#SHAREorg

SHARE

Internals of IBM Integration Bus

Dave Gorman
IBM

13th March 2014
15025

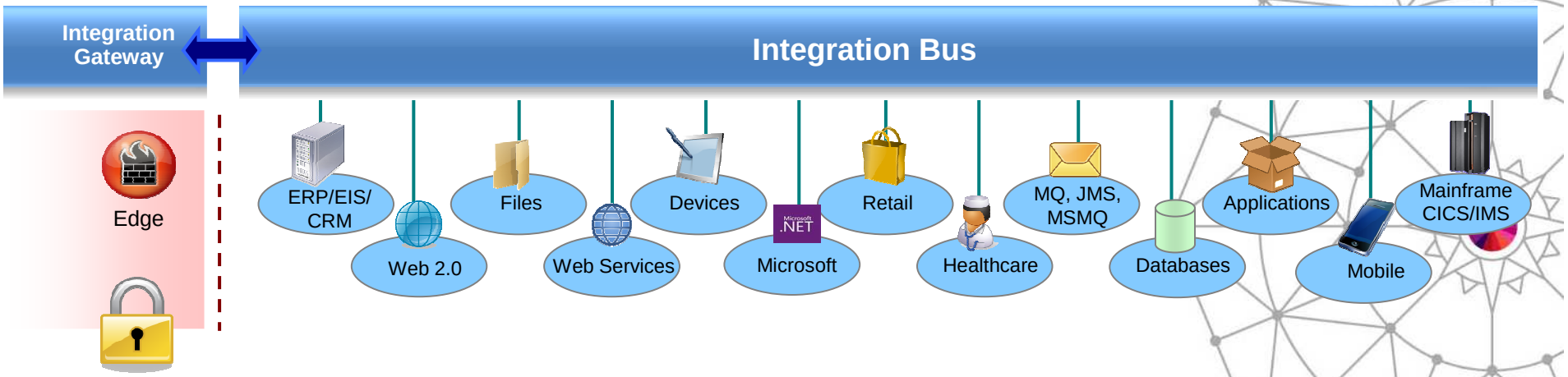
SHARE in Anaheim

Agenda

- Introduction
- Runtime Processes
- Threads
- Memory
- Diagnostic Information

Introducing IBM Integration Bus

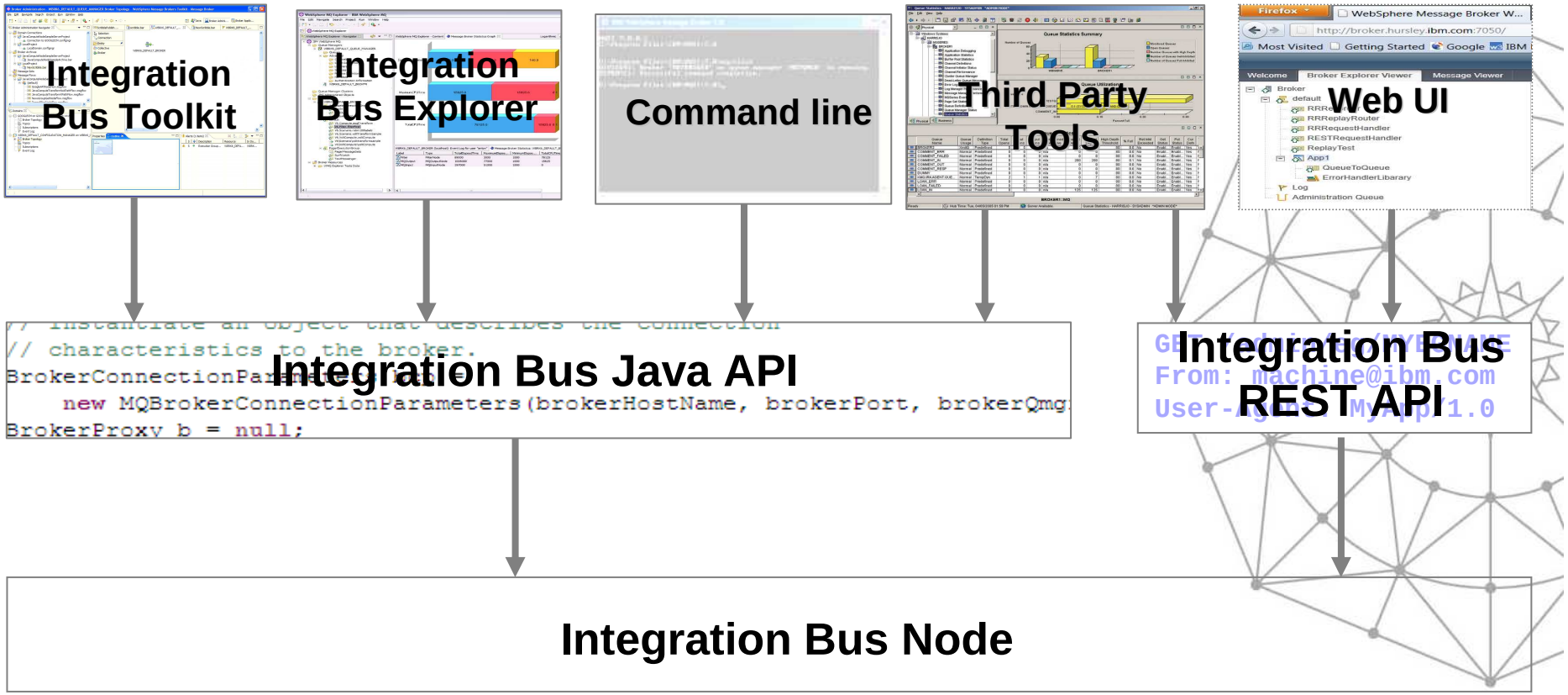
- **IBM's Strategic Integration Technology**
  - Single engineered product for .NET, Java and fully heterogeneous integration scenarios
  - DataPower continues to evolve as IBM's integration gateway



