

***Using Transaction Tracing to Determine Issues with Remote  
MQ Transactions***

***Richard Nikula  
VP, Product Development and Support***

# INTRODUCTION

# Introduction

- **Richard Nikula**
  - ▶ VP of Product Development and Support

3

- **Developing software for management of “middleware” since 1985**
  - ▶ MAINVIEW for CICS, MQ
  - ▶ BMC PATROL
  - ▶ IBM CICSPLEX System Manager (CPSM)
- **Involved in “MQ” since early 90’s**
  - ▶ Primarily at the technology layer
  - ▶ Various certifications
- **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

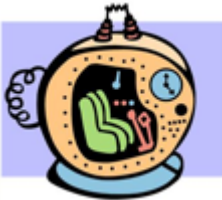
# Overview

- MQ transactions can run on a number of different platforms and locations. They typically interact with other environments such as IBM Integration Bus (Broker) and DataPower. It can be challenging to track the flow of activities in these environments.

- In this session, you will learn:
  - The facilities provided by MQ for tracking MQ activity without changing application code
  - The facilities provide by IIB/Broker for tracking activity within message flows
  - Techniques for correlating data between broker and MQ for cross platform visibility
  - How similar techniques could be used for extending tracking to application servers, DataPower or other platforms

# PRESENTATION...

# Time Travel



**1993**



# Your workstation



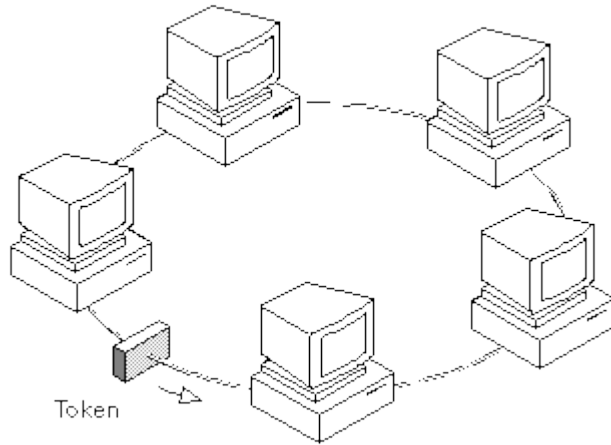
Operating system PC DOS 4.01  
CPU Intel 80386SX @ 16 MHz  
Memory 2 MB ~ 6 MB



7 color  
Reverse video and blink  
Graphical display capable

# Your Network

8

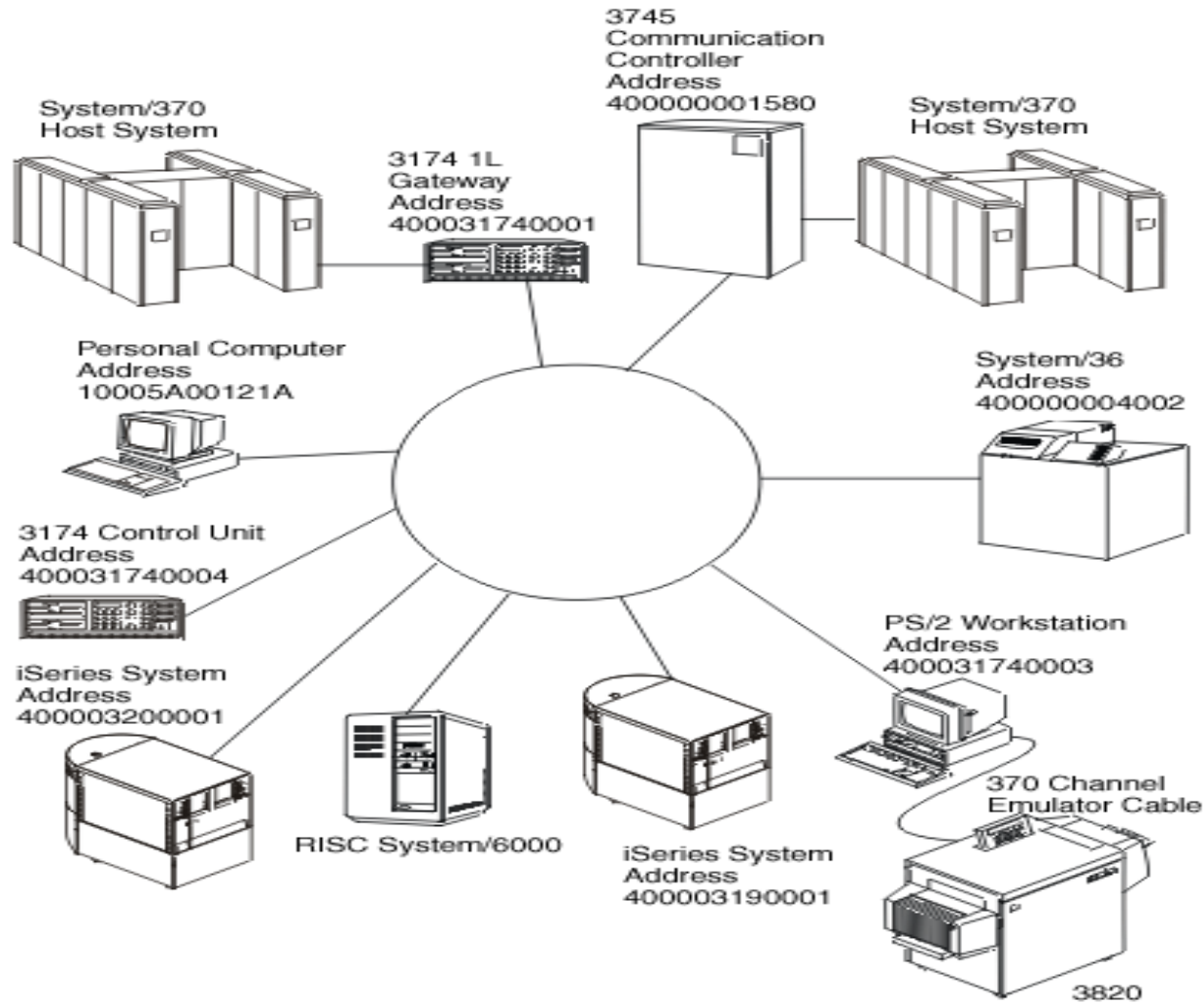


Copyright © 1995 United Feature Syndicate, Inc.  
Redistribution in whole or in part prohibited



# Your Enterprise

9



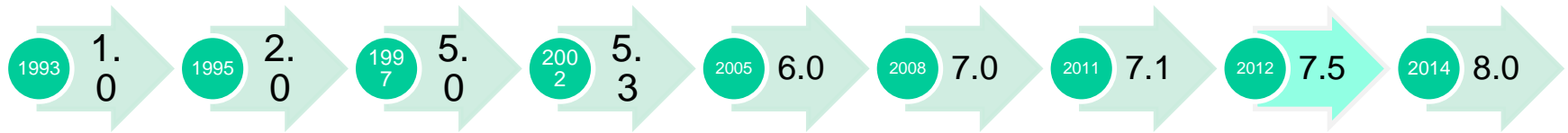
# Your remote network



Network Speed: 2400/4800/9600 Baud  
Reliability?

# MQ Timeline from 1993 to Today

11



# Why use our time machine?

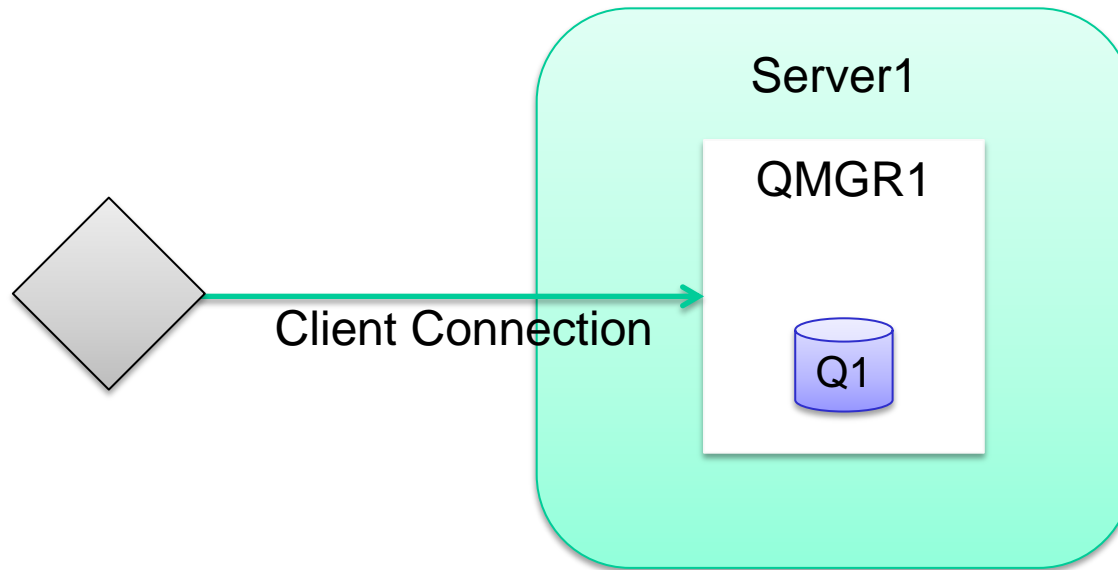
Why use our time machine?



Because we wouldn't have had this discussion in the past.  
Today's environments make remote operation of MQ a requirement.

