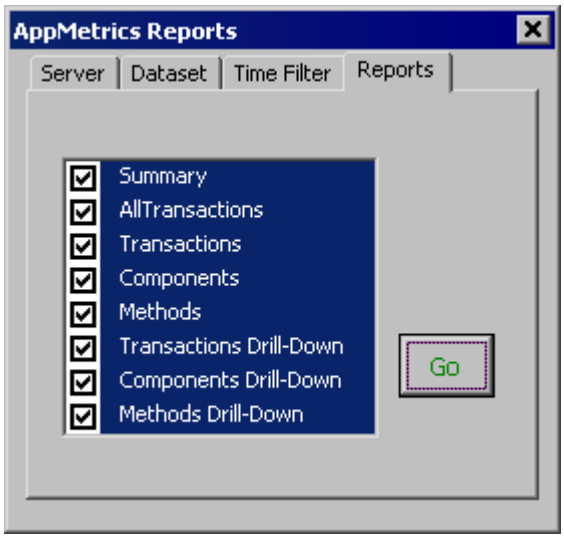


3. Click **Time Filter**. Accept the default **Start date** and **End date**. These correspond to the range of data contained within the database.



(Please note: the *only* data in the sample database corresponds to the date and time displayed here. Selecting other timeframes will result in a series of messages about the lack of data.)

4. Click **Reports** and click **Go**.



We will use the Methods and Method Drill-Down reports in this document; feel free to generate all the reports.

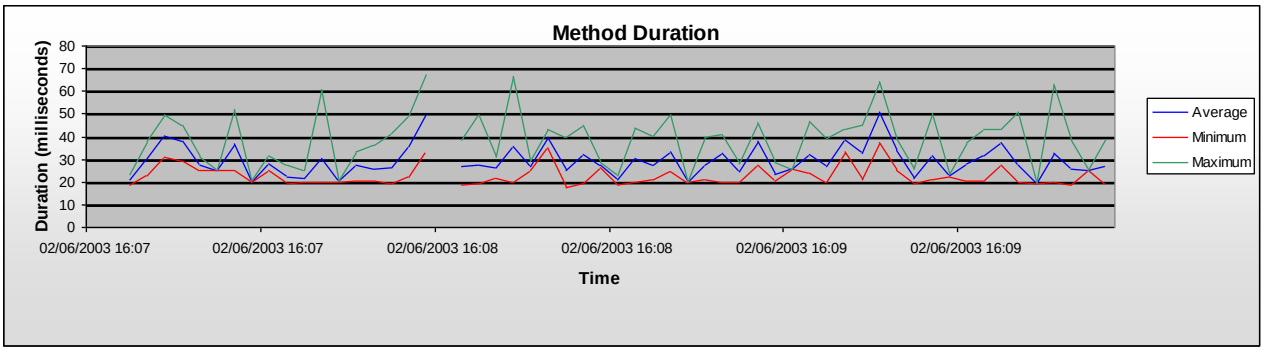
Example Method Reports

Generating all the reports at once, for this time period, will take a few minutes. Our first example will be the Methods report, but next please select the Excel sheet tab labeled "Methods".

Then use the Methods Picker to select FMStocks_Bus.Account->BuyStock, Refresh Charts to get the metrics for the charts.

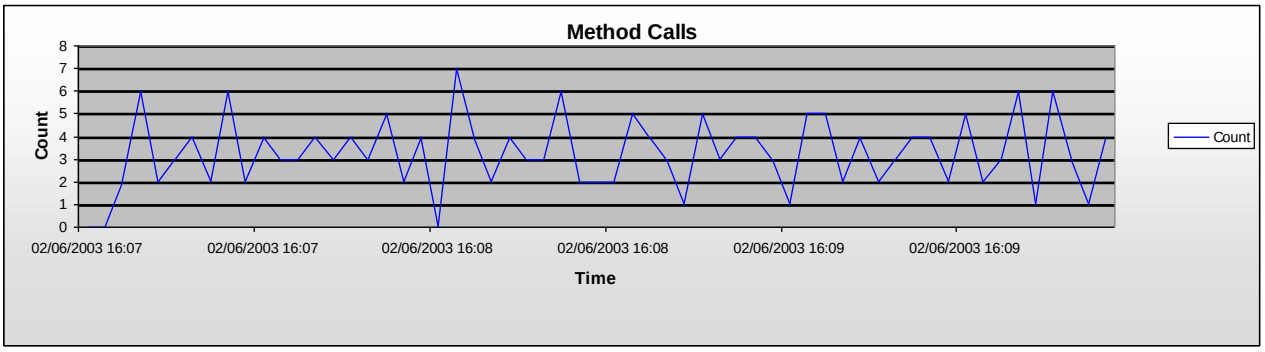
The method reports that follow are for BuyStock