- **A Natural Evolution for WebSphere Message Broker users**
  - Significant innovation and evolution of WMB technology base
  - New features for Policy-based WLM, BPM integration, Business rules and .NET

- **Designed to incorporate WebSphere Enterprise Service Bus use cases**
  - Capabilities for WESB are folded in to IBM Integration Bus over time
  - Conversion tools for initial use cases built in to IIB from day one
  - WESB technology remains in market, supported. Migrate to Integration Bus when ready

Integration Bus Components



Runtime Processes

Processes

- **bipimain**
  - z/OS only. The first process in any IIB address space. APF authorised to set up authorised routines.
- **bipservice**
  - Lightweight and resilient process that starts and monitors the bipbroker process (a.k.a AdminAgent).
  - If the bipbroker process fails, bipservice will restart it.
- **bipbroker**
  - A more substantial process. Contains the deployment manager, for CMP and REST connections, as well as the WebUI server. All commands, toolkit connections and WebUI go through this process.
  - Responsible for starting and monitoring the biphttplistener and DataFlowEngine processes.
  - If either process fails, bipbroker will restart them.
- **biphttplistener**
  - Runs the brokerwide HTTP connector for HTTP and SOAP nodes.
- **DataFlowEngine** (a.k.a Integration Server)
  - Runtime engine for all deployed resources.

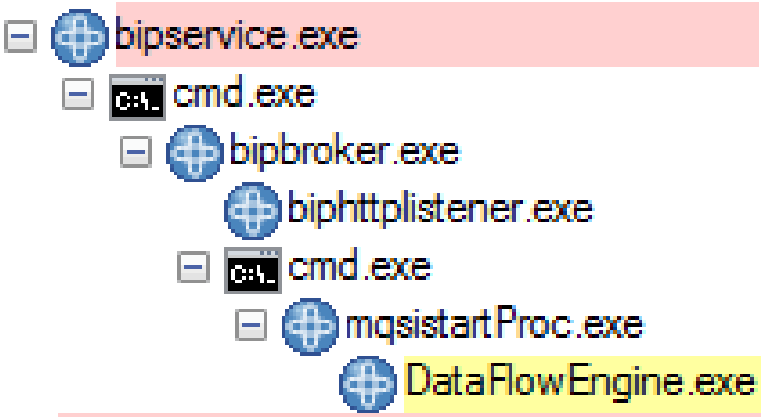
Processes by platform

- Linux and Unix systems

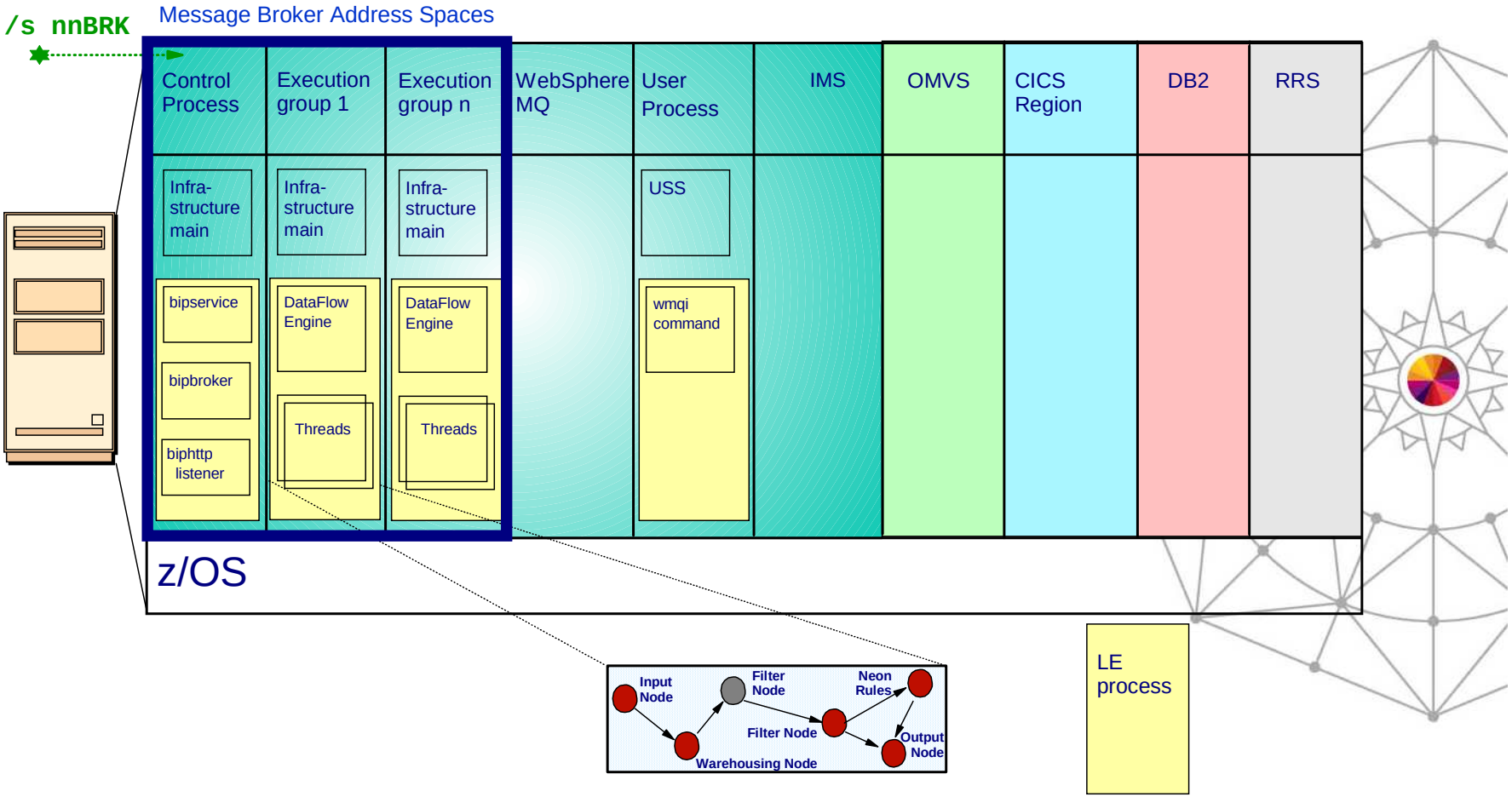
```
mqm      25475     1  0 15:56 ?        00:00:00 bipservice CSIM
mqm      25480 25475  0 15:56 ?        00:00:26 bipbroker CSIM
mqm      25556 25480  0 15:56 ?        00:00:01 biphttplistener CSIM

mqm        722 25480  0 16:28 ?        00:00:00 /bin/bash /opt/mqsi/bin/startDataFlowEngine
mqm        723   722  0 16:28 ?        00:00:10 DataFlowEngine CSIM d8ffc588-4401-0000-0080
```