# CONCEPTS OF REMOTE MANAGEMENT

# You are remote to the system being managed



Not Remote Desktop or Telnet

No install of software on the Server1 (M2000 doesn't allow)

# Types of Remote Access

- **Application**

- **The reason MQ exists**
- 15      - But new considerations in a fully remote environment

- **Administrative**

- **Configuring MQ**
  - Traditional MQ Explorer or similar 3<sup>rd</sup> party tools

- **Diagnostic**

- **Looking at queue managers, queues and messages**
  - Traditional MQ Explorer or similar 3<sup>rd</sup> party tools
- **Tracking messages**
  - New tools are needed

# Primary Consideration for Remote Access

- **Connectivity**

- **Have to have access**
  - Firewalls and network

- **Security**

- **Must be authorized to perform the actions requested**
  - Access control (e.g. setmqaut)
  - AMS (Advanced Message Security)
  - SSL (communication)
  - Channel Authorization (V7.1 and higher)
  - Connection Authentication (V8 and higher)

- **Performance**

- **Has to be able to provide the bandwidth required**

- **Not always the right choice**



# ADMINISTRATION

# Administration

18

Logged in as Admin

Object Name: \\MQM\REMOTE\_QMGRS\MYQM

Attribute schema: Default Queue Manag

Attribute Name	Attribute Value
Accounting Connection Override	Disabled
Accounting Interval (sec.)	1800
Active Channels	0
Activity Recording	MESSAGE
Adopt MCA	
Alteration Date	2015-02-18
Authority Events	Disabled
Auto Cluster Sender Monitoring	Queue Manager
Auto Cluster Sender Statistics	Queue Manager
Behavior Of Undelivered Response Messages	NORMAL
Bridge Events	
CCSID	819
CF Structures Connection Loss Action	Terminate
Channel Auth. Record Checking	Disabled
Channel Auto-Definition	Disabled
Channel Auto-Definition Events	Disabled
Channel Auto-Definition Exit	Disabled
Channel Events	Disabled
Channel Initiator Adapter Subtasks	
Channel Initiator Control	QMGR
Channel Initiator Dispatchers	
Channel Initiator's Trace Data Space Size (MB)	
Channel Monitoring	Disabled
Channel Statistics	Disabled
Cluster Workload Data	
Cluster Workload Exit	
Cluster Workload Length (bytes)	100
Cluster Workload MRU Channels	999999999
Cluster Workload Use Queue	Local Queue
Command Audit Events	0
Command Input Queue Name	SYSTEM.ADMIN.COMMAND.QUEUE
Command Level	801
Command Server Control	QMGR
Configuration Events	0
Creation Date	2015-02-18
Dead Letter Queue	
Default Cluster Transmission Queue Type	SYSTEM.CLUSTER.TRANSMIT.QUEUE
Default Transmission Queue Name	

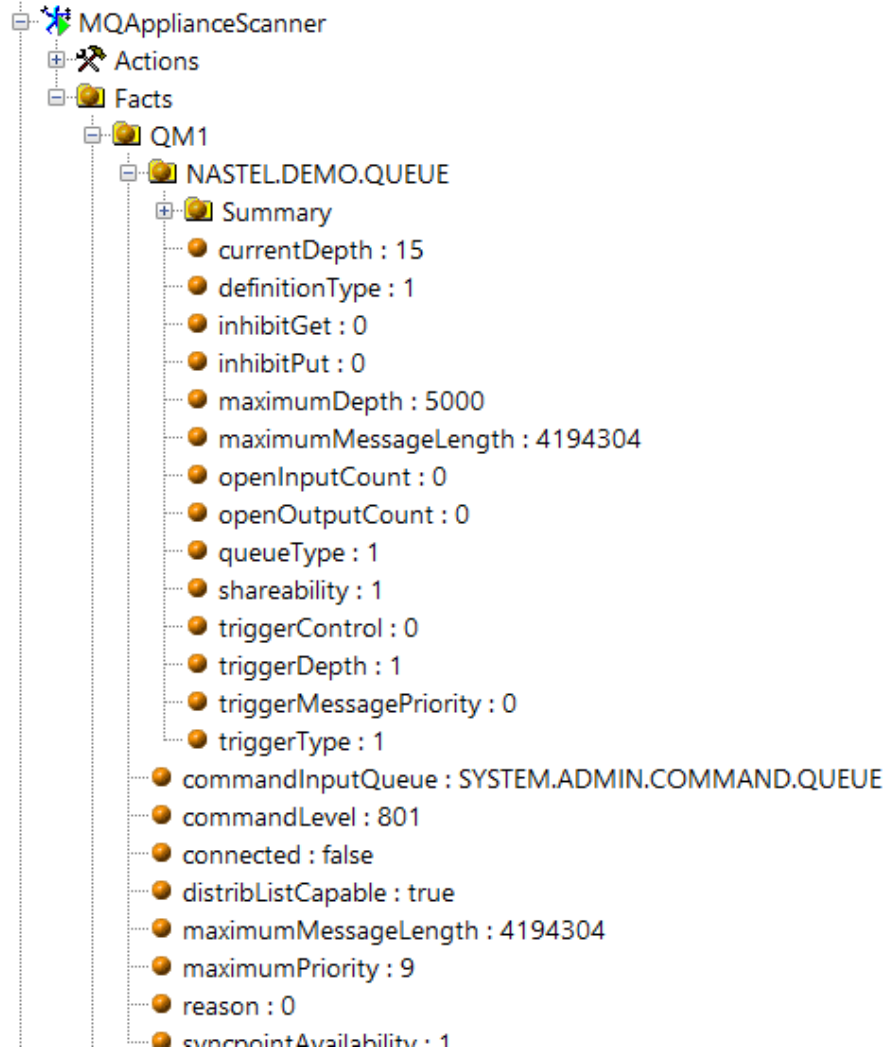
RemoteQM1 on '192.168.75.135(1415)'

Connected	
Client	192.168.75.135(1415)
Channel	SYSTEM.ADMIN.SVRCONN
Queue manager name	RemoteQM1
Description	
Platform	[Unknown]
Command level	801
Version	08000002
Default transmission queue	
Last updated:	10:13:39

- JMS Administered Objects
- Managed File Transfer
- Service Definition Repositories

Queue manager name	RemoteQM1
Description	
Platform	[Unknown]
Command level	801
Version	08000002
Default transmission queue	
Last updated:	10:13:39

# Remote Management Example



This slide was added to show how easily remote management is. This was an MQ appliance being demonstrated by IBM at the conference to which they provided access. It shows that the advantage of remote access is how easily it can be done and the disadvantage is how easily it can be done.

20

# TRACKING USING APPLICATION ACTIVITY TRACE

# Application Activity Trace

- Creates an event message for MQ calls
  - Similar to using MQ API before/after exits
  
- Introduced in MQ 7.1
  - Expanded with MQ Appliance
  - Continuing improvements being made in V8
  
- Not available for MQ on zOS ☹

# Using in MQ V7.1, 7.5 and V8\*

- **Configure mqat.ini (to add application)**
- **Execute Application to be analyzed**
- **Configure mqat.ini**
- **Configure mqat.ini (to remove application)**
  - ▶ If running, change something in QMGR
- **Data captured on SYSTEM.ADMIN.TRACE.ACTIVITY.QUEUE**
- **View Data**
  - **amqsact**
  - **Amqsactz (“freeware”)**
  - **Ms0P**
  - **3<sup>rd</sup> Party tooling**

# Global Monitoring

Alternately, turn on/off Activity trace for “everything”

- Alter QMGR ACTVTRC(ON/OFF)

Change mqat.ini to exclude “everything” by default

- Applications that you don't want
- Monitoring tools
- IBM tools (amqsact)

# MQAT.INI

```
*****#
#* Module Name: mqat.ini                               *#
#* Type       : WebSphere MQ queue manager configuration file *#
# Function    : Define the configuration of application activity *#
#*           : trace for a single queue manager.           *#
#*           *#
#*           *#
*****#
```

```
AllActivityTrace:           # Global settings stanza
  ActivityInterval=1        # Time interval between trace messages
                            # Values: 0-99999999 (0=off)
                            # Default: 0
  ActivityCount=100        # Number of operations between trace msgs
                            # Values: 0-99999999 (0=off)
                            # Default: 0
  TraceLevel=MEDIUM       # Amount of data traced for each operation
                            # Values: LOW | MEDIUM | HIGH
                            # Default: MEDIUM
  TraceMessageData=0      # Amount of message data traced
                            # Values: 0-104857600
                            # Default: 0
  StopOnGetTraceMsg=ON    # Stop trace on get of activity trace message
                            # Values: ON | OFF
                            # Default: ON
```

24



# MQAT.INI (application specific)

```
#####  
# specific application activity trace entry from generating data #  
#####  
ApplicationTrace:          # Application specific settings stanza  
    ApplClass=ALL          # Application type  
                            # Values: (USER | MCA | ALL)  
                            # Default: USER  
    ApplName=amqsput*      # Application name (may be wildcarded)  
                            # (matched to app name without path)  
                            # Default: *  
    ApplFunction=*         # Application function (may be wildcarded)  
                            # (matched to app function)  
                            # Default: *  
    Trace=ON               # Activity trace switch for application  
                            # Values: ( ON | OFF )  
                            # Default: OFF  
    ActivityInterval=0     # Time interval between trace messages  
                            # Values: 0-99999999 (0=off)  
                            # Default: 0  
    ActivityCount=0        # Number of operations between trace msgs  
                            # Values: 0-99999999 (0=off)  
                            # Default: 0  
    TraceLevel=HIGH       # Amount of data traced for each operation  
                            # Values: LOW | MEDIUM | HIGH  
                            # Default: MEDIUM  
    TraceMessageData=1000 # Amount of message data traced  
                            # Values: 0-104857600  
                            # Default: 0
```

25

# MQAT.INI (application specific)

```
#####  
# Prevent the sample activity trace program from generating data #  
#####  
ApplicationTrace:          # Application specific settings stanza  
    ApplClass=ALL          # Application type  
                            # Values: (USER | MCA | ALL)  
                            # Default: USER  
    ApplName=amqsact*      # Application name (may be wildcarded)  
                            # (matched to app name without path)  
                            # Default: *  
    ApplFunction=*        # Application function (may be wildcarded)  
                            # (matched to app function)  
                            # Default: *  
    Trace=OFF              # Activity trace switch for application  
                            # Values: ( ON | OFF )  
                            # Default: OFF  
    ActivityInterval=0     # Time interval between trace messages  
                            # Values: 0-99999999 (0=off)  
                            # Default: 0  
    ActivityCount=0        # Number of operations between trace msgs  
                            # Values: 0-99999999 (0=off)  
                            # Default: 0  
    TraceLevel=MEDIUM     # Amount of data traced for each operation  
                            # Values: LOW | MEDIUM | HIGH  
                            # Default: MEDIUM  
    TraceMessageData=0    # Amount of message data traced  
                            # Values: 0-104857600  
                            # Default: 0
```

26

# amqsact

Usage: amqsact

```
[-m QMgrName]      # Queue manager to connect to
[-q QName]         # Override default queue name
[-t TopicString]  # Subscribe to event topic
[-b]              # Only browse records
[-v]              # Verbose output
[-d <depth>]      # Number of records to display
[-w <timeout>]    # Time to wait (in seconds)
[-s <startTime>] # Start time of record to process
[-e <endTime>]   # End time of record to process
```

Example:

```
amqsact -m QMGR1 -b -v
```

## Sample AMQSACT amqsact -m LocalQM1 -v (edited)

MonitoringType: MQI Activity Trace

Correl\_id:

00000000: 414D 5143 4C6F 6361 6C51 4D31

2020 2020 'AMQCLocalQM1....'

