Understanding Queue Manager Performance

MQ Technical Conference
V2.0.1.3
Introduction
Introduction

Richard Nikula

- VP of Product Development and Support

Involved in “MQ” since early 90’s

- Primarily at the technology layer

About Nastel Technologies

- Founded in 1994
- Middleware-centric Application Performance Management software supplier
- Core competency: Messaging Middleware, Java Application Servers, ESB’s and other SOA technologies
Nastel APWMQ Capabilities

AutoPilot for WebSphere MQ

- Configuration Management
  - Configuration Database
  - Change Management
- Message Management
  - Find, Fix & Reroute
  - View, Edit, Move & Replay
- Performance Monitoring
  - Performance
- History & Reporting
  - Capacity Planning
  - Accounting & Statistics
- Message Tracking
  - Track, Audit & Monitor
  - Transaction Management Database
- Extended Access
  - Self Service
  - Application Access
Overview

In this session, we will demonstrate how to get insight into the behavior of your queue managers using several techniques to determine.

• Do the queue managers perform worse at one time of day compared to another?
• What impact do different options have on message behavior?
• How do the channels perform between different queue managers?
• Where do message waits happen?

Agenda

• Introduction to Benchmarks
• Concepts required for developing a benchmark
• Synthetic versus real measurements
• Benchmark using a simple Ping
• Benchmark using Synthetic Message Tracking
• Tracking actual Messages
• Conclusion
Benchmarking
In computing, a benchmark is the act of running a computer program, a set of programs, or other operations, in order to assess the relative performance of an object, normally by running a number of standard tests and trials against it.

Benchmark Basics

Provides a baseline to which changes can be observed to determine the impact to the environment

• Using a consistent workload eliminates application changes and usage differences from influencing the tests
• Compare (not explain) differences in results between different configurations

Synthetics Workloads

• Use a standard set of activities
• Do the same thing over and over and should produce a consistent result

Real Workloads

• Measure real messages flowing through the systems
• Subject to application and usage differences
Where to Start?

- Observation
- Basic Ping
- Synthetic Message Tracking
- Actual Message Tracking
- Adjacent Application Tracking
Starting with a Basic Ping
Start with a Basic “Ping”

Periodically send a “ping” to a queue manager and measure the component parts

Active sonar. Copyright University of Rhode Island
What do we need?

Ping Component
- To Initiate the request
- Configurable to send different size batches and message sizes

Echo Component
- Listen for requests
- Sends a response

Analysis
- Measure the results
- Capture them into repository
- Analyze and produce results
- Alert to anomalous behavior
The Ping Component

In WebSphere MQ terms
• Puts a message to a queue
• Waits for the reply to be sent back
• Reports the results

Options
• A MQ program that you could write
• A script to call an MQSC Script
• 3rd party programs
  • Including (free) AutoPilot® MQSonar™ from Nastel
The Echo Component

In WebSphere MQ terms
• Listens on a queue for a message to arrive
• Puts a reply back to a “reply” queue

The WebSphere MQ “Command Server”
• Listens on Command Queue
• Supports a ping command

Alternatives
• A Program you write
• 3rd party programs
  • Including (free) AutoPilot® MQSonar™ from Nastel
The Analysis Component

Maintains a Historical Record
Analyzes the results

Options
• A program that you could write
• Interface with existing tooling
• Spreadsheet
• 3rd party options
  • Including AutoPilot® M6 from Nastel
In Operation

An Example

Ping

1.2 seconds

Queue

Echo

ReplyQ
But Wait! There’s More!

Become me
With Programming options we can measure more than Total Time
• Confirm on Arrival – a report message when placed on Queue
• Confirm on Delivery – a report message when delivered to application

0.2 seconds

0.4 seconds

1.2 seconds
Why this is Valuable

High Time to Confirm on Arrival
• Points to transmission path
Large time difference between Confirm on Arrival and Confirm on Delivery
• Points to server utilization or contention
Why this is Valuable

Difference between time spent outbound (propagation) and time spent on the return (reflection)

- Shows which path is contributing factor

![Diagram showing time taken between different nodes: Ping (0.4 seconds), Queue, Echo (0.8 seconds), ReplyQ]
Remote Queuing

But Wait! There’s More!
WebSphere MQ provides remote queuing

- Ping and Echo can be on different Queue Managers
Why this is Valuable

Same visibility as before across the intercommunication layer
- Can be done with or without Confirmation options