- Windows



Processes grouped by Address Space on z/OS



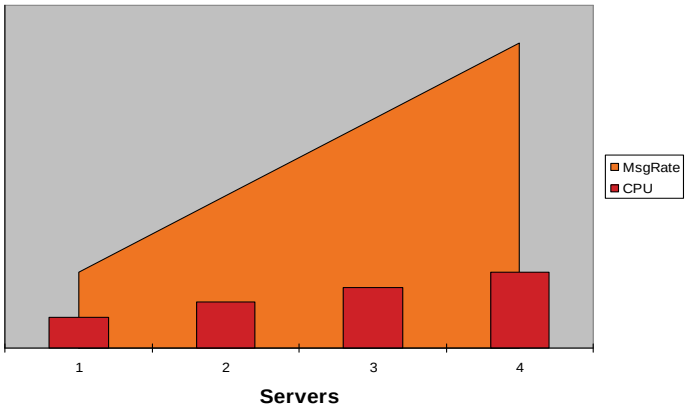
Displaying broker at the process level on z/OS

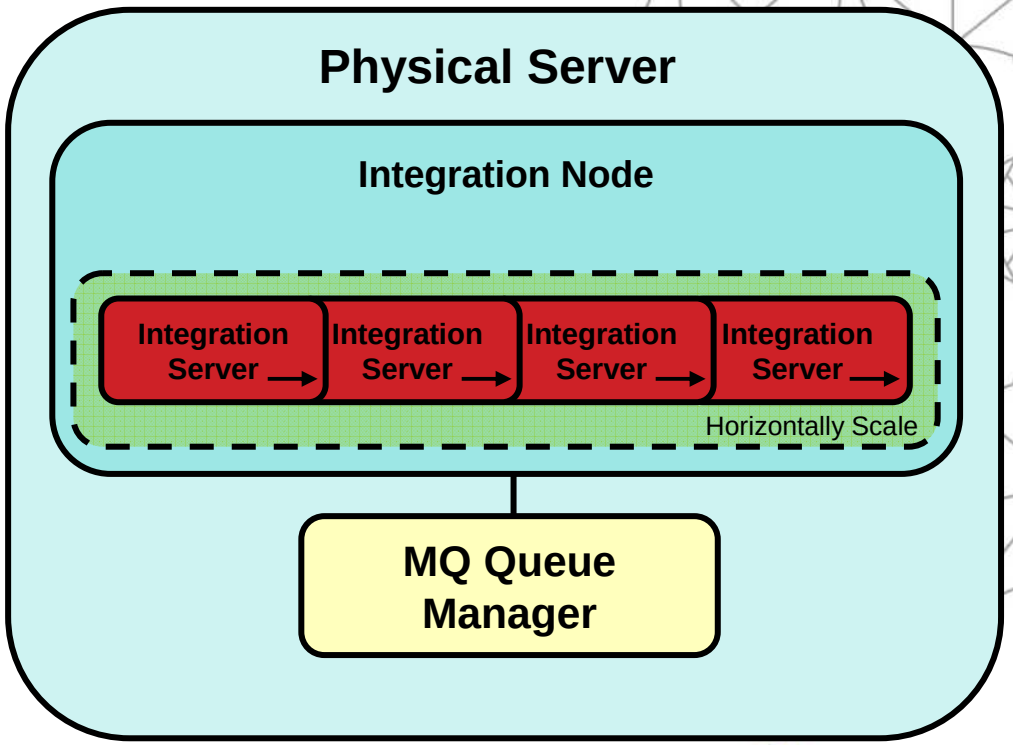
```
SDSF PROCESS DISPLAY  MVD1     ALL                          LINE 1-11 (11)
COMMAND INPUT ===> _                                          SCROLL ===> CSR
PREFIX=MQ05BRK  DEST=(ALL)  OWNER=*  SORT=ASID/A  SYSNAME=*
NP   JOBNAME  JobID    Status                          Owner    Command
     MQ05BRK  STC52908 FILE SYS KERNEL WAIT            MQ05BRK  BPXBATA8
     MQ05BRK  STC52908 RUNNING                         MQ05BRK  biphttplistener MQ05BRK
     MQ05BRK  STC52908 RUNNING                         MQ05BRK  bipservice MQ05BRK AUTO
     MQ05BRK  STC52908 WAITING FOR CHILD               MQ05BRK  /argoinst/DAVE/usr/lpp/mqsi/
     MQ05BRK  STC52908 RUNNING                         MQ05BRK  bipbroker MQ05BRK
     MQ05BRK  STC52910 RUNNING                         MQ05BRK  DataFlowEngine MQ05BRK 4faaf
     MQ05BRK  STC52910 WAITING FOR CHILD               MQ05BRK  /argoinst/DAVE/usr/lpp/mqsi/
     MQ05BRK  STC52910 FILE SYS KERNEL WAIT            MQ05BRK  BPXBATA8
     MQ05BRK  STC52909 RUNNING                         MQ05BRK  DataFlowEngine MQ05BRK cc32d
     MQ05BRK  STC52909 FILE SYS KERNEL WAIT            MQ05BRK  BPXBATA8
     MQ05BRK  STC52909 WAITING FOR CHILD               MQ05BRK  /argoinst/DAVE/usr/lpp/mqsi/
```

```
/u/gormand:>ps -ef | grep MQ05
 MQ05BRK   84017209   50463369 /argoinst/DAVE/usr/lpp/mqsi/bin/bipimain bipservice MQ05BRK AUTO
 MQ05BRK   84017284   84017835 DataFlowEngine MQ05BRK cc32dfb6-3001-0000-0080-a253e63dfe32 DAVE
 MQ05BRK   67240352   84017924 biphttplistener MQ05BRK
 MQ05BRK   67240502   84017209 bipservice MQ05BRK AUTO
 MQ05BRK   84017835   84017546 /argoinst/DAVE/usr/lpp/mqsi/bin/bipimain DataFlowEngine 00071016
     DFS     131792     131828 grep MQ05
 MQ05BRK   84017924   67240502 bipbroker MQ05BRK
```