Method Duration – FMStocks_Bus.Account->BuyStock



Note that BuyStock method durations in this reporting interval were typically close to 40 milliseconds. You'll see that at approximately 16:08 there is a hole in the chart; Look at the chart below and you will also see that there were no calls at that point in time.

Method Calls Count – FMStocks_Bus.Account->Buy Stock



Active BuyStock method calls over the reporting time period were around 2- 6 at a time.

Methods Comparison Report

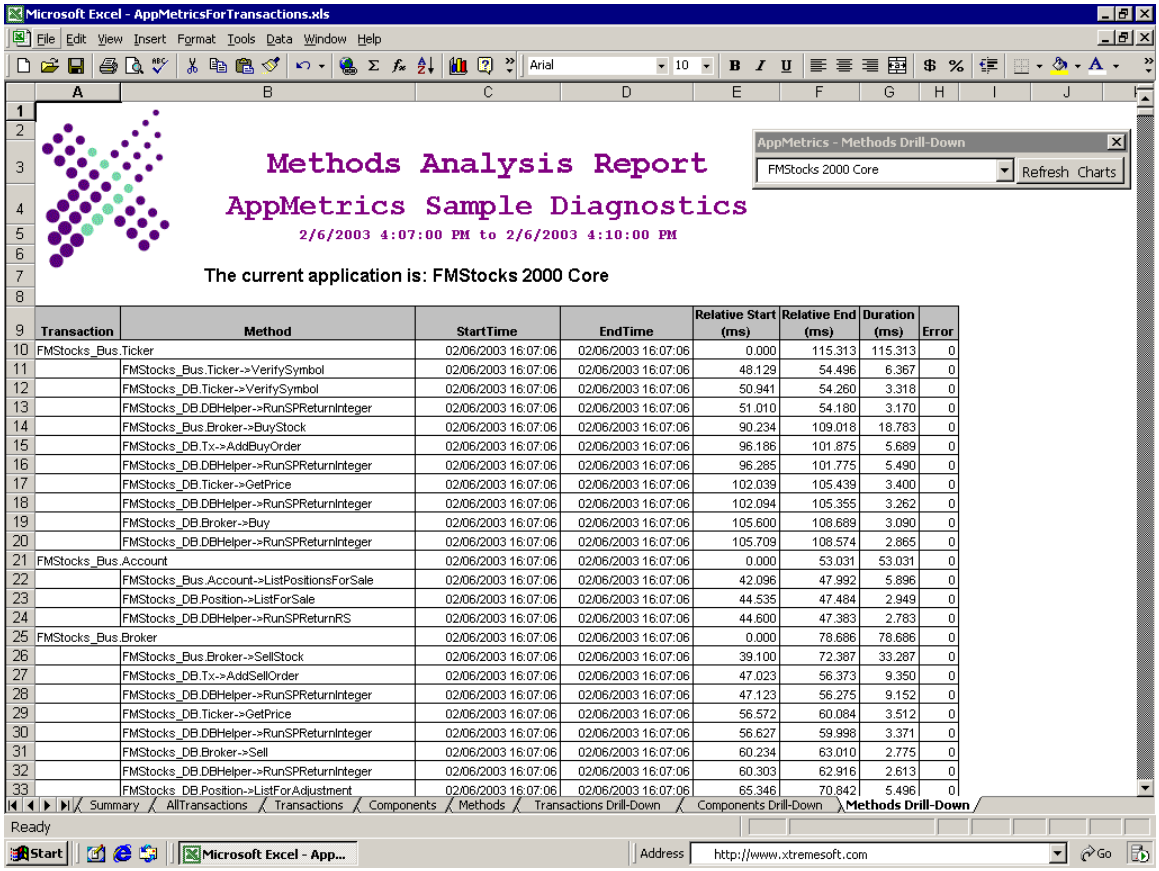
	Duration (ms)			Completed		
Name	Average	Minimum	Maximum	Total	Successful	Exceptions
FMStocks_Bus.Account->GetSummary	11.272	7.250	29.627	98	98	0
FMStocks_Bus.Account->ListPositions	10.180	7.680	26.059	193	193	0
FMStocks_Bus.Account->ListPositionsForSale	8.016	5.561	24.588	195	195	0
FMStocks_Bus.Account->VerifyLogin	8.769	6.365	16.969	99	99	0
FMStocks_Bus.Broker->BuyStock	29.617	17.734	67.777	196	196	0
FMStocks_Bus.Broker->SellStock	37.693	25.430	112.438	196	196	0