Ping → Remote Queue → Transmit Queue

Remote Queue → Sending Channel

Sending Channel → Queue

Queue → Receiving Channel

Receiving Channel → Transmit Queue

Transmit Queue → Sending Channel

Sending Channel → ReplyQ

ReplyQ → Receiving Channel

Receiving Channel → Echo

Echo
Multiple Queue Manager Scenarios

- Identify slowdowns in inter-queue manager communication
- Identify queue managers that contribute to delays
- Identify differences in behaviors of different queue managers
- Validate Cluster configurations

Verify that a path from one sending application to the receiving application is properly configured

  - Includes broker, DataPower, and other..
  - Requires the Echo component be an application program
Analysis
Statistics Summary

The following statistics can be derived from the samples:

- Average and maximum round trip times
- Total time for an entire batch of messages
- Propagation Time (from Ping to Echo)
- Reflection Time (from Echo to Ping)
- Message rates inbound and outbound
  - Actual and Theoretical
- Data rates outbound and inbound
More Statistics that can be produced

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MINIMUM_ROUND_TRIP</td>
<td>The minimum time for the reply to be returned for the request in seconds.</td>
</tr>
<tr>
<td>MAXIMUM_ROUND_TRIP</td>
<td>The maximum time for the reply to be returned for the request in seconds.</td>
</tr>
<tr>
<td>AVERAGE_ROUND_TRIP</td>
<td>The average time for the reply to be returned for the request in seconds for all messages sent in a batch.</td>
</tr>
<tr>
<td>AVERAGE_PROPAGATION_TIME</td>
<td>Average time in seconds for the message to propagate from the ping component to the echo component and the reply to be sent. For this statistic to be accurate, the time stamps between the servers sending the ping and sending the echo must be synchronized.</td>
</tr>
<tr>
<td>AVERAGE_REFLECTION_TIME</td>
<td>Average time in seconds for the message to reflect back from the echo component to the ping component and the reply to be read. For this statistic to be accurate, the time stamps between the servers sending the ping and sending the echo must be synchronized.</td>
</tr>
<tr>
<td>MESSAGES_SENT</td>
<td>The total number of messages sent included in this report.</td>
</tr>
<tr>
<td>CONFIRMED_EXPIRIES</td>
<td>Number of messages which expired before being delivered to the echo application (if coe specified).</td>
</tr>
<tr>
<td>CONFIRMED_DELIVERIES</td>
<td>Number of messages which were delivered to the echo application (if cod specified).</td>
</tr>
<tr>
<td>CONFIRMED_ARRIVALS</td>
<td>Number of messages which arrived to the echo application (if coa specified).</td>
</tr>
<tr>
<td>CONFIRMED_EXCEPTIONS</td>
<td>Number of messages which resulted in exceptions (if coe specified).</td>
</tr>
<tr>
<td>REPORTS_RECEIVED</td>
<td>Total number of confirmation report messages received.</td>
</tr>
<tr>
<td>RESPONSES_RECEIVED</td>
<td>Total number of response messages received.</td>
</tr>
<tr>
<td>MESSAGES_RECEIVED</td>
<td>Total number of response and report messages received.</td>
</tr>
<tr>
<td>BYTES_SENT</td>
<td>Total number of bytes sent.</td>
</tr>
<tr>
<td>BYTES_RECEIVED</td>
<td>Total number of bytes received.</td>
</tr>
<tr>
<td>RESPONSE_REQUEST_RATIO</td>
<td>The ratio of responses received to requests sent. A value of 100 means that all messages sent received responses.</td>
</tr>
</tbody>
</table>

General Performance Indicators

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TOTAL_PUT_TIME</td>
<td>The total time in seconds to put all of the messages</td>
</tr>
<tr>
<td>TOTAL_GET_TIME</td>
<td>The total time in seconds spent waiting for the responses to arrive</td>
</tr>
<tr>
<td>AVERAGE_PUT_RATE</td>
<td>The potential messages put rate calculated based on messages processed.</td>
</tr>
<tr>
<td>AVERAGE_PUT_BYTES_PER_SEC</td>
<td>Average put bytes per second</td>
</tr>
<tr>
<td>AVERAGE_GET_RATE</td>
<td>The potential messages get rate calculated based on messages processed.</td>
</tr>
<tr>
<td>AVERAGE_GET_BYTES_PER_SEC</td>
<td>Average get bytes per second</td>
</tr>
<tr>
<td>PUT_GET_RATIO</td>
<td>The ratio of put rate to get rate. A value greater than one means that MQSonar could put messages faster than get responses.</td>
</tr>
</tbody>
</table>