Horizontal Scaling with additional processes

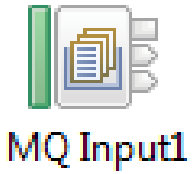
- IBM Integration Bus supports horizontal scaling. This is achieved by increasing the number of integration servers the service is running in.
- Reasons to add additional integration servers:
  - Increased throughput
  - Operational simplicity
  - Workload isolation
  - Better H/W utilisation
  - Higher Availability





Threads

Integration Flow threads (instances)

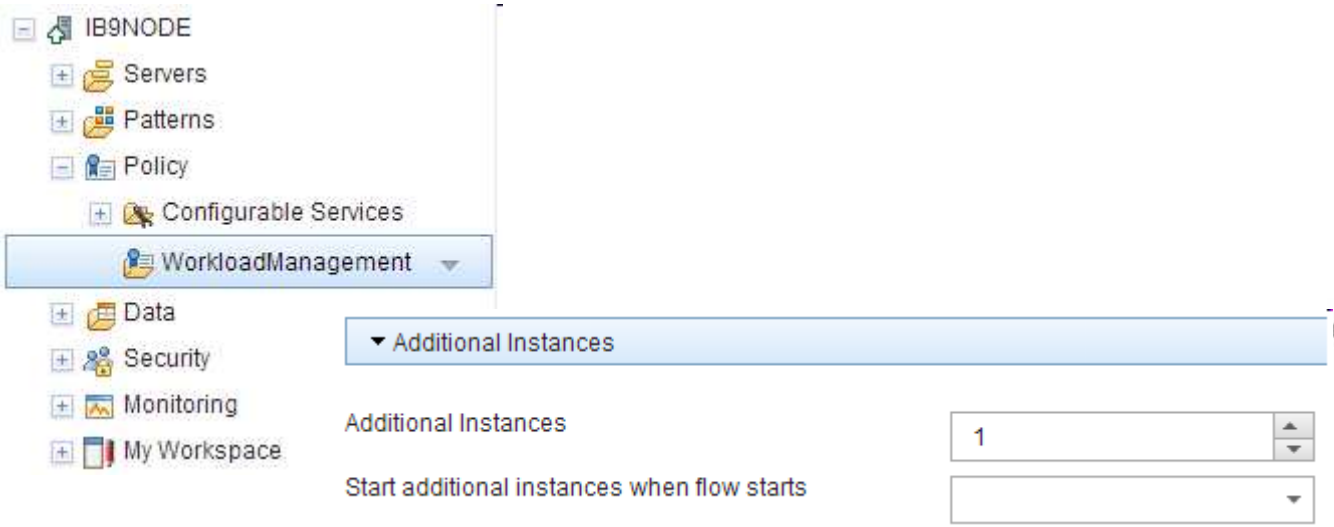


- Every flow will have at least 1 thread per input node.
  - On z/OS, these threads are TCBs.
- Held within a pool for each input node.
- Increase threads by adding “additional instances”.
- The integration server will start additional instances on demand, as the amount of incoming work increases, up to the limit specified.
- If they are idle then additional instances are stopped.
- Additional instances <u>can</u> be started immediately when the integration server starts.

