Consulting
Services

Performance Testing With JmsStream

This document describes how use JmsStream and MS Excel to determine the message throughput performance of a process or system. This type of performance testing only works with processes that accept a JMS message as the process starter (initiator), and produce a JMS message as a final output. Also, the process must pass a timestamp or correlation ID from the starter message to the final message.

This document assumes the reader is familiar with JMS and MS Excel.

TIBCO®
The Power of Now®

http://www.tibco.com

Global Headquarters
3303 Hillview Avenue
Palo Alto, CA 94304
Tel: +1 650-846-1000
Toll Free: 1 800-420-8450
Fax: +1 650-846-1005

Table of Contents

1 Introduction

1.1 Message Rate Performance

The purpose of the performance test is to determine the maximum sustained message processing rate of a process or system. This is done by calculating the process latency of the system. When determining the maximum throughput of a system a constant latency is irrelevant. The maximum performance is determined when the latency of the processes starts to increase. The increase in latency is easily seen by graphing the processing time superimposed on the message rate.

1.2 Microsoft Excel

Microsoft Excel is used to import the coma separated value (CSV) file produced by JmsStream, calculate the latency of the test system, and display the performance chart.

2 How to Gather the Data

2.1 Capture the Input Messages using JmsStream

In order to start the test, one must capture or create messages to start the process. It is recommended that the input message is captured, because it can be difficult to accurately create an input message. When capturing JMS Text Type Message it is better to save it as a text file type, because it is easier to modify and has better publishing performance. Also, JmsStream has the capability to loop over the publishing file in order to repeat the message publications. This will allow the ability to only capture a few messages, but have long test durations. The command below is an example of capturing a text type message:

```
C:\TIBCO\ems\bin>java –jar JmsStream.jar -listen queue.in -queue -providerurl tibjmsnaming://svr-esb:7222 -user admin -password mypass –file c:\jmss_in.sav
```

The captured message in this case is a text type XML message, which is the most common message type. An example of the captured message is give below:

```
#---------- #1 : ID:EMS-SERVER.1044445FBC273E:2 ----------#

<MSG_INFO>
    <message type="TextMessage" messageSelector="" receiveTime="2006-05-09T03:49:22.251"
jmsServerTimestamp="2006-05-09T03:49:22.241">
        <header JMSMessageID="ID:EMS-SERVER.1044445FBC273E:2" JMSDestination="queue.in"
JMSReplyTo="queue.out" JMSDeliveryMode="2" JMSPriority="4" JMSTimestamp="1147139362241"/>
        <properties>
            <property name="ae_tracking" type="String">NGpb1mam4VSeqERMwZ-EOUEkges</property>
            <property name="ae_pfmt" type="Short">10</property>
            <property name="ae_tracking_info_1" type="String">IJMZ2O0g/RYuIkFunH-EObDEges</property>
            <property name="ae_type" type="Short">2</property>
            <property name="ae_ver" type="Short">50</property>
            <property name="ae_encoding" type="Short">2</property>
        </properties>
    </message>
</MSG_INFO>
BodyLength=1323
<?xml version="1.0" encoding="UTF-8"?>
```

```
<root xmlns:ns0="http://www.tibco.com/SCHEMA/ARCH/ENVELOPE">
    <ns0:MsgEnvelope xmlns:ns0="http://www.tibco.com/SCHEMA/ARCH/ENVELOPE" creationDateTime="2005-
11-04T18:09:10.647-05:00" sourceApplication="ORCL" projectName="SCN_PurchaseOrders_ORCL_P01"
processId="450000" messageGuid="LEdAMki9/RYwRkZZEXNds9CEmAg"/>
    <PurchaseOrderType>
        <ns0:PurchaseOrder xmlns:ns0="http://www.tibco.com/CDM/FIN/PO">
            <ns0:POHeader PO_HEADER_ID="2763" PURCHASE_ORDER_NUMBER="200063" PASI_PO_NUMBER=""
PO_TYPE="2" CURRENCY_CODE="USD" VENDOR_NAME="TIBCO COMPANY, INC" VENDOR_CODE="TBX" VENDOR_ID="61"
VENDOR_SITE_CODE="TBX" VENDOR_SITE_ID="62" SHIP_TO_LOCATION_ID="141" BILL_TO_LOCATION_ID="141"
PAYMENT_TERMS="20TH DAY OF 4TH MONTH" TERMS_ID="10009" OEM_FLAG="NO" APPROVAL_STATUS="APPROVED"
RIVISION_NUM="0" PO_ACTION="2" ENABLED_FLAG="" MERCHANT_CODE="" REQUESTED_SHIPMENT_DAYS="20041031"
TRANSPORTATION_MODE="OCEAN" PRODUCTION_SHIPMENT_MONTH="200410">
                <ns0:POLines PO_LINE_ID="31723" PO_HEADER_ID="2763" PURCHASE_ORDER_NUMBER="200063"
LINE_NUM="1" ITEM="410619" ITEM_ID="206" ITEM_DESCRIPTION="TIBCO AFICIO 1035" UNIT_PRICE="2496.0"
MEASUREMENT_OF_UNIT="EA" QUANTITY="10" APPROVAL_STATUS="">
                </ns0:POLines>
            </ns0:POHeader>
        </ns0:PurchaseOrder>
    </PurchaseOrderType>
</root>
```

The line of the message is the identifier and indicates the start of the message. It is made up of an internal message number and the JMS message ID. Neither of these values are used during playback, but the line must exist. The next line beginning with `<MSG_INFO>`, starts the JmsStream message header information. The element `<message type="TextMessage"/>` is required, but all other header information is optional. The line starting with `BodyLength=` is very important. It indicates the length of the text message body that begins on the next line. If you change any information in the body you must change the `BodyLength` value to reflect the change. A new message starts on a new line following the message body.