00000010: 7186 6B55 2000 3602

'qåkU..6. '

QueueManager: 'LocalQM1'

Host Name: 'RNIKULA-PC'

CommandLevel: 800

SeqNumber: 1

ApplicationName: 'bSphere  
MQ\bin64\amqsput.exe'

Application Type: MQAT\_WINDOWS\_NT

ApplicationPid: 5004

UserId: 'Richard'

Pointer size: 8

Platform: MQPL\_WINDOWS\_NT

Operation Id: MQXF\_PUT

OperationDate: '2015-05-31'

OperationTime: '17:21:46'

High Res Time: 1433110906406803

QMgr Operation Duration: 70

Completion Code: MQCC\_OK

Reason Code: 0

Msg length: 28

Object\_type: MQOT\_Q

Object\_name: 'Q1'

Object\_Q\_mgr\_name: ''

Resolved\_Q\_Name: 'Q1'

Resolved\_Q\_mgr: 'LocalQM1'

Resolved\_local\_Q\_name: 'Q1'

Resolved\_local\_Q\_mgr: 'LocalQM1'

Resolved\_type: MQOT\_Q

Report Options: 0

Msg\_type: MQMT\_DATAGRAM

Expiry: -1

Format\_name: 'MQSTR'

Priority: -1

Persistence: 2

MQCFH (PCF Header)

Use this page to view the PCF values contained by the MQCFH structure for an activity trace message

29

For an activity trace message, the MQCFH structure contains the following values: Type

Description: Structure type that identifies the content of the message.

Data type: MQLONG.

Value: MQCFT\_APP\_ACTIVITY

StrucLength

Description: Length in bytes of MQCFH structure.

Data type: MQLONG.

Value: MQCFH\_STRUC\_LENGTH

Version

Description: Structure version number.

Data type: MQLONG.

Values: MQCFH\_VERSION\_3

Command

Description: Command identifier. This field identifies the category of the message.

Data type: MQLONG.

Values: MQCMD\_ACTIVITY\_TRACE

MsgSeqNumber

Description: Message sequence number. This field is the sequence number of the message within a group of

Data type: MQLONG.

Values: 1

Control

Description: Control options.

Data type: MQLONG.

Values: MQCFT\_LAST

CompCode

# Notes

- **What are you Tracing?**

- **If you turn on at the Queue Manager Level, most applications**

- **Unless application connects using MQCONNX using**

- MQCNO\_ACTIVITY\_TRACE\_DISABLED

- **Need to edit MQAT.INI to specific applications**

- How to know which applications read which queues?

- **Who can view the trace?**

- **Anyone with access to the queue**

- **Information mixed with other users**

- **Other users could remove your information**

- **Tricky to get just your information**

# Using in MQ Appliance using V8\*

- (Configure mqat.ini to change defaults)
- Use Dynamic mode to collect trace data
- View Data
  - amqsact (c)
  - Ms0P
  - 3<sup>rd</sup> Party tooling

31

# dspmqini

```
M2000(mqcli)# dspmqini -m MQAPP_QM1
```

```
AllActivityTrace:
```

```
ActivityInterval    = 1
```

```
ActivityCount       = 100
```

```
TraceLevel          = MEDIUM
```

```
TraceMessageData    = 0
```

```
StopOnGetTraceMsg   = ON
```

```
SubscriptionDelivery = BATCHED
```

32



# Setmqini

## M2000(mqcli)# setmqini

Usage: setmqini -m QMgrName -s StanzaName -k KeyName [-d | -v Val

33

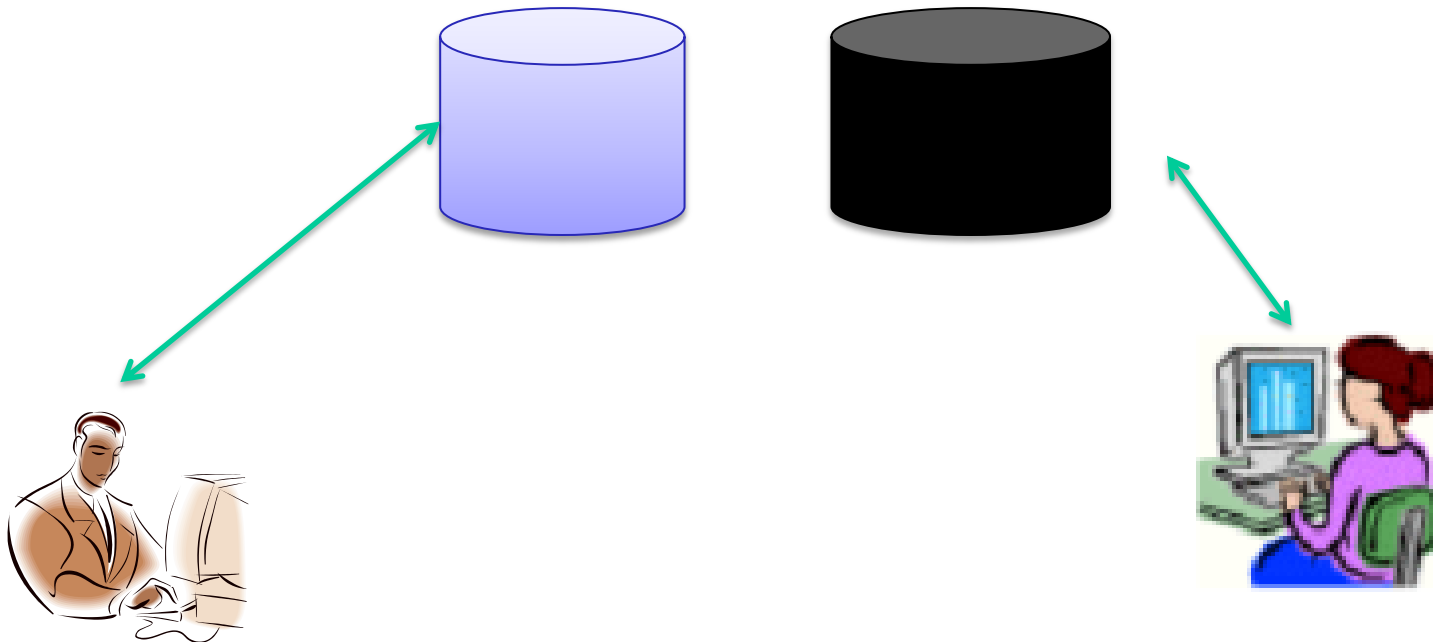
- d Delete the attribute.
- k Key name of attribute to set.
- m Queue manager name.
- s Stanza name.
- v Attribute value to set.

### Example:

```
setmqini -m QMAPP_QM1 -s AllActivityTrace -k TraceLevel -v HIGH
```

# Use Case for Application Activity Tracing (V8\*)

- Opens up new use cases
  - Traditionally tracing an administration function
  - Now can be done at individual developer level



# Amqsact(c)

Usage: amqsact (c)

```
[-m QMgrName] # Queue manager to connect to
[-a ApplName] # Name of application to trace
[-c ChannelName] # Name of channel to trace
[-i ConnId] # Unique connection id to trace
[-q QName] # Override default queue name
[-t TopicString] # Subscribe to event topic
[-b] # Only browse records
[-v] # Verbose output
[-d <depth>] # Number of records to display
[-w <timeout>] # Time to wait (in seconds)
[-s <startTime>] # Start time of record to process
[-e <endTime>] # End time of record to process
```

Example:

```
amqsact -m QMGR1 -w 30 -a amqsput.exe
```

# Dynamic Subscription

```
amqsact -mLocalQM1 -w 60 -a amqsput.exe
```

Subscribing to the activity trace topic:

```
'$SYS/MQ/INFO/QMGR/LocalQM1/ActivityTrace/AppName/amqsput.exe'
```

36

The screenshot shows the 'General' tab of the MQ console configuration for a dynamic subscription. The configuration is as follows:

- Subscription name:** (empty field)
- Topic:**
  - Topic name:** (empty field)
  - Topic string:** `$SYS/MQ/INFO/QMGR/LocalQM1/ActivityTrace/AppName/amqsput.exe`
- Wildcard usage:** `Topic level wildcard`
- Scope:** `All`
- Destination:**
  - Destination class:** `Managed`
  - Destination queue manager:** `LocalQM1`
  - Destination name:** `SYSTEM.MANAGED.NDURABLE.556B867103220020`
  - Correlation identifier:**

00000	41	4D	51	20	4C	6F	63	61	--6C	51	4D	31	20	20	20	20	20
00010	71	86	6B	55	20	00	22	04	--								

Navigation buttons: < > Edit...