- Be aware of adding additional instance at the flow level when there are multiple input nodes in the same flow!

Setting additional instances

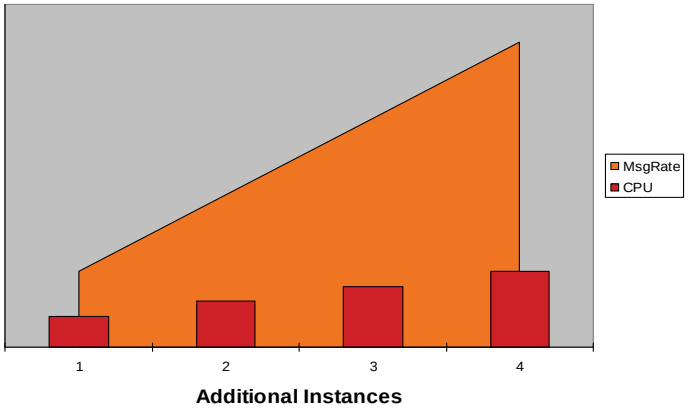
- In IIB V9, additional instances can be set by creating a workload management policy in the WebUI

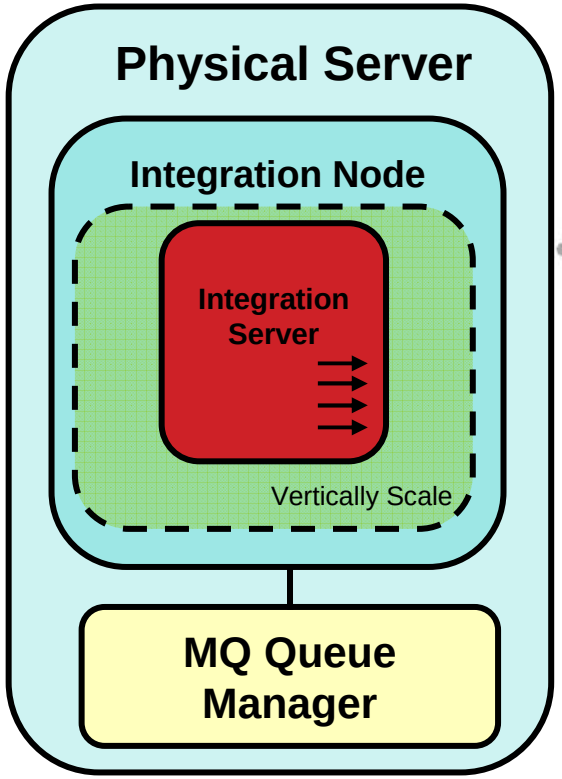


- Additional instances can also be defined on the input node, as a BAR file override and from IIB Explorer.

Vertical Scaling with additional threads

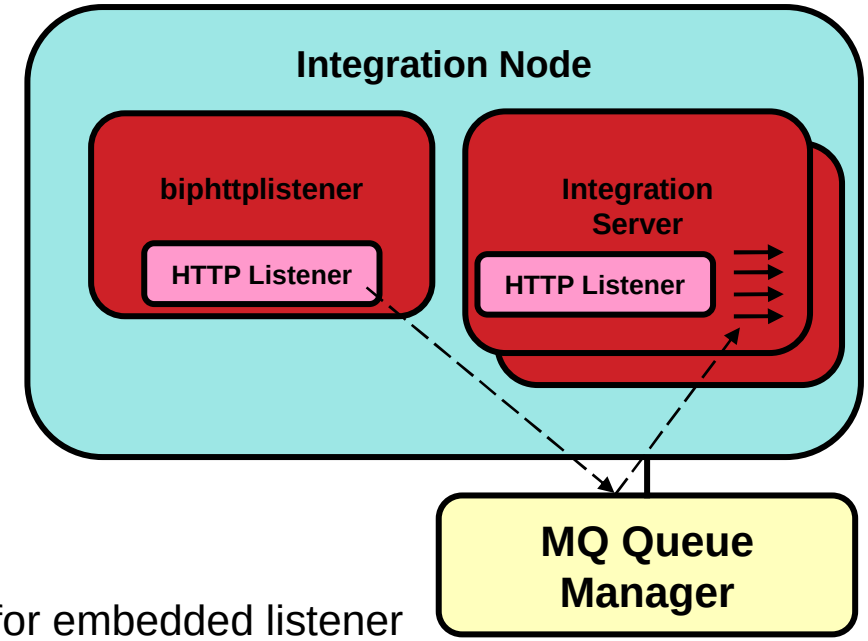
- IBM Integration Bus supports vertical scaling. This is achieved by increasing additional instances (threads) of the service running within the integration server.
- Reasons to add additional instances:
  - Increased throughput
  - Lower memory requirement
  - Better H/W utilisation
  - Policy management





Other thread pools

- There are other thread pools for the HTTP connectors.



- HTTP Connector for embedded listener
  - mqsireportproperties IB9NODE -e default -o HTTPConnector -a
- HTTP Connector for broker wide listener
  - mqsireportproperties IB9NODE -b httplistener -o HTTPConnector -a

HTTP Connector parameters

```
HTTPConnector
  uuid='HTTPConnector'
  address=''
  port='7080'
  maxPostSize=''
  acceptCount=''
  compressableMimeTypes=''
  compression=''
  connectionLinger=''
  connectionTimeout=''
  maxHttpHeaderSize=''
  maxKeepAliveRequests=''
  maxThreads=''
  minSpareThreads=''
  noCompressionUserAgents=''
  restrictedUserAgents=''
  socketBuffer=''
  tcpNoDelay=''
  enableLookups='false'
```

- Check the IIB Infocenter for a list of all the HTTP Connector parameters, and their default values.