If there is any doubt of the message body length, use `-getbodylength` argument to ascertain the body length. To use, create a file that contains only the message body. Then execute `java -jar JmsStream.jar -getbodylength <file-name>`. With `<file-name>` being the name of the file that contains the message body. The body length will be displayed on the screen. Copy this body length into message `BodyLenght=` in the replay (send) file.

2.2 Playing the Input Messages and Capture the Output Messages

2.2.1 Starting a Test

JmsStream can add three properties to the JMS messages when it is published. The fist property is the sending timestamp, and it must be passed from the starter message (published from JmsStream) to the final output message (published to JmsStream). The second is the messages sequence number and the third is the sending message rate. Ideally all three properties are passed to the final output message of the system, but the process or system to be tested must pass the timestamp for the performance test to work. This is typically done by having JmsStream put a publishing timestamp in the JMSType, JMSCorrelationID, or custom property of the starter message and then mapping this to the JMSType, JMSCorrelationID, or custom property of the final outgoing message. JMSType is convenient, because it is a string property in the JMS header and it is not use by any known JMS implementation. JMSCorrelationID property can also be used.

After the process or system is started, start the JmsStream listener. Example:

```
C:\TIBCO\ems\bin>java -jar jmsstream.jar -listen queue.out -queue -providerurl tibjmsnaming://svr-
esb:7222 –user admin -password admin –csvfile c:/perftes_db_out.csv -stats 4000
```

In this example JmsStream is listening to a queue named `queue.out` and writing a CSV output file. It is also reporting JMS connection statistics on the console every 4 seconds. The CSV file will be described in section 3. The stats output looks like this:

```
2006-04-10T10:59:22.526: New messages received (within 4000ms): 22
*** Average message rate per second: 5 *** total count at: 63
```

Form this output one can tell the average message rate per second through the connection. This is a rough estimate of the output performance, but it is not necessarily reliable. For example, if a process is only capable of producing 5 msg/sec and there is an input of 50 msg/sec, the output rate will be considerably less that 5 msg/sec due to system backup. However if the input message rate is 6 msg/sec the output rate in the stats display will be fairly accurate.

Playing the messages from JmsStream is relatively easy. If using the JMSXSndTimestamp property to put a timestamp, make you put the `–sndtimestamp JMSXSndTimestamp` argument in JmsStream command. The same applies for the `–sequence` and `–ratestamp` arguments. Example:

```
C:\TIBCO\ems\bin>java -jar jmsstream.jar -send c:/perftest_in.sav -queue -providerurl
tibjmsnaming://svr-esb:7222 -user admin -password admin -sndtimestamp JMSXSndTimestamp -sequence
JMSXGroupSeq -ratestamp JMSXMsgRate -startrate 1 -maxrate 20 -numberofintervals 20 -intervalsize 100
–noecho
```

This command will connect to a queue (the name of the queue is in the file), read the messages form `c:/perftest_in.sav` file, put a timestamp in the JMSXSndTimestamp property, put the message sequence number in the JMSXGroupSeq property, put the JmsStream publishing rate in the JMSXMsgRate property, start publishing at a rate of 1 msg/sec, and increase by 1 msg/sec every 100 messages until the rate reaches 20 msg/sec. There will be a total of 2000 messages published in this example. If there are not enough messages in the file the messages are repeated.

Note: Make sure there are enough published messages to get accurate data. Typically the interval size is three to five times the max rate.

3 How to Analyze the Data

3.1 JmsStream CSV File

The performance data is in the CSV files created by the JmsStream listener. The first line in the CSV file is the field names, and subsequent lines are the actual data. An example of a CSV file is given below:

```
JMSMessageID,JMSDestination,JMSReplyTo,JMSCorrelationID,JMSDeliveryMode,JMSPriority,JMSType,JMSExpir
ation,JMSTimestamp,UserProperties--->
ID:EMS-LAPTOP3.F5F445F0C4891:307,queue.out,,,2,4,,,2006-05-10T13:20:27.775,2006-05-
10T13:20:15.822,1,1
ID:EMS-LAPTOP3.F5F445F0C4894:308,queue.out,,,2,4,,,2006-05-10T13:20:28.787,2006-05-
10T13:20:16.824,1,1
ID:EMS-LAPTOP3.F5F445F0C4893:309,queue.out,,,2,4,,,2006-05-10T13:20:29.788,2006-05-
10T13:20:17.825,1,1
```

In this example the original JmsStream send timestamp was mapped by the test process to the JMSXSndTimestamp property of the output message. This is the first of the user property fields. The JMSTimestamp is the time when the output message arrived at the JMS server. You can determine the system latency by subtracting the JMSXSndTimestamp from the JMSTimestamp. The JMSXGroupSeq is in the second user property field, and JMSXMsgRate is in the third user property field.

3.2 Importing CSV File into an Excel Workbook

3.2.1 Importing the Data

Open a new Microsoft Excel Workbook. Next, select open file on the Excel menu bar. In the Open dialogue box select the Text Files type and then select the JmsStream CSV files. See *Figure 1*