Message Performance Indicators

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GROSS_ROUND_TRIP_RATE</td>
<td>Message throughput rate (request + response) / time</td>
</tr>
<tr>
<td>GROSS_ROUND_BYTES_PER_SEC</td>
<td>Throughput (request + response) bytes per second</td>
</tr>
<tr>
<td>EFFECTIVE_ROUND_TRIP_RATE</td>
<td>Effective throughput rate (request + response + report) / time</td>
</tr>
<tr>
<td>CONFIRMATION_OVERHEAD</td>
<td>Percent of messages resulting from report options</td>
</tr>
<tr>
<td>AVERAGE_ARRIVAL_RATE</td>
<td>Average rate messages arrived at the destination (based on coa messages)</td>
</tr>
<tr>
<td>AVERAGE_DELIVERY_RATE</td>
<td>Average rate messages arrived at the destination (based on cod messages)</td>
</tr>
<tr>
<td>AVERAGE_MSG_LATENCY</td>
<td>Average time between arrival and delivery and number of messages on queue</td>
</tr>
<tr>
<td>AVERAGE_MSG_LATENCY_WITH_QDEPTH</td>
<td>Average time between arrival and delivery and number of messages on queue with queue depth</td>
</tr>
<tr>
<td>MAXIMUM_MSG_LATENCY</td>
<td>Maximum time between arrival and delivery and number of messages on queue</td>
</tr>
<tr>
<td>MAXIMUM_MSG_LATENCY_WITH_QDEPTH</td>
<td>Maximum time between arrival and delivery and number of messages on queue with queue depth</td>
</tr>
<tr>
<td>TOTAL_BATCH_TIME</td>
<td>The Elapsed time in seconds to process the entire batch.</td>
</tr>
<tr>
<td>TEST_COMPLETION_CODE</td>
<td>Completion code for the test. 0 indicates that all processing was normal.</td>
</tr>
</tbody>
</table>
Example Comparison using Excel

<table>
<thead>
<tr>
<th>Persistent Messages</th>
<th>Non Persistent Messages</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVERAGE_ROUND_TRIP</td>
<td>AVERAGE_ROUND_TRIP</td>
</tr>
</tbody>
</table>

Copyright © 2013 Nastel Technologies, Inc.
Is Faster Better than Slower?

Is Being Fast better than being slow?

But is it?
Another Comparison

A remote z/OS Queue Manager being accessed over channel over a secure connection

A local Queue Manager
Integration with AutoPilot (optional)

Statistics Collected are published to AutoPilot Server

- Alert on abnormal results
- Historical trending
Tracking Messages
Concepts

Requirements for the ping technique:
• “Special” messages
• “Special” message paths

Using MQ Exit points you can capture the actual MQ calls and messages without changing them
• Capture synthetic message flows
• Capture real message flows using actual application logic
• Richer detail on the message flow
Message Intercept

Captures details about the message
• Header details (time sent, size, …)
• Application details
• All or part of payload

Options
• A MQ program that you could write
• IBM Support Pac (mirrorq, MA0W)
• 3rd party programs
  • Including AutoPilot® TransactionWorks™ from Nastel
How it works