-n maxThreads

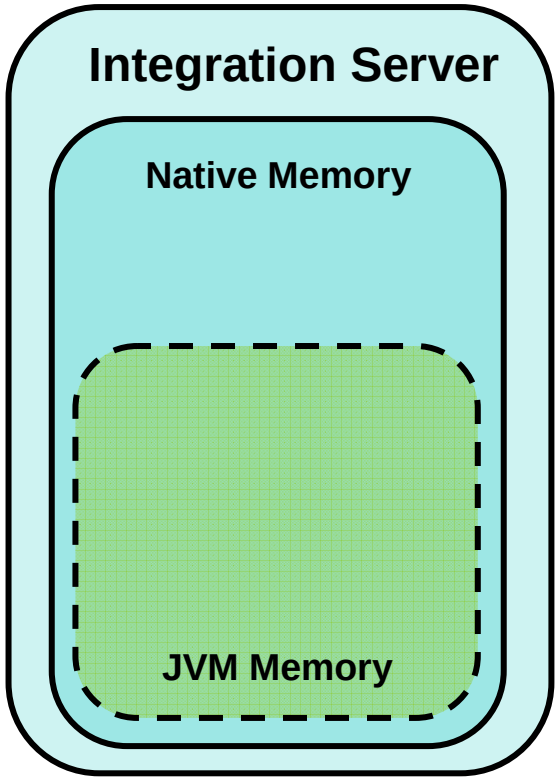
Set the value to the maximum number of threads that can be created by the HTTPConnector.

- Value type - integer
- Initial value - 200

Memory

Memory use within an Integration Server

- The Integration Server runtime comprises both C++ and Java. This means there is a multi faceted approach to checking memory.



- The Integration Server is a process (DataFlowEngine). Its size is limited by available memory and configuration on the operating system.

- There is 1 JVM within each Integration Server. The JVM has preconfigured minimum and maximum HEAP settings.

What are the main consumers of memory?

- Normal (low) consumers of memory include:
  - Deployment artefacts
  - Configuration, such as increasing the min JVM HEAP
  - Message flows
  - Additional instance
  - …
- What to watch for:
  - Size of input messages (1K or 1GB)
  - How messages are parsed (whole file or records)
  - Message Tree copying (transformation nodes)
  - Custom code (ESQL, Java etc)
  - Caching (Shared Variable, GlobalCache etc)
  - JVM settings

See Parser Stats later!

How to check process memory usage

- Use operating system tools to view the DataFlowEngine process size
  - **Windows. Process Explorer**

Process	PID	C...	Private Bytes
DataFlowEngine.exe	13196	0.02	265,524 K

  - **AIX**

```
-bash-3.00$ ps -e -o vsz=,rss=,comm= | grep Data
190456 190496 DataFlowEngine
```

  - **Unix/Linux**

```
-bash-3.2$ ps -e -o vsz,rss,cmd | grep DataFlowEngine
 57984  1400 /bin/bash /opt/mqsi/bin/startDataFlowEngine CSIM bba82c62-7344-44b3-9568-1790fdab483b default
1137476 200208 DataFlowEngine CSIM bba82c62-7344-44b3-9568-1790fdab483b default
```

  - **z/OS**

NP	JOBNAME	StepName	ProcStep	JobID	Owner	SrvClass	RptClass	SysName	C	Pos	DP	Real	Paging	SIO	CPU%	ASID	ASIDX	EX
	MQ05BRK	LMEXPERT	BROKER	STC63060	MQ05BRK	STCFAST	COLIN	MVD1		IN	F4	55,031	0.00	3.62	0.03	205	00CD	
	MQ05BRK	MQ05BRK	BROKER	STC63059	MQ05BRK	STCUSER	MQ05	MVD1		IN	EC	30,901	0.00	0.00	0.05	208	00D0	
	MQ05BRK	DAVE	BROKER	STC63061	MQ05BRK	STCUSER	DQ05BASE	MVD1		IN	EC	56,046	0.00	3.62	0.03	240	00F0	

  - vsz = The size in kilobytes of the core image of the process
  - rss = Indicates the real memory (resident set) size of the process (in 1 KB units)
  - Real = 4k pages

How to check JVM memory usage

1. Check defaults:
   - mqsireportproperties IB9NODE -e default -o ComIbmJVMManager –a

```
jvmMinHeapSize='-1'
jvmMaxHeapSize='-1'
```

   - **jvmMinHeapSize**
     - Initial value: -1, which represents 33554432 bytes (32MB) with the global cache disabled, or 100663296 (96MB) with the global cache enabled
   - **jvmMaxHeapSize**
     - Initial value: -1, which represents 268435456 bytes (256 MB)

2. Check usage with Resource Statistics

default Resources Statistics (Snapshot time 12:19:47 - 12:20:07)