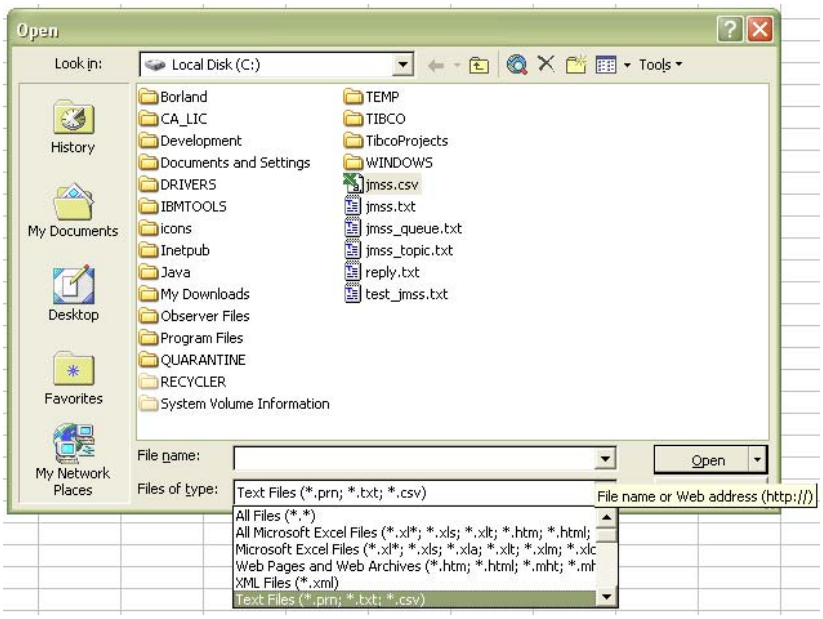


Figure 1

Copy the data from the CSV file generated by the first test into the new Excel Workbook. Highlight the spreadsheet fields and select AutoFit Selection. See *Figure 2*.

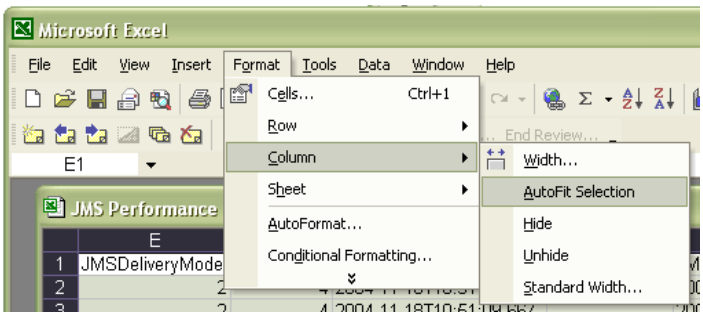


Figure 2

Next, label the JMSXGroupSeq and JMSXMsgRate fields. The fields are labeled “Sequence” and “Msg/Sec” in column K and L in *Figure 3*. Column J is the JMSXSndTimestamp property, but does not have to be labeled.

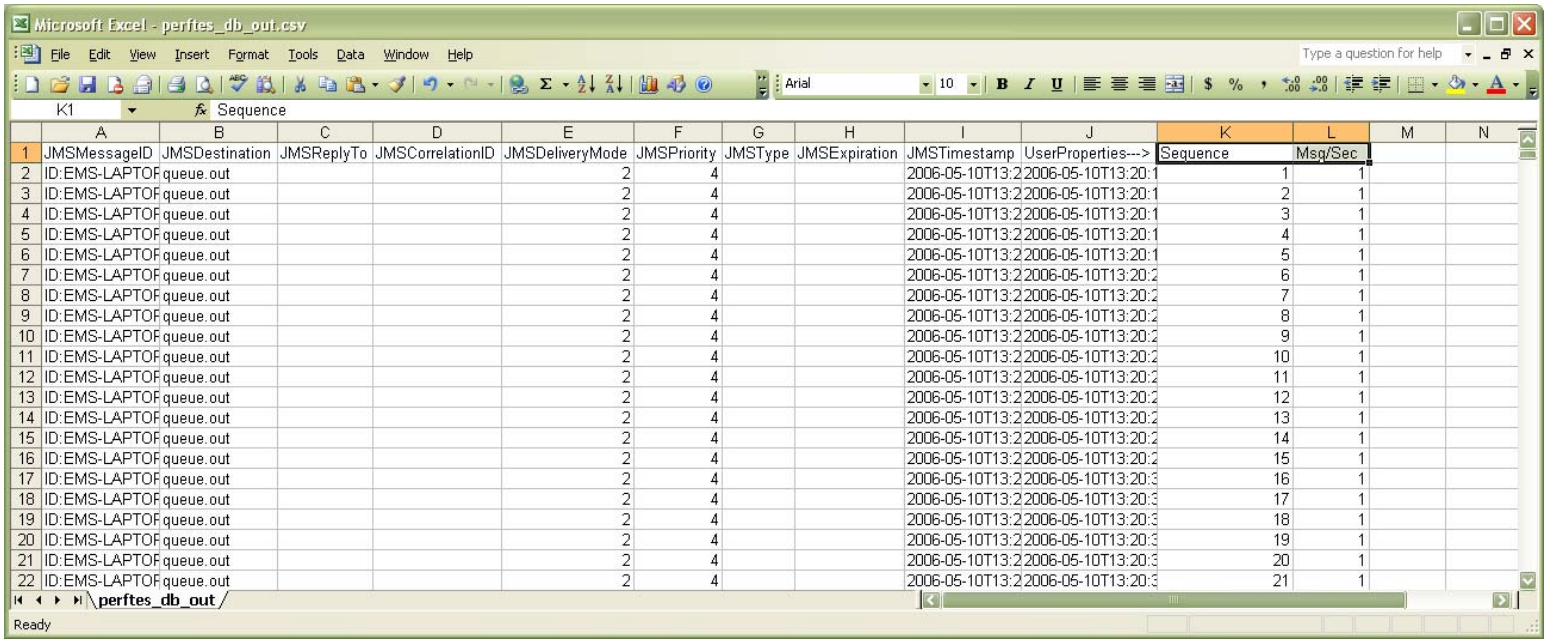


Figure 3

To create the graph base on the sequence of publication, sort the messages on the Sequence column. See *Figure 4* for an example. Sorting by sequence number is not necessary and in only useful to identify abnormally long delay for a particular message. To sort click the column containing the message sequence and then click the “Sort Ascending” button. When the message box appears, click “Sort”. The rows are now sorted