FMStocks_Bus.Ticker->VerifySymbol	6.895	4.566	22.070	196	196	0
FMStocks_DB.Account->Summary	5.457	3.693	18.260	98	98	0
FMStocks_DB.Account->VerifyUser	4.064	2.951	9.453	99	99	0
FMStocks_DB.Broker->Buy	4.154	3.064	18.180	196	196	0
FMStocks_DB.Broker->Sell	3.576	2.717	19.334	196	196	0
FMStocks_DB.DBHelper->GetConnectionString	1.004	0.746	2.363	99	99	0
FMStocks_DB.DBHelper->RunSPReturnInteger	5.444	2.096	74.691	1,372	1,372	0
FMStocks_DB.DBHelper->RunSPReturnRS	4.913	2.740	23.156	486	486	0
FMStocks_DB.DBHelper->RunSPReturnRS_RW	5.790	3.207	32.939	196	196	0
FMStocks_DB.Position->ListForAdjustment	6.017	3.398	33.137	196	196	0
FMStocks_DB.Position->ListForSale	4.170	2.904	15.678	195	195	0
FMStocks_DB.Position->ListSummary	6.540	4.980	23.328	193	193	0
FMStocks_DB.Ticker->GetPrice	5.442	2.420	24.004	392	392	0
FMStocks_DB.Ticker->VerifySymbol	3.214	2.238	11.797	196	196	0
FMStocks_DB.Tx->AddBuyOrder	9.401	3.578	35.422	196	196	0
FMStocks_DB.Tx->AddSellOrder	8.271	3.621	74.934	196	196	0