# Amqsact(c) remote access

```
amqsactc -m RemoteQM1 -w 60 -a amqsputc.exe -v
```

```
Subscribing to the activity trace topic:
```

```
'$SYS/MQ/INFO/QMGR/RemoteQM1/ActivityTrace/AppName/amqsputc.exe'
```

37

```
MonitoringType: MQI Activity Trace
```

```
Correl_id:
```

```
00000000: 414D 5120 5265 6D6F 7465 514D 3120 2020 'AMQ.RemoteQM1.'
```

```
00000010: 0D4D 6B55 1023 0020 '.MkU.#..'
```

```
QueueManager: 'RemoteQM1'
```

```
CommandLevel: 801
```

```
SeqNumber: 0
```

```
ApplicationName: 'Sphere MQ\bin64\amqsputc.exe'
```

```
Application Type: MQAT_WINDOWS_NT
```

```
ApplicationPid: 14666
```

```
UserId: 'mqm'
```

```
API Caller Type: MQXACT_EXTERNAL
```

```
API Environment: MQXE_MCA_SVRCONN
```

```
Channel Name: 'SYSTEM.ADMIN.SVRCONN'
```

```
ConnName: '192.168.75.1'
```

```
Channel Type: MQCHT_SVRCONN
```

# Tracking activity from a Business Partner

```
amqsactc -m RemoteQM1 -w 60 -c From.BP0302.Primary -v
```

Subscribing to the activity trace topic:

```
'$SYS/MQ/INFO/QMGR/RemoteQM1/ActivityTrace/ChannelName/From.BP0302.Primary
```

38

```
MonitoringType: MQI Activity Trace
```

```
Correl_id:
```

```
00000000: 414D 5120 5265 6D6F 7465 514D 3120 2020 'AMQ.RemoteQM1..
```

```
00000010: 0D4D 6B55 0429 0020 '.MkU.).. '
```

```
QueueManager: 'RemoteQM1'
```

```
ApplicationName: 'amqrmppa'
```

```
Application Type: MQAT_QMGR
```

```
ApplicationPid: 14666
```

```
UserId: 'mqsystem'
```

```
API Caller Type: MQXACT_INTERNAL
```

```
API Environment: MQXE_MCA
```

```
Channel Name: 'From.BP0302.Primary'
```

```
ConnName: '192.168.75.1'
```

```
Channel Type: MQCHT_RECEIVER
```

# Operation 0

MQI Operation: 0

Operation Id: MQXF\_CONNX

ApplicationTid: 21

OperationDate: '2015-05-31'

OperationTime: '14:33:35'

ConnectionId:

00000000: 414D 5143 5265 6D6F 7465 514D 3120 2020 'AMQCRemoteQM

00000010: 0D4D 6B55 0128 0020 '.MkU(.. ')

QueueManager: 'RemoteQM1'

QMgr Operation Duration: 96098215

Completion Code: MQCC\_OK

Reason Code: 0

Connect Options: 1

39

# Operation 1

MQI Operation: 1

Operation Id: MQXF\_OPEN

ApplicationTid: 21

OperationDate: '2015-05-31'

OperationTime: '14:33:35'

Object\_type: MQOT\_Q

Object\_name: 'Q1'

Object\_Q\_mgr\_name: 'RemoteQM1'

QMgr Operation Duration: 74

Completion Code: MQCC\_OK

Reason Code: 0

Open\_options: 43024

Object\_type: MQOT\_Q

Object\_name: 'Q1'

Object\_Q\_mgr\_name: 'RemoteQM1'

Resolved\_Q\_Name: 'Q1'

Resolved\_Q\_mgr: 'RemoteQM1'

Resolved\_local\_Q\_name: 'Q1'

Resolved\_local\_Q\_mgr: 'RemoteQM1'

Resolved\_type: MQOT\_Q

40



# Operation 2

MQI Operation: 2

Operation Id: MQXF\_PUT

ApplicationTid: 21

OperationDate: '2015-05-31'

OperationTime: '14:33:35'

High Res Time: 1433097215499226

QMgr Operation Duration: 44

Completion Code: MQCC\_OK

Reason Code: 0

Hobj: 2

Put Options: 272388

Msg length: 20

Object\_Q\_mgr\_name: 'RemoteQM1'

Resolved\_Q\_Name: 'Q1'

Resolved\_Q\_mgr: 'RemoteQM1'

Resolved\_local\_Q\_name: 'Q1'

Resolved\_local\_Q\_mgr: 'RemoteQM1'

Resolved\_type: MQOT\_Q

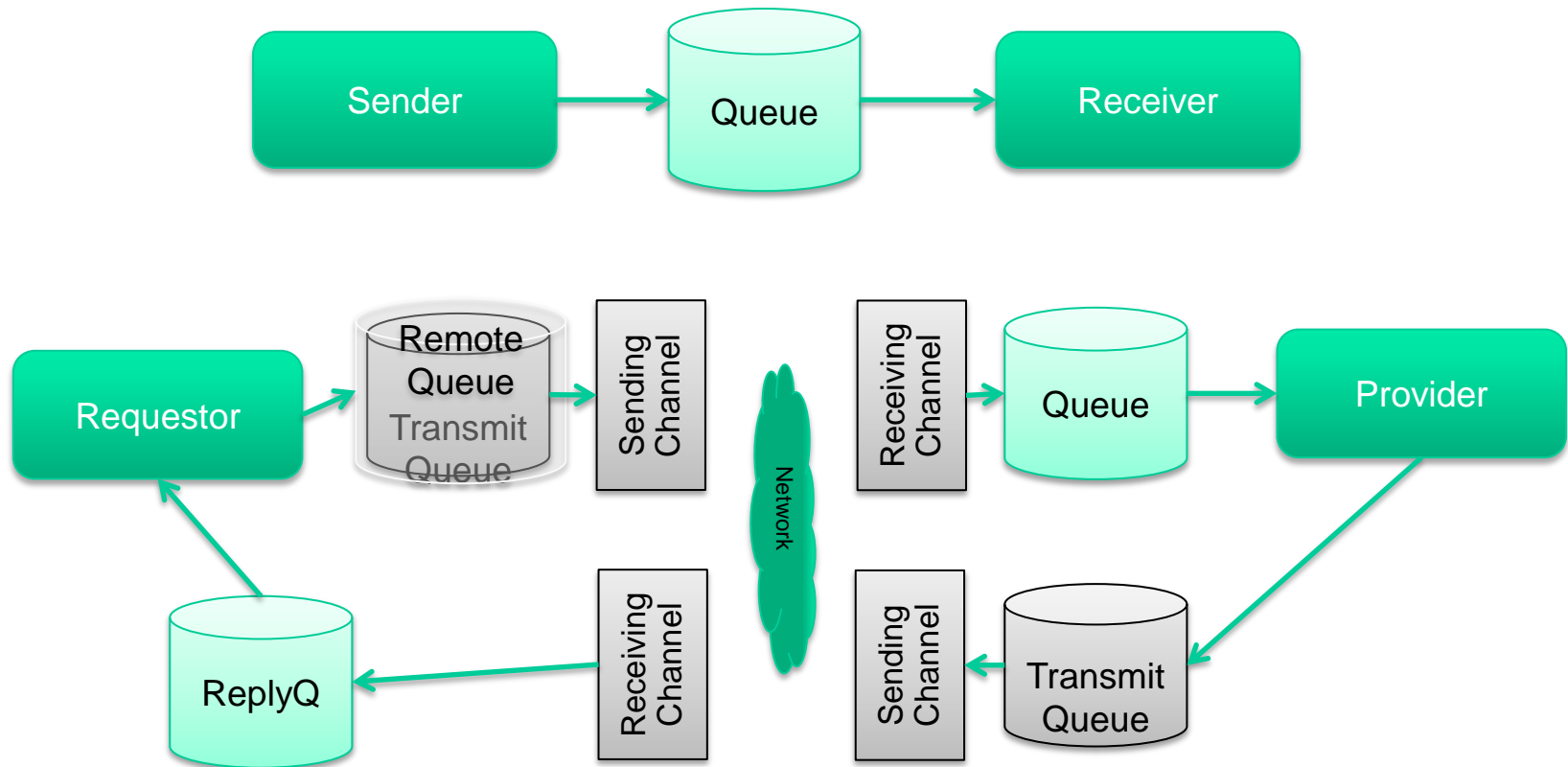
Expiry: -1

Format\_name: 'MQSTR'

# Considerations

- How to Pair up put and get of the same messages?
- Dealing with complex message flows.

42



# Sample display

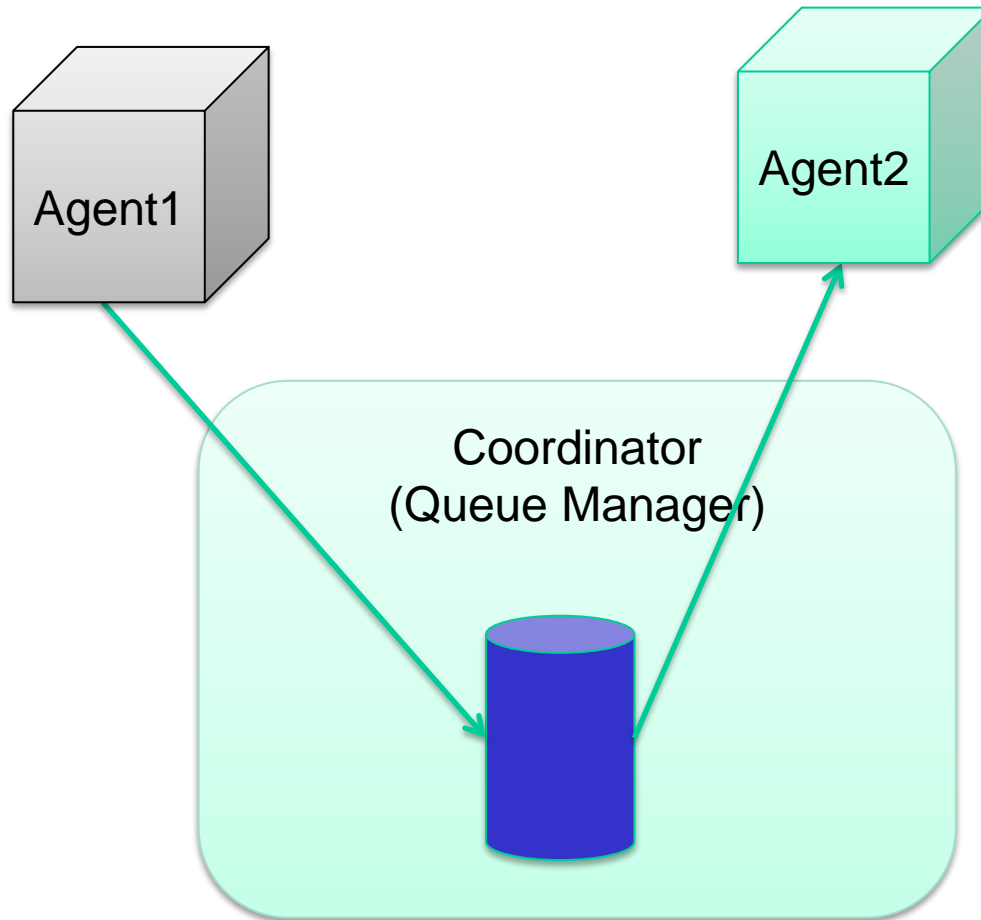
Transaction Flow Diagram | Transaction Timeline | Transaction Trace(3) | Transaction Milestones