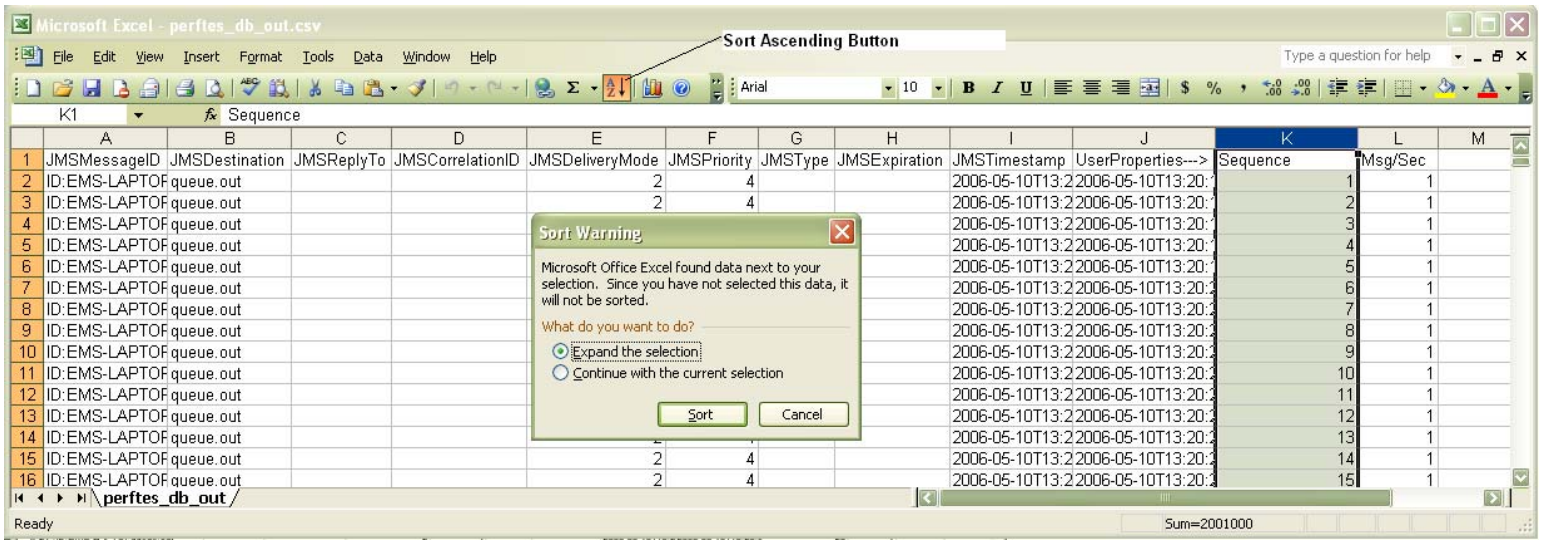


Figure 4

3.2.2 Calculating Latency

In a blank cell (Column M) on the heading line after the user properties type a label named “Latency”. See *Figure 5*.

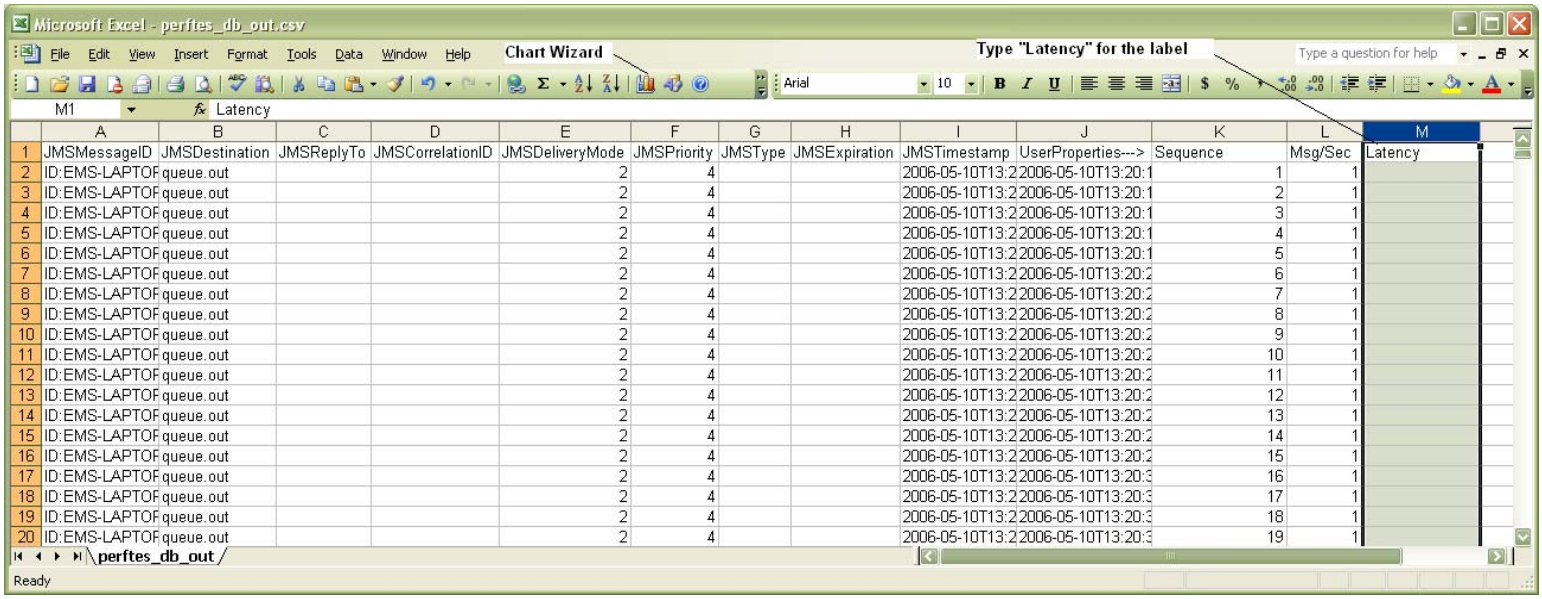


Figure 5

Next, calculate the process latency by subtracting column J (JMSXSndTimestamp property) from column I (JMSTimestamp propery). In the “Latency column (Column M) on row 2 type the formula `=TIMEVALUE(MID(I2,12,12))-TIMEVALUE(MID(J2,12,12))`. Copy this formula to all the other rows. This formula will calculate the latency of the process in Microsoft time format. This is NOT seconds. For the purposes of a performance test, the time units are irrelevant and converting the value to milliseconds is unnecessary.