Message Intercept

Application

WebSphere MQ

Exit Program

Collected Data

MQCONN
MQOPEN
MQPUT
MQGET

MQOPEN
MQPUT
MQGET

MQOPEN
MQPUT
MQGET

Copyright © 2013 Nastel Technologies, Inc.
With Remote Queueing

1. Requestor
2. Remote Queue
3. Queue
4. Provider
5. Transmit Queue
6. Sending Channel
7. Receiving Channel
8. ReplyQ

Copyright © 2013 Nastel Technologies, Inc.
## Cross Application Tracking

### Time Tracking Table

<table>
<thead>
<tr>
<th>Time</th>
<th>Operation Name</th>
<th>Resource</th>
<th>Elapsed Time (ms)</th>
<th>Message Id</th>
<th>Message Age (ms)</th>
<th>Completion Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>2013-03-26 13:41:47.17</td>
<td>/trading/verification</td>
<td>/trading/verification</td>
<td>7875</td>
<td>0</td>
<td>0</td>
<td>Succeeded</td>
</tr>
<tr>
<td>2013-03-26 13:41:47.19</td>
<td>com/acs/trading/verify.start</td>
<td>java</td>
<td>5749</td>
<td>0</td>
<td>0</td>
<td>Succeeded</td>
</tr>
<tr>
<td>2013-03-26 13:41:47.19</td>
<td>HTTP/GET/trading/verification</td>
<td>/trading/verification</td>
<td>7665</td>
<td>2195</td>
<td>0</td>
<td>Succeeded</td>
</tr>
<tr>
<td>2013-03-26 13:41:47.19</td>
<td>Statement.executeQuery(String)</td>
<td>MySQL5.1.4DB3en</td>
<td>1823</td>
<td>2196</td>
<td>0</td>
<td>Succeeded</td>
</tr>
<tr>
<td>2013-03-26 13:41:47.20</td>
<td>com/acs/trading/verify.request java</td>
<td></td>
<td>1495</td>
<td>0</td>
<td>0</td>
<td>Succeeded</td>
</tr>
<tr>
<td>2013-03-26 13:41:47.20</td>
<td>QueueSender.send</td>
<td>queue/trading/</td>
<td>6565</td>
<td>2186</td>
<td>0</td>
<td>Succeeded</td>
</tr>
<tr>
<td>2013-03-26 13:41:47.20</td>
<td>MQPUT</td>
<td>TradeVerification</td>
<td>10107</td>
<td>2186</td>
<td>0</td>
<td>Succeeded</td>
</tr>
<tr>
<td>2013-03-26 13:41:47.22</td>
<td>MQGET</td>
<td>TradeVerification</td>
<td>10900</td>
<td>2106</td>
<td>0</td>
<td>Succeeded</td>
</tr>
<tr>
<td>2013-03-26 13:41:47.22</td>
<td>MQPUT</td>
<td>TradeValidation.1</td>
<td>9399</td>
<td>2107</td>
<td>0</td>
<td>Succeeded</td>
</tr>
<tr>
<td>2013-03-26 13:41:47.24</td>
<td>MQGET</td>
<td>TradeValidation.1</td>
<td>9507</td>
<td>2187</td>
<td>116</td>
<td>Succeeded</td>
</tr>
<tr>
<td>2013-03-26 13:41:47.24</td>
<td>MQPUT</td>
<td>TradeValidation.1</td>
<td>9803</td>
<td>2188</td>
<td>0</td>
<td>Succeeded</td>
</tr>
<tr>
<td>2013-03-26 13:42:23.22</td>
<td>MQGET</td>
<td>TradeValidation.1</td>
<td>25970000</td>
<td>2188</td>
<td>652</td>
<td>Succeeded</td>
</tr>
<tr>
<td>2013-03-26 13:42:23.22</td>
<td>MQPUT</td>
<td>TradeAudit.Input.</td>
<td>10511</td>
<td>2189</td>
<td>0</td>
<td>Succeeded</td>
</tr>
</tbody>
</table>
Application Dashboard

**fincorp INVESTOR SERVICES**
- Trading Value: EUR 33,324,142 €
- Trading Volume: Current Volume: 31,102
- Service Level Agreement

**Trade Verification Enrichment**
- Trade Verification: Tx / SLA Limit = 81.8%

**Trade Audit**
- Trade Audit: Tx / SLA Limit = 89.2%

**Trade Validation**
- Trade Validation: Tx / SLA Limit = 58.7%

**ForEx Trading**
- FX Trading: Tx / SLA Limit = 84.7%

**IBM WebSphere**
- WMQ
- WMB

**IBM WebSphere DataPower**
- DataPower SOA Appliance

**TIBCO EMS**
- EMS

**TIBCO Rendezvous**
- RV

**Infrastructure**
Using the MQ Recovery Log Files

Log files can be used to get insight into MQ message traffic
Automatically generated so no special setup required to collect them
But some challenges

• Capture recoverable messages only
• Don’t capture error conditions (queue full, not authorized)
• Not easily to expand to other applications
• Lack of real time awareness of problems
Summary

• Starting with a Simple benchmark to periodically ping a queue manager can be valuable
• Combining with other options, such as Confirm on Delivery increases the information that can be determined
• Performing this regularly provides insight into the changes in behavior of the queue manager
• Tracking actual messages increases the data available about the queue manager operation
Thank You!

Questions:
info@nastel.com