Show Hierarchy

Time	Application	Operation Name	Resource	Resource Manager	Elapsed Time (usec)	Completion Code ▲	Reason Code
2015-06-01 10:52:15.000	amqspout	MQPUT	To.PayRoll.Primary	LocalQM1	69	Succeeded	0
2015-06-01 10:52:18.000	WMQ_ChannelPooling...	MQPUT	PayRoll.Primary	RemoteQM1	68	Succeeded	0
2015-06-01 10:52:24.000	amqsgetc	MQGET	PayRoll.Primary	RemoteQM1	39	Succeeded	0

# FTE/MFT

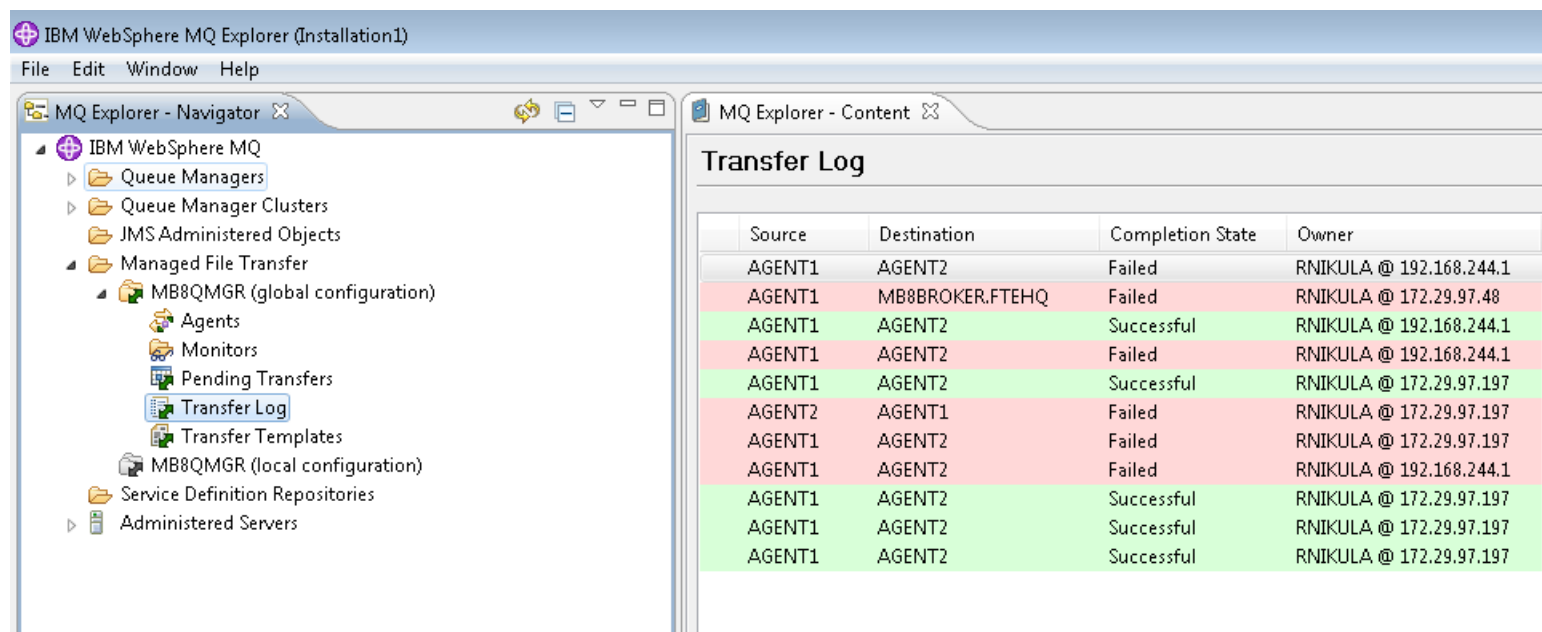
# What is it?



# Subscribing to events

- `SYSTEM.FTE/Log/#`
  
- Types of data
  - Agents
  - Transfers
  - Schedules
  - Logs

# MQ Explorer Views



The screenshot displays the IBM WebSphere MQ Explorer interface. The left pane shows a tree view of the MQ environment, with the 'Transfer Log' folder selected under 'Managed File Transfer > MB8QMGR (global configuration)'. The right pane displays the 'Transfer Log' table, which contains 12 rows of data. The table has four columns: Source, Destination, Completion State, and Owner. The rows are color-coded: red for 'Failed' and green for 'Successful'.

Source	Destination	Completion State	Owner
AGENT1	AGENT2	Failed	RNIKULA @ 192.168.244.1
AGENT1	MB8BROKER.FTEHQ	Failed	RNIKULA @ 172.29.97.48
AGENT1	AGENT2	Successful	RNIKULA @ 192.168.244.1
AGENT1	AGENT2	Failed	RNIKULA @ 192.168.244.1
AGENT1	AGENT2	Successful	RNIKULA @ 172.29.97.197
AGENT2	AGENT1	Failed	RNIKULA @ 172.29.97.197
AGENT1	AGENT2	Failed	RNIKULA @ 172.29.97.197
AGENT1	AGENT2	Failed	RNIKULA @ 192.168.244.1
AGENT1	AGENT2	Successful	RNIKULA @ 172.29.97.197
AGENT1	AGENT2	Successful	RNIKULA @ 172.29.97.197
AGENT1	AGENT2	Successful	RNIKULA @ 172.29.97.197

# Sample log Entry

```
<?xml version="1.0" encoding="UTF-8"?>
<transaction version="6.00" ID="414d5120434f52442020202020202020a59e525520118c24" agentRole="sourceAgent" xmlns="http://www.ibm.com/xmlns/qm/transaction">
  <action time="2015-05-13T14:08:07.568Z">completed</action>
  <sourceAgent agent="AGNT1A" QMgr="AGNT1" agentType="STANDARD">
    <systemInfo architecture="x86" name="Windows 7" version="6.1 build 7601 Service Pack 1"/>
  </sourceAgent>
  <destinationAgent agent="AGNT2A" QMgr="AGNT2">
  </destinationAgent>
  <originator>
    <hostName>192.168.188.1</hostName>
    <userID>user</userID>
    <mqmdUserID>user</mqmdUserID>
  </originator>
  <status resultCode="49">
    <supplement>BFGELO013E: The transfer with id '414d5120434f52442020202020202020a59e525520118c23' and a reason code of '2087' from WebSphere MQ when sending a message to destination queue 'S...
    </supplement>
  </status>
  <transferSet startTime="2015-05-13T14:08:07.552Z" total="1" bytesSent="0">
  </transferSet>
  <job>
    <name>Job1</name>
  </job>
</transaction>
```

48



# Starting

```
<?xml version="1.0" encoding="UTF-8"?>
<transaction ID="414d51204d4238514d475220202020201d07c9552030d113" agentRole="sourceAgent" ...>
  <action time="2015-08-13T23:59:33.917Z">started</action>
  <sourceAgent QMgr="MB8QMGR" agent="AGENT1" agentType="STANDARD">
    <systemInfo architecture="x86" name="Windows 7" version="6.1 build 7601 Service Pack 1"/>
  </sourceAgent>
  <destinationAgent QMgr="MB8QMGR" agent="AGENT2"/>
  <originator>
    <hostName>192.168.244.1</hostName>
    <userID>RNIKULA</userID>
    <mqmdUserID>RNIKULA</mqmdUserID>
  </originator>
  <transferSet bytesSent="0" startTime="2015-08-13T23:59:33.917Z" total="1">
    <metaDataSet>
      <metaData key="com.ibm.wmqfte.SourceAgent">AGENT1</metaData>
      <metaData key="com.ibm.wmqfte.DestinationAgent">AGENT2</metaData>
      <metaData key="com.ibm.wmqfte.MqmdUser">RNIKULA</metaData>
      <metaData key="com.ibm.wmqfte.OriginatingUser">RNIKULA</metaData>
      <metaData key="com.ibm.wmqfte.OriginatingHost">192.168.244.1</metaData>
      <metaData key="com.ibm.wmqfte.TransferId">414d51204d4238514d475220202020201d07c9552030d113</metaData>
      <metaData key="com.ibm.wmqfte.ScheduleId">10</metaData>
      <metaData key="com.ibm.wmqfte.JobName">AAAAA</metaData>
      <metaData key="com.ibm.wmqfte.Priority">0</metaData>
    </metaDataSet>
  </transferSet>
  <job>
    <name>AAAAA</name>
  </job>
  <scheduleLog ID="10"/>
</transaction>
```

49

# In Progress

50

```
<?xml version="1.0" encoding="UTF-8"?>
<transaction ID="414d51204d4238514d475220202020201d07c9552030d113" agentRole="sourceAgent" ...>
  <action time="2015-08-13T23:59:34.198Z">progress</action>
  <sourceAgent QMgr="MB8QMGR" agent="AGENT1" agentType="STANDARD">
    <systemInfo architecture="x86" name="Windows 7" version="6.1 build 7601 Service Pack 1"/>
  </sourceAgent>
  <destinationAgent QMgr="MB8QMGR" agent="AGENT2" agentType="STANDARD">
    <systemInfo architecture="x86" name="Windows 7" version="6.1 build 7601 Service Pack 1"/>
  </destinationAgent>
  <originator>
    <hostName>192.168.244.1</hostName>
    <userID>RNIKULA</userID>
    <mqmdUserID>RNIKULA</mqmdUserID>
  </originator>
  <transferSet bytesSent="14285" index="0" size="1" startTime="2015-08-13T23:59:33.917Z" total="1">
    <item mode="binary">
      <source disposition="leave" type="file">
        <file last-modified="2014-12-10T14:57:18.000Z" size="14237">C:\TEMP\setup.ini</file>
        <checksum method="MD5">dec47d003341090ce9007ab5eca38623</checksum>
      </source>
      <destination exist="overwrite" type="file">
        <file last-modified="2015-08-13T23:59:34.073Z" size="14237">C:\nastel2\setup.ini</file>
        <checksum method="MD5">dec47d003341090ce9007ab5eca38623</checksum>
      </destination>
      <status resultCode="0"/>
    </item>
  </transferSet>
</job>
  <name>AAAAA</name>
</job>
</transaction>
```