3.3 Creating a Chart

3.3.1 Graphing Raw Values

Highlight all of the data in Column L and Column M and select the Excel Chart Wizard button on the task bar (Shown in *Figure 5*). Next, click Custom Type tab and select Lines on 2 Axis. See *Figure 6*.

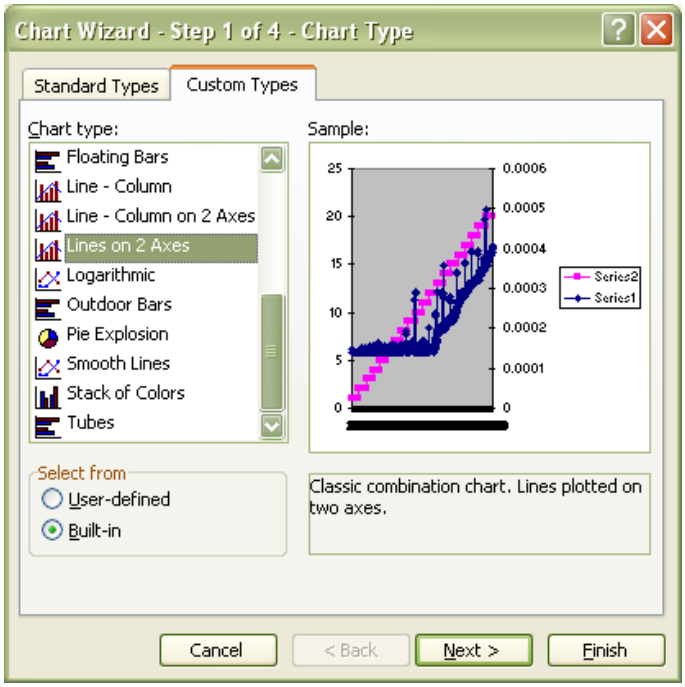


Figure 6

Keep clicking Next, and at the last step select Place chart as new sheet, and except the defaults. Click Finish. It is best to create the chart as a new sheet. Column I2 is the JMSTimestamp property and Column J2 is the JMSXSndTimestamp property (the send timestamp of the input message).

In the JMS Performance `perftes_db_out.csv` example a message rate from 1 msg/sec to 20 msg/sec was sent to a two thread BW engine with a stored procedure call. From the chart in *Figure 7*, we can determine that the BW engine cannot handle the 13 msg/sec rate because the line starts to have a positive slope. The spikes in the message latency are cause when certain messages take longer to process than others. The spikes should be disregarded, and the observed slope should the slope of a moving average trend-line for the latency.