DotNet App Domains | CICS | DotNet GC | CORBA | ConnectDirect | DecisionServices | FTEAgent | FTP | File | GlobalCache | JDBCConnectionPools | JMS | JVM | ODBC | Parsers | SOAPInput | Sec

name	InitialMemoryInMB	UsedMemoryInMB	CommittedMemoryInMB	MaxMemoryInMB	CumulativeGCTimeInSeconds	CumulativeNumberOfGCCollections
summary	32	89	117	-1	4	102
Heap Memory	32	12	36	256		
Non-Heap Memory	0	77	81	-1		
Garbage Collection - Copy					0	44
Garbage Collection - MarkSweepCompact					4	58

Diagnostic Information

Diagnostic Information in WMB

- Diagnostic Information
  - Resource Statistics
    - Parser Statistics
  - Flow Statistics
  - Activity Log
  - Administration Log
  - System Log
  - Trace
  - Stdout/Stderr

Resource Statistics

- **Graphical Performance Monitor**
  - Reports comprehensive usage according of well known resources
    - Message Flows, Nodes, JVM, HTTP, SOAP/HTTP sockets etc
    - Optionally partitioned by Broker, Execution Group and Message Flow
- **Reporting Mechanisms**
  - Graphically reported through IIB Explorer
    - Sort, filter and chart performance characteristics
    - View CPU, IO and other metrics
    - Log data to file in CSV/Excel readable format for post processing
  - User Configurable Reporting Interval
    - XML report messages consumed by any end user application
- **Examples of Available Resource Report Metrics**
  - **JVM:** Memory used, thread count, heap statistics…
  - **Sockets:** Socket host/port open; bytes sent, bytes received

default Resources Statistics (Snapshot time 12:19:47 - 12:20:07)

DotNet App Domains | CICS | DotNet GC | CORBA | ConnectDirect | DecisionServices | FTEAgent | FTP | File | GlobalCache | JDBCConnectionPools | JMS | JVM | ODBC | Parsers | SOAPInput | Sec

name	InitialMemoryInMB	UsedMemoryInMB	CommittedMemoryInMB	MaxMemoryInMB	CumulativeGCTimeInSeconds	CumulativeNumberOfGCCollections
summary	32	89	117	-1	4	102
Heap Memory	32	12	36	256		
Non-Heap Memory	0	77	81	-1		
Garbage Collection - Copy					0	44
Garbage Collection - MarkSweepCompact					4	58

Parser Statistics

- Use Parser statistics to understand memory costs associated with processing messages.
  - This is a simple XML file, parsed and serialised using **XMLNSC**.
  - The actual size of the file is **118** bytes (matches **0.12KB** reported).
  - Not all fields have been parsed, as the flow has parse on demand.

```
<Customer>
  <FirstName>Joe</FirstName>
  <LastName>Bloggs</LastName>
  <ID>1234567890123456789</ID>
</Customer>
```

```
CREATE FUNCTION Main() RETURNS BOOLEAN
BEGIN
    CALL CopyEntireMessage();
    RETURN TRUE;
END;
```

DotNet App Domains | CICS | DotNet GC | CORBA | ConnectDirect | DecisionServices | FTEAgent | FTP | File | GlobalCache | JDBCConnectionPools

ExecutionGroup	name	Threads	ApproxMemKB	MaxReadKB	MaxWrittenKB	Fields	Reads	FailedReads	Writes	FailedWrites
default	summary	1	111.78	0.47	0.47	17	7	0	4	0
default	ParserStats.MQMD	1	15.97	0.36	0.43	2	2	0	1	0
default	ParserStats.MQROOT	1	55.89	0.47	0.00	7	1	0	1	0
default	ParserStats.Properties	1	23.95	0.00	0.00	6	2	0	1	0
default	ParserStats.XMLNSC	1	15.97	0.12	0.47	2	2	0	1	0
default	[Deleted]	0	0.00	0.00	0.00	0	0	0	0	0
default	[Administration]	2	63.88	0.52	0.00	73	4	0	0	0

More Parser Statistics

- The same input message as before, but different ESQL. This time the ESQL statement refers to the last element (Customer.ID).
- The parser statistics show how many more fields are read.

```
<Customer>
  <FirstName>Joe</FirstName>
  <LastName>Bloggs</LastName>
  <ID>1234567890123456789</ID>
</Customer>
```

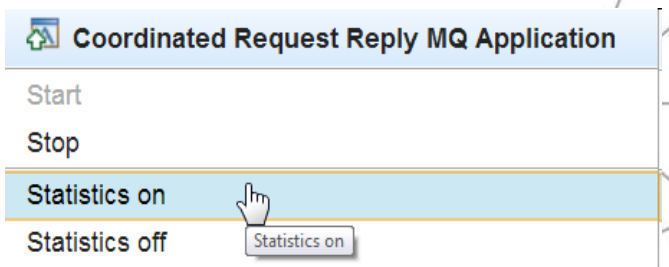
```
CREATE FUNCTION Main() RETURNS BOOLEAN
BEGIN
    --CALL CopyEntireMessage();
    SET OutputRoot.XMLNSC.ID = InputRoot.XMLNSC.Customer.ID;
    RETURN TRUE;
END;
```

DotNet App Domains | CICS | DotNet GC | CORBA | ConnectDirect | DecisionServices | FTEAgent | FTP | File | GlobalCache | JDBCConnectionPools | JMS | JVM | ODBC | Parsers

name	Threads	ApproxMemKB	MaxReadKB	MaxWrittenKB	Fields	Reads	FailedReads	Writes	FailedWrites
summary	1	95.81	0.47	0.03	24	4	0	3	0
ParserStats.MQMD	1	7.98	0.36	0.00	1	1	0	0	0
ParserStats.MQROOT	1	55.89	0.47	0.00	7	1	0	1	0
ParserStats.Properties	1	15.97	0.00	0.00	9	1	0	1	0
ParserStats.XMLNSC	1	15.97	0.12	0.03	7	1	0	1	0
[Deleted]	0	0.00	0.00	0.00	0	0	0	0	0
[Administration]	2	71.86	4.88	0.00	144	8	0	0	0