# SUCCESSFUL

```
<?xml version="1.0" encoding="UTF-8"?>
<transaction ID="414d51204d4238514d475220202020201d07c9552030d113" agentRole="sourceAgent" ...>
  <action time="2015-08-13T23:59:34.213Z">completed</action>
  <sourceAgent QMgr="MB8QMGR" agent="AGENT1" agentType="STANDARD">
    <systemInfo architecture="x86" name="Windows 7" version="6.1 build 7601 Service Pack 1"/></sourceAgent>
  <destinationAgent QMgr="MB8QMGR" agent="AGENT2" agentType="STANDARD">
    <systemInfo architecture="x86" name="Windows 7" version="6.1 build 7601 Service Pack 1"/></destinationAgent>
  <originator>
    ...
    <mqmdUserID>RNIKULA</mqmdUserID>
  </originator>
  <status resultCode="0">
    <supplement>BFGRP0032I: The file transfer request has successfully completed.</supplement>
  </status>
  <transferSet bytesSent="14285" startTime="2015-08-13T23:59:33.917Z" total="1">
    <metaDataSet>
      <metaData key="com.ibm.wmqfte.SourceAgent">AGENT1</metaData>
      <metaData key="com.ibm.wmqfte.DestinationAgent">AGENT2</metaData>
      ...
      <metaData key="com.ibm.wmqfte.TransferId">414d51204d4238514d475220202020201d07c9552030d113</metaData>
      <metaData key="com.ibm.wmqfte.ScheduleId">10</metaData>
      <metaData key="com.ibm.wmqfte.JobName">AAAAA</metaData>
      <metaData key="com.ibm.wmqfte.Priority">0</metaData>
    </metaDataSet>
  </transferSet>
  <job>
    <name>AAAAA</name>
  </job>
  <statistics>
    <actualStartTime>2015-08-13T23:59:34.010Z</actualStartTime>
    <retryCount>0</retryCount>
    <numFileFailures>0</numFileFailures>
    <numFileWarnings>0</numFileWarnings>
  </statistics>
</transaction>
```

51

# Failure

```
<?xml version="1.0" encoding="UTF-8"?>
<transaction ID="414d51204d4238514d47522020202020dd843f552004d50c" agentRole="sourceAgent" ...>
  <action time="2015-08-13T23:59:01.121Z">completed</action>
  <sourceAgent QMgr="MB8QMGR" agent="AGENT1" agentType="STANDARD">
    <systemInfo architecture="x86" name="Windows 7" version="6.1 build 7601 Service Pack 1"/>
  </sourceAgent>
  <destinationAgent QMgr="MB8QMGR" agent="AGENT2" agentType="STANDARD">
    <systemInfo architecture="x86" name="Windows 7" version="6.1 build 7601 Service Pack 1"/>
  </destinationAgent>
  <originator>
    <hostName>192.168.244.1</hostName>
    <userID>RNIKULA</userID>
    <mqmdUserID>RNIKULA</mqmdUserID>
  </originator>
  <status resultCode="40">
    <supplement>BFGRP0034I: The file transfer request has completed with no files being transferred.</supplement>
  </status>
  <transferSet bytesSent="0" startTime="2015-08-13T23:58:59.247Z" total="1">
    <metaDataSet>
      <metaData key="com.ibm.wmqfte.SourceAgent">AGENT1</metaData>
      <metaData key="com.ibm.wmqfte.DestinationAgent">AGENT2</metaData>
      <metaData key="com.ibm.wmqfte.MqmdUser">RNIKULA</metaData>
      <metaData key="com.ibm.wmqfte.OriginatingUser">RNIKULA</metaData>
      <metaData key="com.ibm.wmqfte.OriginatingHost">192.168.244.1</metaData>
      <metaData key="com.ibm.wmqfte.TransferId">414d51204d4238514d475220202020dd843f552004d50c</metaData>
      <metaData key="com.ibm.wmqfte.Priority">0</metaData>
    </metaDataSet>
  </transferSet>
  <statistics>
    <actualStartTime>2015-08-13T23:59:00.700Z</actualStartTime>
    <retryCount>0</retryCount>
    <numFileFailures>1</numFileFailures>
    <numFileWarnings>0</numFileWarnings>
  </statistics>
</transaction>
```

52

# Sample Formatted

Admin Overview > Summary > Trace Details

Show: Select From: 2015-05-13 10:08:00 AM To: 2015-05-13 10:09:00 AM Show

Trace

Start Date	Applications	Transaction Status	SLA Status	SLA Status Text	Workload (HH:MM:SS:mm)	Transaction Duration	Operations	Messages	Transaction ID	Transaction Groups	Servers	Man
015-05-13 10:08:08	AGNT1A:AGNT1	Failed ❌		Within SLA 🟢	0:00:00.000	0:00:00.016	2	0	96581		192.168.188.1	Mes
015-05-13 10:08:08	AGNT1A:AGNT1	Failed ❌		Within SLA 🟢	0:00:00.000	0:00:00.000	2	0	96574		192.168.188.1	Mes

Transaction ID 96581

Transaction Flow Diagram Transaction Timeline Transaction Trace(2) Transaction Milestones

Show Hierarchy

Time	Operation Name	Resource	Elapsed Time (usec)	Message Id	Message Age (usec)	Completion Code	Server	Resource Manager Type	Resource Manager	Application	Total Runtime (HH:MM:SS:mm)
015-05-13 10:08:08.552	started	JOB:CHEVREOUX	0		0	Succeeded	192.168.188.1	Messaging Ser...	AGNT1	AGNT1A:AGNT1	0:00:00.000
015-05-13 10:08:08.568	completed	JOB:CHEVREOUX	0		0	Failed ❌	192.168.188.1	Messaging Ser...	AGNT1	AGNT1A:AGNT1	0:00:00.000

53

# IIB (BROKER)

# Type of Broker Management Data Available Remotely

- **Resource Statistics**
  - ▶ Resources used by execution groups
- **Monitoring Statistics**
  - ▶ Usage Statistics of execution groups, nodes and threads
- **Flow Tracking**
  - ▶ Tracking of execution flow through message flows

# Broker Monitoring Statistics

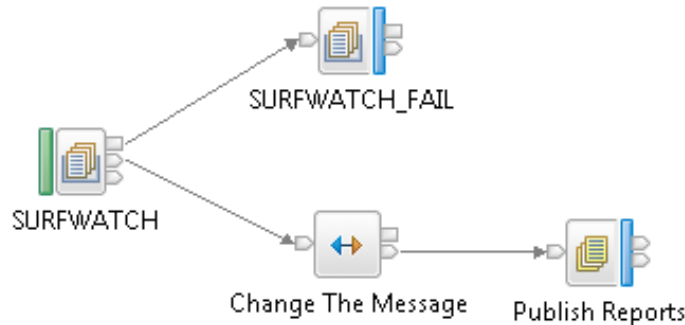
- **The broker provides detailed statistics**
  - **Message Flow Level (for the flow)**
    - Total Messages Processed
    - Total Messages in Error
    - CPU Time Spent
    - Message Statistics
  - **Node Level (for each processing node)**
    - Invocations
    - CPU Time
    - Information
  - **Thread Level (for each thread)**
    - Messages Processed
    - CPU Time Spent
    - Message Statistics



# Tracking within the Message Flows

- **The Broker Supports Tracking within the Message Flows**
  - **Transaction Start / Stop (default)**
  - **See when a given node was processed**
  - **See details about the message being processed by the flow**
  - **Track message flows in and across brokers**
  
- **Activated at the Message Flow and Node Level**
  - **Whether to collect**
  - **Data to Collect**

# Configuring Message Flow Events



Graph User Defined Properties

Properties Problems Deployment Log

## Default Values for Message Flow Properties - SurfWatch

Description

Configure monitoring events.

**Monitoring**

Events

3 events defined. Events are defined via the Monitoring tab of a selected node in the message flow.

Enabled	Node	Event Source	Event Source Address	Event Name
<input checked="" type="checkbox"/>	Publish Reports	In terminal	Publish Reports.terminal.in	Publish Reports.InTerminal
<input checked="" type="checkbox"/>	SURFWATCH	Transaction start	SURFWATCH.transaction.St...	SURFWATCH.TransactionStart
<input checked="" type="checkbox"/>	SURFWATCH_FAIL	In terminal	SURFWATCH_FAIL.terminal....	SURFWATCH_FAIL.InTerminal

# Event Attributes – Basic

**Add event**

Basic | Correlation | Transaction

**Event Source**  
Select the source of the event.  
Transaction end

**Event Source Address**  
The broker identifies an event source using an event source address. Use this value when you enable and disable event sources using runtime commands.  
SURFWATCH.transaction.End

**Event Name**  
Provide the name by which events emitted from this source are to be known. Specify either a literal name, or the location of a character field in the message tree or elsewhere in the message assembly.  
 Literal SURFWATCH.TransactionEnd  
 Data location Edit...

**Event Filter**  
Provide an expression to control whether the event is emitted. The expression must evaluate to true or false, and can reference fields in the message tree or elsewhere in the message assembly. If you do not specify a value, the value true() is used.  
true() Edit...

**Event Payload**  
Most events need to contain data taken from fields in the message tree or from elsewhere in the message assembly. Data taken from simple fields or complex fields appears in the event in XML character format. An event can also contain bitstream data, which appears in the event as hexadecimal bytes.

Data location	

Add...  
Edit...  
Delete

Include bitstream data in payload  
Content Encoding

# Event Attributes – Event Source

The screenshot shows the 'Add event' configuration window with the 'Transaction' tab selected. The 'Event Source' dropdown menu is open, displaying a list of options: Transaction start (highlighted), Transaction end, Transaction rollback, Failure terminal, Out terminal, and Catch terminal. The 'Event Source Address' field contains 'SURFWATCH.transaction.End'. The 'Event Name' section has 'SURFWATCH.TransactionEnd' selected under the 'Literal' radio button. The 'Event Filter' field contains 'true()'. The 'Event Payload' section has an empty table with 'Add...', 'Edit...', and 'Delete' buttons. The 'Include bitstream data in payload' checkbox is unchecked. The 'Content' and 'Encoding' dropdowns are also visible.