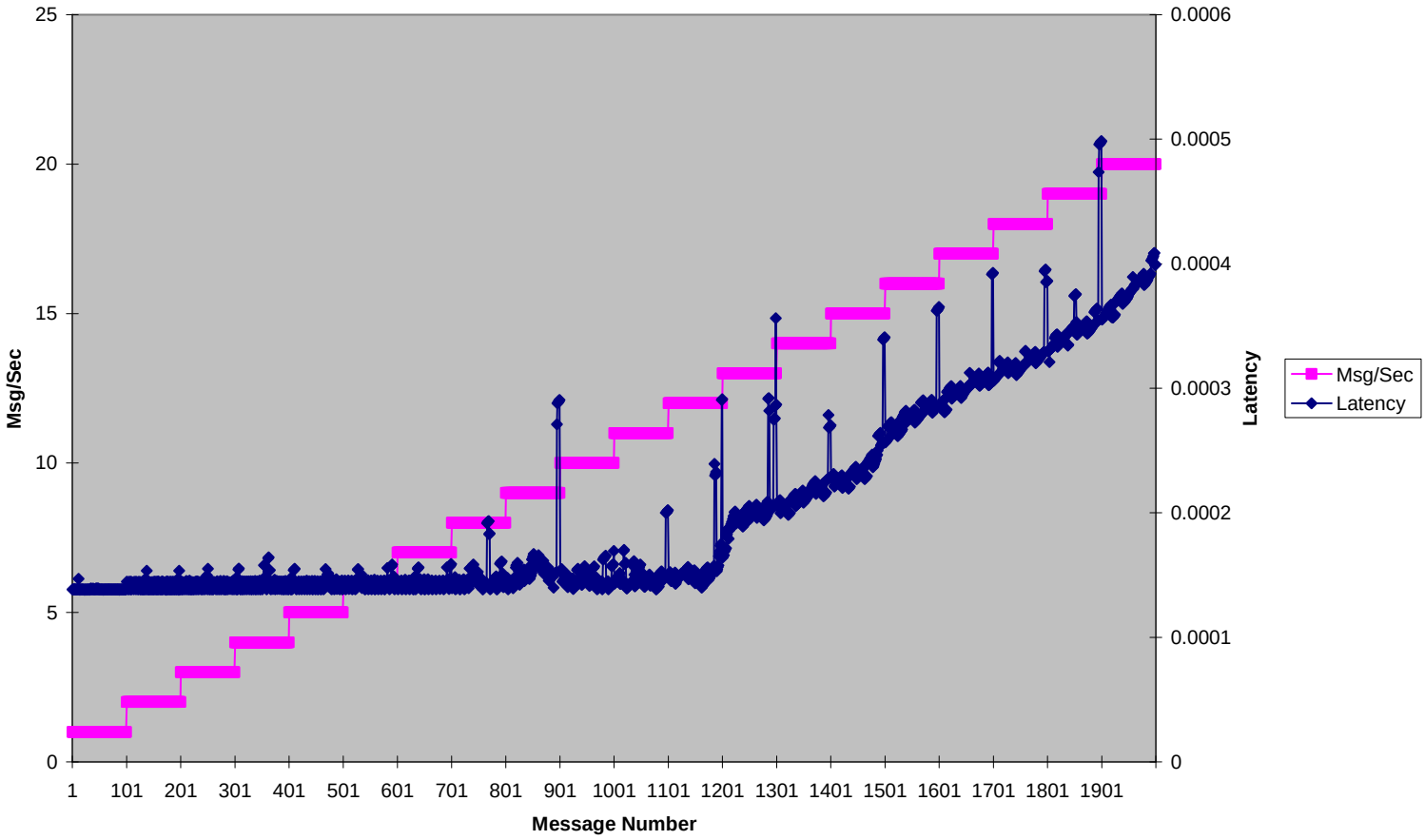


Figure 7

In *Figure 8* a 100 point moving average trend-line was added by right clicking on the Latency series and selecting Add Trendline. It is shown in yellow. Also, the chart type was changed to an XY (Scatter) by right clicking on the data series and selecting Chart Type.

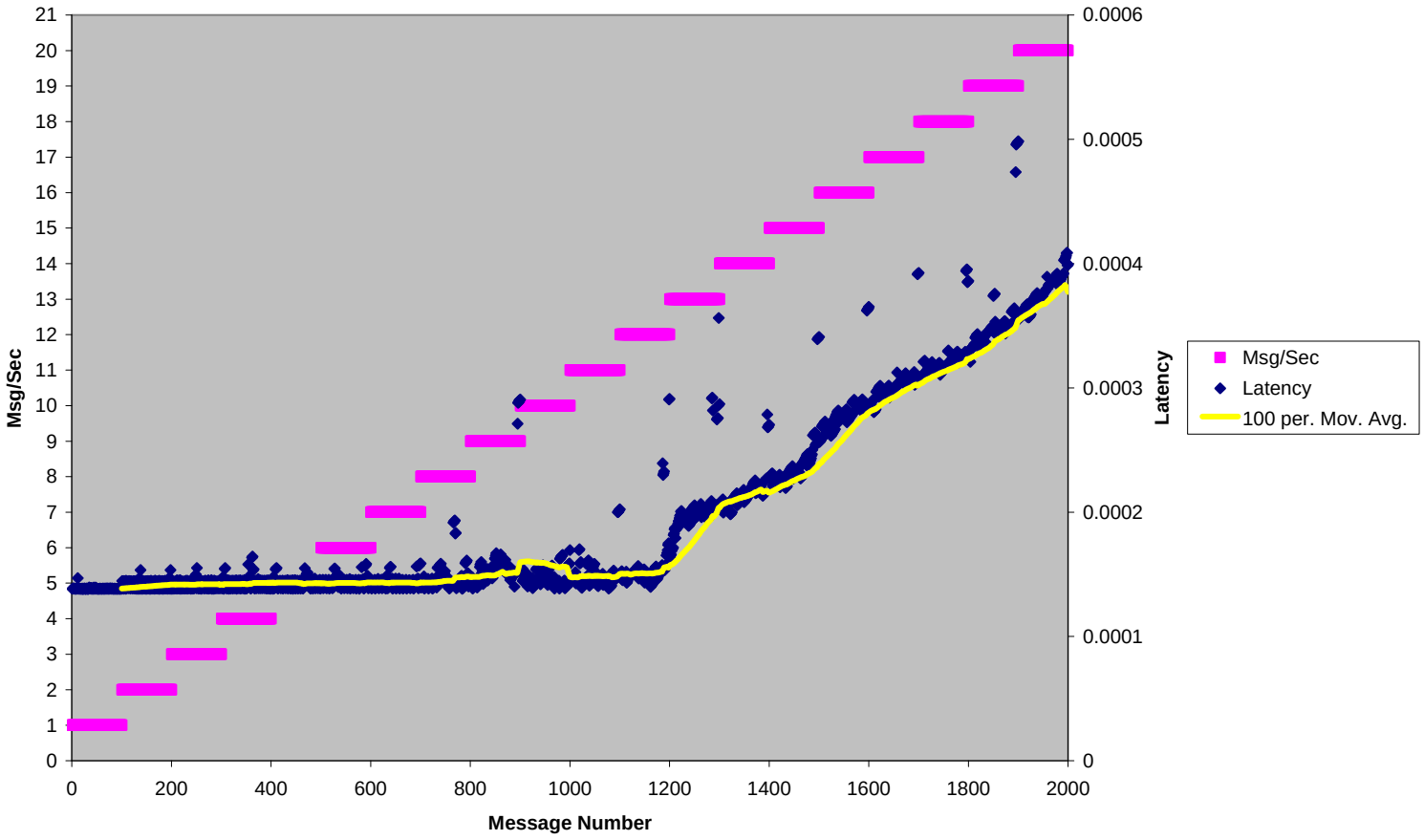
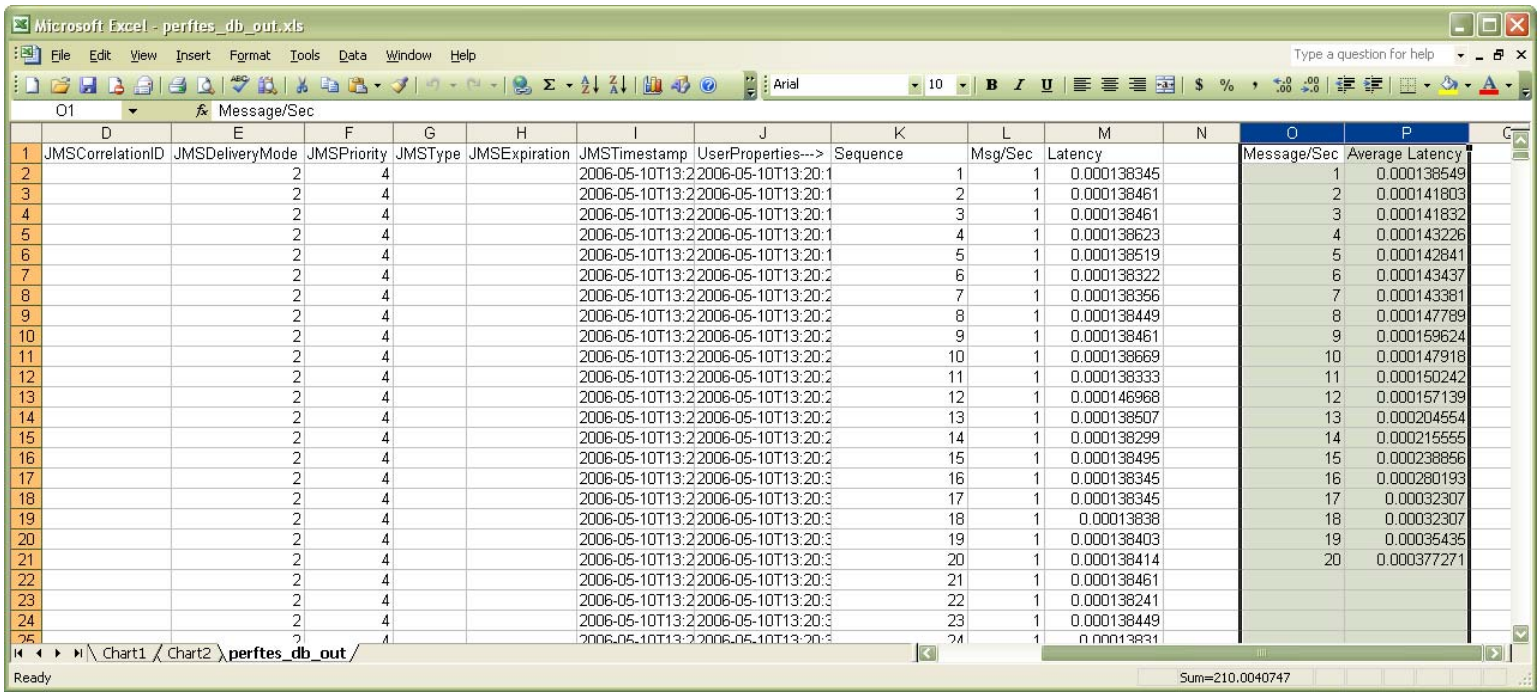