Here you can compare and contrast the method durations and volumes over the selected reporting period.

Methods that are taking much longer than the others may very well indicate a bottleneck in your system. Methods that are called very often (higher numbers under ‘Completed’) may be good candidates for spending optimization resources.

Method Analysis Report

When you select the Methods Drill-Down tab, you will see the Methods Analysis Report. The purpose of the report is to enable you to see the sequence of method calls involved in each transaction that AppMetrics has discovered in the application. Notice the start and end time for each method, as well as the duration. Zoom in to see the details better.



Microsoft Excel - AppMetricsForTransactions.xls

AppMetrics - Methods Drill-Down

FMStocks 2000 Core | Refresh Charts

Methods Analysis Report

AppMetrics Sample Diagnostics

2/6/2003 4:07:00 PM to 2/6/2003 4:10:00 PM

The current application is: FMStocks 2000 Core

Transaction	Method	StartTime	EndTime	Relative Start (ms)	Relative End (ms)	Duration (ms)	Error
FMStocks_Bus.Ticker		02/06/2003 16:07:06	02/06/2003 16:07:06	0.000	115.313	115.313	0
	FMStocks_Bus.Ticker->VerifySymbol	02/06/2003 16:07:06	02/06/2003 16:07:06	48.129	54.496	6.367	0
	FMStocks_DB.Ticker->VerifySymbol	02/06/2003 16:07:06	02/06/2003 16:07:06	50.941	54.260	3.318	0
	FMStocks_DB.DBHelper->RunSPReturnInteger	02/06/2003 16:07:06	02/06/2003 16:07:06	51.010	54.180	3.170	0
	FMStocks_Bus.Broker->BuyStock	02/06/2003 16:07:06	02/06/2003 16:07:06	90.234	109.018	18.783	0
	FMStocks_DB.Tx->AddBuyOrder	02/06/2003 16:07:06	02/06/2003 16:07:06	96.186	101.875	5.689	0
	FMStocks_DB.DBHelper->RunSPReturnInteger	02/06/2003 16:07:06	02/06/2003 16:07:06	96.285	101.775	5.490	0
	FMStocks_DB.Ticker->GetPrice	02/06/2003 16:07:06	02/06/2003 16:07:06	102.039	105.439	3.400	0
	FMStocks_DB.DBHelper->RunSPReturnInteger	02/06/2003 16:07:06	02/06/2003 16:07:06	102.094	105.355	3.262	0
	FMStocks_DB.Broker->Buy	02/06/2003 16:07:06	02/06/2003 16:07:06	105.600	108.689	3.090	0
	FMStocks_DB.DBHelper->RunSPReturnInteger	02/06/2003 16:07:06	02/06/2003 16:07:06	105.709	108.574	2.865	0
FMStocks_Bus.Account		02/06/2003 16:07:06	02/06/2003 16:07:06	0.000	53.031	53.031	0
	FMStocks_Bus.Account->ListPositionsForSale	02/06/2003 16:07:06	02/06/2003 16:07:06	42.096	47.992	5.896	0
	FMStocks_DB.Position->ListForSale	02/06/2003 16:07:06	02/06/2003 16:07:06	44.535	47.484	2.949	0
	FMStocks_DB.DBHelper->RunSPReturnRS	02/06/2003 16:07:06	02/06/2003 16:07:06	44.600	47.383	2.783	0
FMStocks_Bus.Broker		02/06/2003 16:07:06	02/06/2003 16:07:06	0.000	78.686	78.686	0
	FMStocks_Bus.Broker->SellStock	02/06/2003 16:07:06	02/06/2003 16:07:06	39.100	72.387	33.287	0
	FMStocks_DB.Tx->AddSellOrder	02/06/2003 16:07:06	02/06/2003 16:07:06	47.023	56.373	9.350	0
	FMStocks_DB.DBHelper->RunSPReturnInteger	02/06/2003 16:07:06	02/06/2003 16:07:06	47.123	56.275	9.152	0
	FMStocks_DB.Ticker->GetPrice	02/06/2003 16:07:06	02/06/2003 16:07:06	56.572	60.084	3.512	0
	FMStocks_DB.DBHelper->RunSPReturnInteger	02/06/2003 16:07:06	02/06/2003 16:07:06	56.627	59.998	3.371	0
	FMStocks_DB.Broker->Sell	02/06/2003 16:07:06	02/06/2003 16:07:06	60.234	63.010	2.775	0
	FMStocks_DB.DBHelper->RunSPReturnInteger	02/06/2003 16:07:06	02/06/2003 16:07:06	60.303	62.916	2.613	0
	FMStocks_DB.Position->ListForAdjustment	02/06/2003 16:07:06	02/06/2003 16:07:06	65.346	70.842	5.496	0

Summary | AllTransactions | Transactions | Components | Methods | Transactions Drill-Down | Components Drill-Down | Methods Drill-Down

Ready

Start | Microsoft Excel - App... | Address http://www.xtremesoft.com | Go

This report provides you with a trace of the methods that were called during the selected time period. The "Transaction" in this report is a root object typically created by an Active Server Page, or other caller; the methods are all subordinate to this root object instance.

Next Steps

We hope this document and the accompanying sample reports have helped you gain a better understanding of how AppMetrics can add value to your day-to-day operations. By seeing the inner workings of your COM+ and .NET Serviced Component applications you'll be better prepared to diagnose problems and prevent them before they even occur.

Please e-mail any questions you might have to Xtremesoft Customer Support

Imagine seeing your actual application metrics in these reports. Please contact Xtremesoft Sales to take the next step.

Contact Information

Sales: salesgroup@xtremesoft.com
Customer Support: supportlist@xtremesoft.com

Address:
1020 Winter Street
Suite 1000
Waltham, MA 02451 USA
+1 781-759-1220 phone
+1 781-530-3605 fax