**Add event**

Basic Correlation Transaction

**Event Source**  
Select the source of the event.  
Transaction end

**Event Source Address**  
The broker identifies an event source address. You can enable or disable event sources using runtime configuration.  
SURFWATCH.transaction.End

**Event Name**  
Provide the name by which events are emitted, or the location of a character field in the message.  
 Literal SURFWATCH.TransactionEnd  
 Data location Edit...

**Event Filter**  
Provide an expression to control whether the event is emitted. The expression must evaluate to true or false, and can reference fields in the message tree or elsewhere in the message assembly. If you do not specify a value, the value true() is used.  
true() Edit...

**Event Payload**  
Most events need to contain data taken from fields in the message tree or from elsewhere in the message assembly. Data taken from simple fields or complex fields appears in the event in XML character format. An event can also contain bitstream data, which appears in the event as hexadecimal bytes.

Data location	

Add... Edit... Delete

Include bitstream data in payload

Content Encoding

# Event Attributes – Event Name

**Add event**

Basic | Correlation | Transaction

**Event Source**  
Select the source of the event.  
Transaction end

**Event Source Address**  
The broker identifies an event source using an event source address. Use this value when you enable and disable event sources using runtime commands.  
SURFWATCH.transaction.End

**Event Name**  
Provide the name by which events emitted from this source are to be known. Specify either a literal name, or the location of a character field in the message tree or elsewhere in the message assembly.

Literal SURFWATCH.TransactionEnd  
 Data location Edit...

**Event Filter**  
Provide an expression to control whether the event is emitted. The expression must evaluate to true or false, and can reference fields in the message tree or elsewhere in the message assembly. If you do not specify a value, the value true() is used.  
true() Edit...

**Event Payload**  
Most events need to contain data taken from fields in the message tree or from elsewhere in the message assembly. Data taken from simple fields or complex fields appears in the event in XML character format. An event can also contain bitstream data, which appears in the event as hexadecimal bytes.

Data location	

Add...  
Edit...  
Delete

Include bitstream data in payload  
Content Encoding

# Event Attributes – Dynamic Event Name

The screenshot displays the XPATH Expression Builder interface. On the left, there are several configuration sections: 'Event Source' (Transaction start), 'Event Source Address' (SURFWATCH.trans), 'Event Name' (Data location selected), 'Event Filter' (true()), and 'Event Payload'. The main window is titled 'XPath Expression Builder' and contains a 'Data Types Viewer' and an 'Operators' list. The 'Data Types Viewer' shows a tree structure under 'Data Types' with the following items: \$Root, \$Properties, \$LocalEnvironment, \$DestinationList, \$ExceptionList, and \$Environment. The 'Operators' list includes: |, /, <=, <, >=, >, =, !=, and, or, +, and -. Below the operators is a checkbox for 'Show XML Schema groups' and a text field for the 'XPath Expression'. At the bottom, there is a 'Namespace settings' section.

**Event Source**  
Select the source of the event.  
Transaction start

**Event Source Address**  
The broker identifies the event source by address. You can disable event source address.  
SURFWATCH.trans

**Event Name**  
Provide the name of the event, the name, or the location.  
 Literal  
 Data location

**Event Filter**  
Provide an expression that evaluates to true or false, and can refer to event attributes.  
If you do not specify an expression, the filter is true().  
true()

**Event Payload**  
Most events need to be formatted for message assembly. An event can be formatted in one of the following ways:  
Data location

**XPath Expression Builder**  
Select the target from the Schema viewer or Operator viewer and drag and drop the nodes in the source viewer below.

**Data Types Viewer**

- Data Types
  - \$Root
  - \$Properties
  - \$LocalEnvironment
    - \$DestinationList
    - \$ExceptionList
    - \$Environment

**Operators**

- |
- /
- <=
- <
- >=
- >
- =
- !=
- and
- or
- +
- 

Show XML Schema groups

**XPath Expression**

▶ **Namespace settings**

# Event Attributes – Event Filter

**Add event**

Basic | Correlation | Transaction

**Event Source**  
Select the source of the event.  
Transaction end

**Event Source Address**  
The broker identifies an event source using an event source address. Use this value when you enable and disable event sources using runtime commands.  
SURFWATCH.transaction.End

**Event Name**  
Provide the name by which events emitted from this source are to be known. Specify either a literal name, or the location of a character field in the message tree or elsewhere in the message assembly.  
 Literal SURFWATCH.TransactionEnd  
 Data location Edit...

**Event Filter**  
Provide an expression to control whether the event is emitted. The expression must evaluate to true or false, and can reference fields in the message tree or elsewhere in the message assembly. If you do not specify a value, the value true() is used.  
true() Edit...

**Event Payload**  
Most events need to contain data taken from fields in the message tree or from elsewhere in the message assembly. Data taken from simple fields or complex fields appears in the event in XML character format. An event can also contain bitstream data, which appears in the event as binary data.  

Data location		

Add... Edit... Delete  
 Include bitstream data in payload  
Content Encoding

# Event Attributes – Event Payload

The screenshot shows a 'Add event' dialog box with several sections. The 'Event Payload' section at the bottom is highlighted with a green rounded rectangle. It contains a table for adding data locations, a checkbox for 'Include bitstream data in payload', and dropdown menus for 'Content' and 'Encoding'.

**Add event**

Basic | Correlation | Transaction

**Event Source**  
Select the source of the event.  
Transaction end

**Event Source Address**  
The broker identifies an event source using an event source address. Use this value when you enable and disable event sources using runtime commands.  
SURFWATCH.transaction.End

**Event Name**  
Provide the name by which events emitted from this source are to be known. Specify either a literal name, or the location of a character field in the message tree or elsewhere in the message assembly.  
 Literal SURFWATCH.TransactionEnd  
 Data location Edit...

**Event Filter**  
Provide an expression to control whether the event is emitted. The expression must evaluate to true or false, and can reference fields in the message tree or elsewhere in the message assembly. If you do not specify a value, the value true() is used.  
true() Edit...

**Event Payload**  
Most events need to contain data taken from fields in the message tree or from elsewhere in the message assembly. Data taken from simple fields or complex fields appears in the event in XML character format. An event can also contain bitstream data, which appears in the event as hexadecimal bytes.

Data location	

Add...  
Edit...  
Delete

Include bitstream data in payload

Content: [ ] Encoding: [ ]



# Event Attributes - Correlation

The screenshot shows a software window titled "Edit event" with a close button in the top right corner. Below the title bar are three tabs: "Basic", "Correlation", and "Transaction". The "Correlation" tab is selected and active. The main content area is titled "Event Correlation" and contains a descriptive paragraph about event correlators. Below this are three sections, each with a radio button selection and a description box:

**Event Correlation**  
A monitoring application uses event correlators to match events emitted by the same, or related, business transactions. A local transaction correlator links the events emitted by a single invocation of a message flow. A parent transaction correlator links the events from a message flow to a parent message flow or an external application. A global transaction correlator links events from a message flow to one or more related message flows or external applications. An event must contain a local transaction correlator, but need not contain a parent transaction correlator or global transaction correlator.

Local transaction correlator:  
 Automatic  Specify location of correlator

Description  
The local correlator used by the most recent event for this message flow invocation will be used. If no local correlator exists yet, a new unique value will be generated.

Parent transaction correlator:  
 Automatic  Specify location of correlator

Description  
The parent correlator used by the most recent event for this message flow invocation will be used. If no correlator exists yet, no parent correlator will be used.

Global transaction correlator:  
 Automatic  Specify location of correlator

Description  
The global correlator used by the most recent event for this message flow invocation will be used. If no correlator exists yet, no global correlator will be used.

# mqsichangeflowmonitoring Examples

- `mqsichangeflowmonitoring BROKER -c active -g -j`
- > Activate event flow monitoring for all execution groups and flows
  
- `mqsichangeflowmonitoring BROKER -c active -e default -k application1 -f myflow1`
- > Activate monitoring for message flow *myflow1* referenced by application *application1*, in execution group *default*
  
- `mqsichangeflowmonitoring BROKER -c inactive -g -j`
- > Deactivate event flow monitoring for everything

# Getting Tracking Data

- The statistics Tracking data is published
- `$SYS/Broker/<brokerName>/Monitoring/<executionGroupName>/<flowName>`
- **Example Subscriptions**
  - `$SYS/Broker/Broker1/Monitoring/#`
  - `$SYS/Broker+/Monitoring/#`
  - `$SYS/Broker/Broker2/Monitoring/EGRP/Flow1`

The resultant data is then processed directly by a subscribing application or placed on a queue for processing by an application.