Figure 8

3.3.2 Graphing Average Values

An average value graph is easier to read and not as cluttered. To create an average value graph, first type "Message/Sec" and "Average Latency" headers in the spreadsheet. Next, under "Messages/Sec" type each rate interval, and under "Average Latency" type the formula `=AVERAGE(M2:M101)`. Calculate the average for each rate interval. For instance the first rate interval is 1 Msg/Sec and the average is `=AVERAGE(M2:M101)`, and the next rate interval is 2 Msg/Sec and the average is `=AVERAGE(M102:M201)`. See example in *Figure 9*.



	D	E	F	G	H	I	J	K	L	M	N	O	P
1	JMSCorrelationID	JMSDeliveryMode	JMSPriority	JMSType	JMSExpiration	JMSTimestamp	UserProperties--->	Sequence	Msg/Sec	Latency		Message/Sec	Average Latency
2		2	4			2006-05-10T13:2	2006-05-10T13:20:1	1	1	0.000138345		1	0.000138549
3		2	4			2006-05-10T13:2	2006-05-10T13:20:1	2	1	0.000138461		2	0.000141803
4		2	4			2006-05-10T13:2	2006-05-10T13:20:1	3	1	0.000138461		3	0.000141832
5		2	4			2006-05-10T13:2	2006-05-10T13:20:1	4	1	0.000138623		4	0.000143226
6		2	4			2006-05-10T13:2	2006-05-10T13:20:1	5	1	0.000138519		5	0.000142841
7		2	4			2006-05-10T13:2	2006-05-10T13:20:2	6	1	0.000138322		6	0.000143437
8		2	4			2006-05-10T13:2	2006-05-10T13:20:2	7	1	0.000138356		7	0.000143381
9		2	4			2006-05-10T13:2	2006-05-10T13:20:2	8	1	0.000138449		8	0.000147789
10		2	4			2006-05-10T13:2	2006-05-10T13:20:2	9	1	0.000138461		9	0.000159624
11		2	4			2006-05-10T13:2	2006-05-10T13:20:2	10	1	0.000138669		10	0.000147918
12		2	4			2006-05-10T13:2	2006-05-10T13:20:2	11	1	0.000138333		11	0.000150242
13		2	4			2006-05-10T13:2	2006-05-10T13:20:2	12	1	0.000146968		12	0.000157139
14		2	4			2006-05-10T13:2	2006-05-10T13:20:2	13	1	0.000138507		13	0.000204554
15		2	4			2006-05-10T13:2	2006-05-10T13:20:2	14	1	0.000138299		14	0.000215555
16		2	4			2006-05-10T13:2	2006-05-10T13:20:2	15	1	0.000138495		15	0.000238856
17		2	4			2006-05-10T13:2	2006-05-10T13:20:3	16	1	0.000138345		16	0.000280193
18		2	4			2006-05-10T13:2	2006-05-10T13:20:3	17	1	0.000138345		17	0.00032307
19		2	4			2006-05-10T13:2	2006-05-10T13:20:3	18	1	0.0013838		18	0.00032307
20		2	4			2006-05-10T13:2	2006-05-10T13:20:3	19	1	0.000138403		19	0.00035435
21		2	4			2006-05-10T13:2	2006-05-10T13:20:3	20	1	0.000138414		20	0.000377271
22		2	4			2006-05-10T13:2	2006-05-10T13:20:3	21	1	0.000138461			
23		2	4			2006-05-10T13:2	2006-05-10T13:20:3	22	1	0.000138241			
24		2	4			2006-05-10T13:2	2006-05-10T13:20:3	23	1	0.000138449			

Figure 9

To graph the average values select the “Message/Sec” and “Average Latency” columns (see *Figure 9*). Select the Standard XY (Scatter) chart type, click Next until the wizard ask where to create the new graph. It is best to create the chart as a new sheet. Click Finished. *Figure 10* shows the Chart Wizard.