# Tracking Data

```
<wmb:event xmlns:wmb="http://www.ibm.com/xmlns/prod/websphere/messagebroker/6.1.0/monitor/event">
  <wmb:eventPointData>
    <wmb:eventData wmb:productVersion="8001" wmb:eventSchemaVersion="6.1.0.2">
      <wmb:eventIdentity wmb:eventName="transactionStart" />
      <wmb:eventSequence wmb:creationTime="2015-09-25T21:06:10.008Z" wmb:counter="1" />
      <wmb:eventCorrelation
        wmb:localTransactionId="414d51204d4238514d47522020202020bf172454201558fe"
        wmb:parentTransactionId="" wmb:globalTransactionId="" />
      </wmb:eventData>
    <wmb:messageFlowData>
      <wmb:broker wmb:name="MB8BROKER" wmb:UUID="61f8eda0-81f5-43b6-8cf5-b9a1fef8f91b" />
      <wmb:executionGroup wmb:name="PagerExecutionGroup"
        wmb:UUID="a4f0fff6-4501-0000-0080-c644e460ccff" />
      <wmb:messageFlow
        wmb:uniqueFlowName="MB8BROKER.PagerExecutionGroup.SurfWatch"
        wmb:name="SurfWatch" wmb:UUID="6c0000f7-4501-0000-0080-d6b3e1d5c115"
        wmb:threadId="10044" />
      <wmb:node wmb:nodeLabel="SURFWATCH" wmb:nodeType="ComIbmMQInputNode"
        wmb:detail="SURFWATCH" />
      </wmb:messageFlowData>
    </wmb:eventPointData>
  </wmb:event>
```

# Tracking Data

```
<wmb:event
xmlns:wmb="http://www.ibm.com/xmlns/prod/websphere/messagebroker/6.1.0/monitoring/event">
<wmb:eventPointData>
  <wmb:eventData wmb:productVersion="8001" wmb:eventSchemaVersion="6.1.0.2">
    <wmb:eventIdentity wmb:eventName="transactionEnd" />
    <wmb:eventSequence wmb:creationTime="2015-09-25T21:06:48.273998Z"
wmb:counter="2" />
    <wmb:eventCorrelation wmb:localTransactionId="
414d51204d4238514d47522020202020bf172454201558fe "
      wmb:parentTransactionId="" wmb:globalTransactionId="" />
    </wmb:eventData>
    <wmb:messageFlowData>
      <wmb:broker wmb:name="MB8BROKER" wmb:UUID="61f8eda0-81f5-43b6-8cf5-
b9a1fef8f91b" />
      <wmb:executionGroup wmb:name="PagerExecutionGroup"
        wmb:UUID="a4f0fff6-4501-0000-0080-c644e460ccff" />
      <wmb:messageFlow
wmb:uniqueFlowName="MB8BROKER.PagerExecutionGroup.SurfWatch"
        wmb:name="SurfWatch" wmb:UUID="6c0000f7-4501-0000-0080-d6b3e1d5c115"
        wmb:threadId="10044" />
      <wmb:node wmb:nodeLabel="SURFWATCH" wmb:nodeType="ComIbmMQInputNode"
        wmb:detail="SURFWATCH" />
    </wmb:messageFlowData>
  </wmb:eventPointData>
</wmb:event>
```

# Example

- Message Flows can be tracked by capturing the flow tracking events



Transaction Group > Summary > Trace Details

Show:  From: 2014-02-17 10:45:00 AM To: 2014-02-17 10:46:00 AM

Trace

Start Date	Applications	Transaction Status	SLA Status	SLA Status Text	Workload (HH:MM:SS.mm)	Transaction Duration	Operations	Messages	Transaction ID	Transaction Groups	Servers	Resource Manager Typ
2014-02-17 10:45:20	MB8BROKER.P...	Complete ✓	Within SLA	Within SLA	0:00:00.000	0:00:00.025	2	1 3			DESKTOP99	Messaging S
2014-02-17 10:45:35	MB8BROKER.P...	Complete ✓	Within SLA	Within SLA	0:00:00.000	0:00:00.015	2	1 2			DESKTOP99	Messaging S
2014-02-17 10:45:17	MB8BROKER.P...	Complete ✓	Within SLA	Within SLA	0:00:00.000	0:00:00.008	2	1 1			DESKTOP99	Messaging S

Transaction ID 3

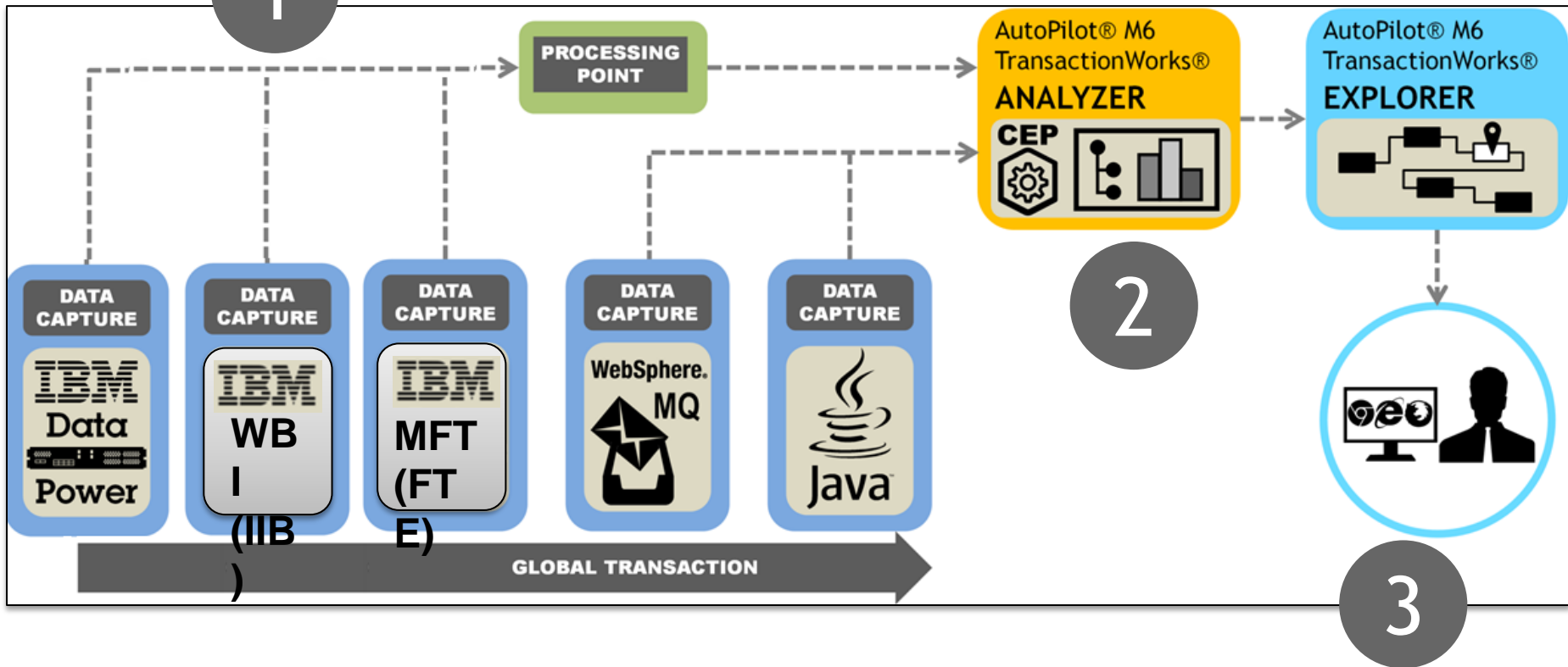
Transaction Flow Diagram | Transaction Timeline | Transaction Trace(2) | Transaction Milestones

Show Hierarchy

Time	Application	Operation Name	Resource	Message Id	Message Age (usec)	Completion Code	Correlator	Sen
2014-02-17 10:45:20.214	MB8BROKER.PagerExecutionGroup.SurfWatch	transactionStart	SURFWATCH	1	0	Succeeded	414d51204d4238514d47522020202035b8fb5220687f1d	DESKTOP
2014-02-17 10:45:20.239	MB8BROKER.PagerExecutionGroup.SurfWatch	transactionEnd	SURFWATCH		0	Succeeded	414d51204d4238514d47522020202035b8fb5220687f1d	DESKTOP

# SUMMARY

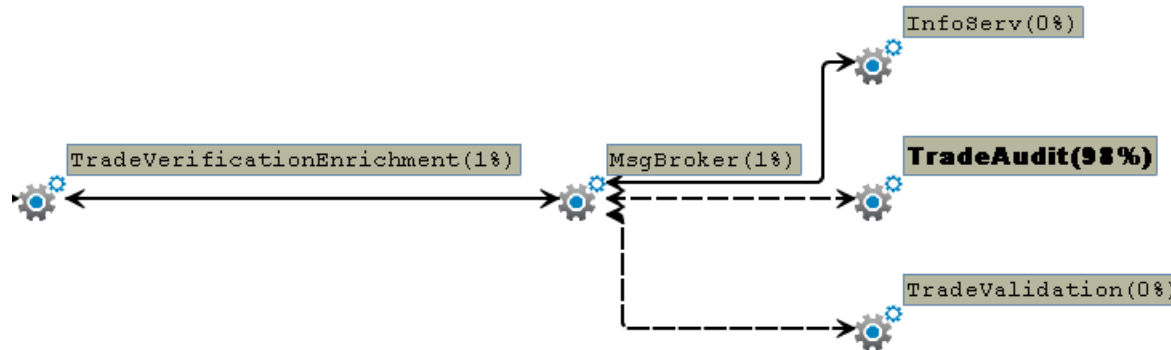
# Other Transaction Monitoring





# Cross Application Tracking

73



Time	Operation Name	Resource	Elapsed Time (usec)	Message Id	Message Age (usec)	Completion Code
2013-03-26 13:41:47.17	/trading/verification	/trading/verificati	7875		0	Succeeded L
2013-03-26 13:41:47.19	com/acme/trading/verify.start	java	5749		0	Succeeded L
2013-03-26 13:41:47.19	HTTP/GET/trading/verification	/trading/verificati	7665	2195	0	Succeeded L
2013-03-26 13:41:47.19	Statement.executeQuery(String)	MySQL5.1:DBSeri	1823	2196	0	Succeeded L
2013-03-26 13:41:47.20	com/acme/trading/verify.request	java	1495		0	Succeeded L
2013-03-26 13:41:47.20	QueueSender.send	queue://trading/:	6565	2186	0	Succeeded L
2013-03-26 13:41:47.20	MQPUT	TradeVerification.	10107	2186	0	Succeeded L
2013-03-26 13:41:47.22	MQGET	TradeVerification.	10908	2186	834	Succeeded B
2013-03-26 13:41:47.22	MQPUT	TradeValidation.I	9399	2187	0	Succeeded B
2013-03-26 13:41:47.24	MQGET	TradeValidation.I	9507	2187	116	Succeeded W
2013-03-26 13:41:47.24	MQPUT	TradeValidation.C	9803	2188	0	Succeeded W
2013-03-26 13:42:23.22	MQGET	TradeValidation.C	35970000	2188	652	Succeeded B
2013-03-26 13:42:23.22	MQPUT	TradeAudit.Input.	10511	2189	0	Succeeded B

# Additional Information

- For additional information on products provided by Nastel including AutoPilot services for IBM MQ, Integration Bus, Managed File Transfer, DataPower and WebSphere Application Server

74

- See <http://www.nastel.com>
- Or contact [info@nastel.com](mailto:info@nastel.com)
- Thank You

# *Questions*