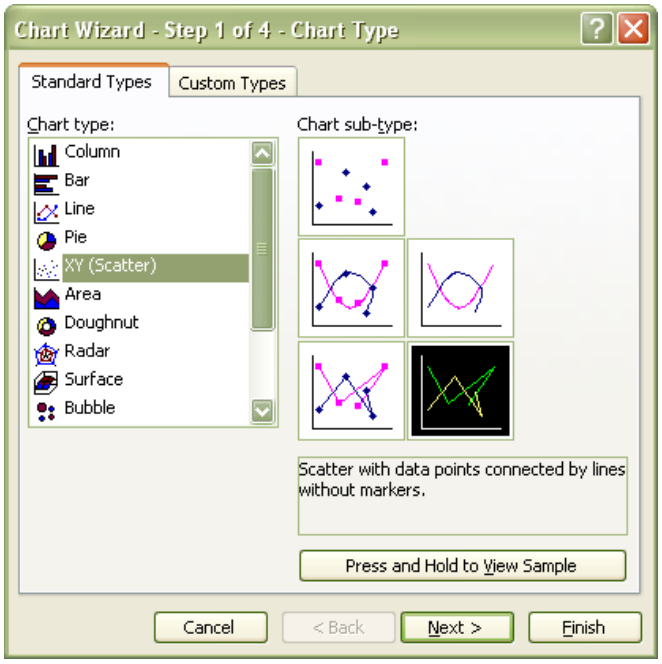


Figure 10

Figure 11 shows the graph with the legend removed and the X and Y axis formatted and labeled.

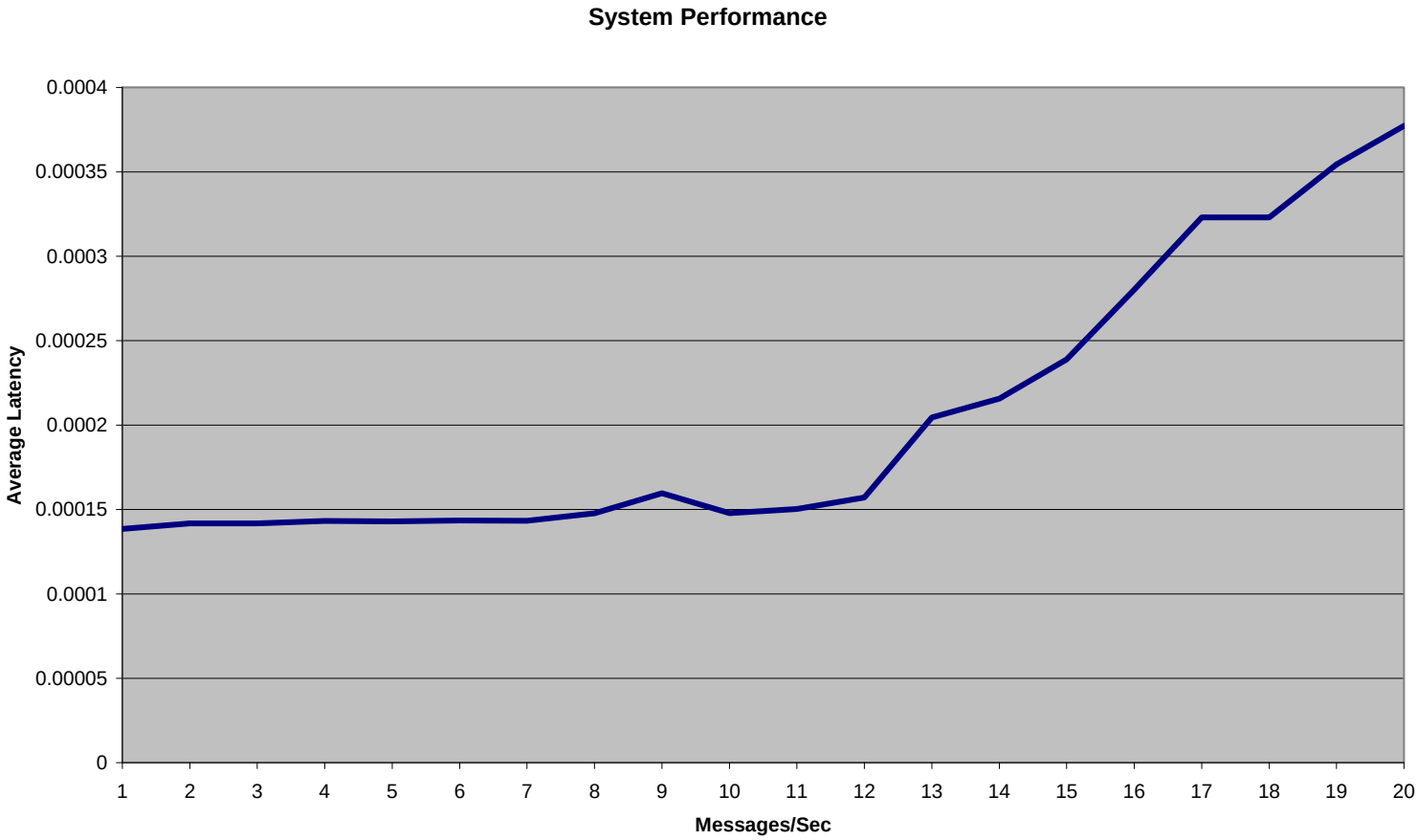

System Performance
Average Latency
0.0004
0.00035
0.0003
0.00025
0.0002
0.00015
0.0001
0.00005
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
Messages/Sec

Figure 11