Flow Statistics

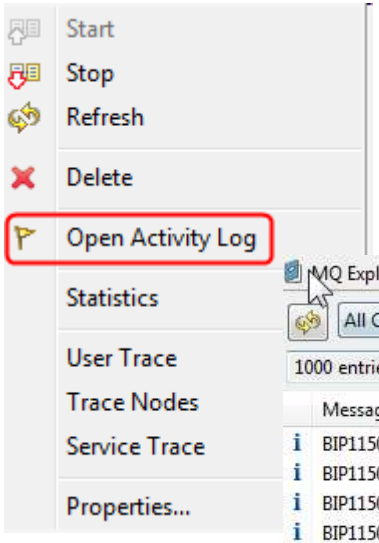
- Using the WebUI in Integration Bus v9:
  - Control statistics at all levels
  - Easily view and compare flows, helping to understand which are processing the most messages or have the highest elapsed time
  - Easily view and compare nodes, helping to understand which have the highest CPU or elapsed times.
  - View all statistics metrics available for each flow
  - View historical flow data



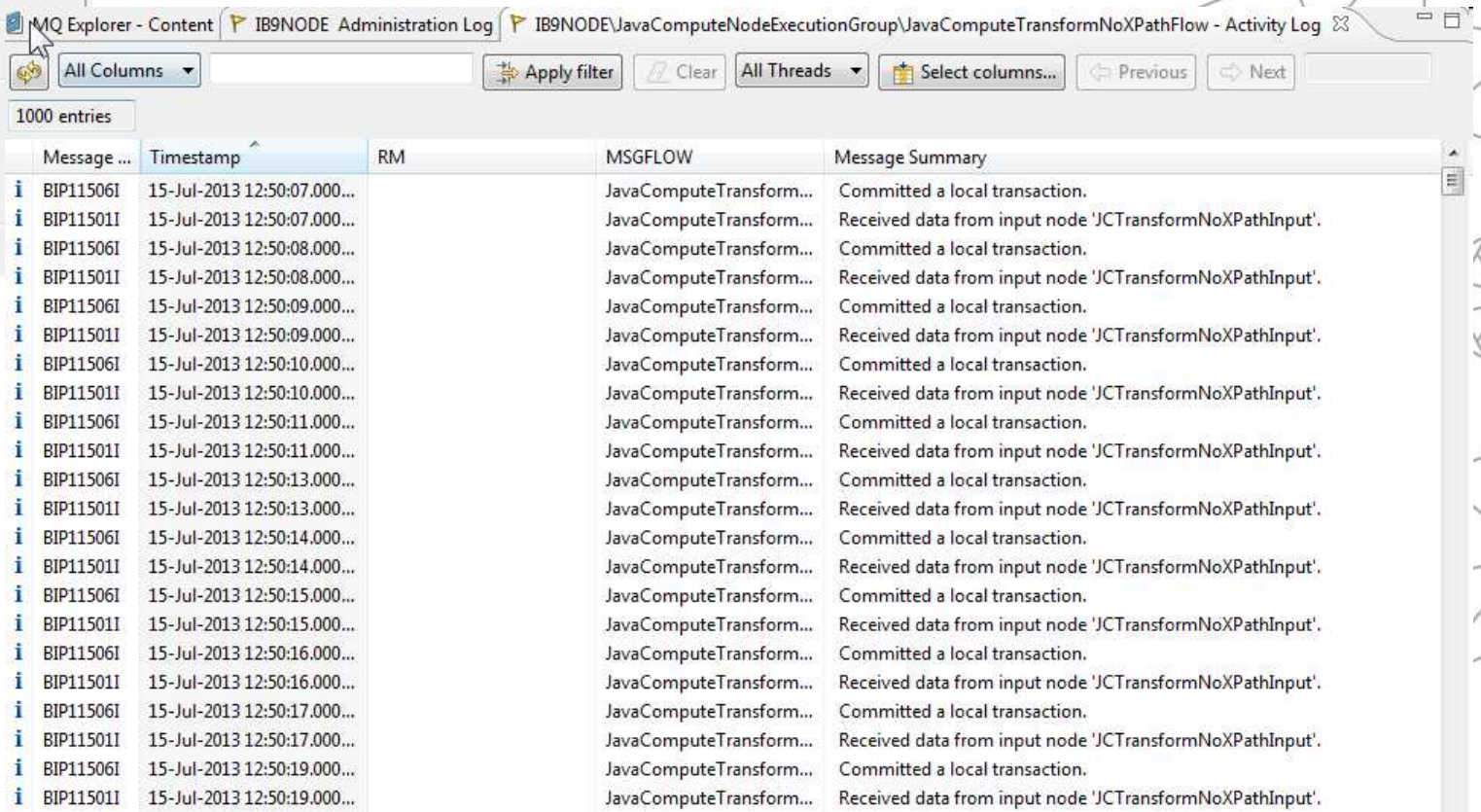
Throughput per message flow for last 15 seconds. Last updated at 09:50:29 GMT Daylight Time.

Flow name	Message Rate (messages/s)	Average Elapsed Time/ Invocation (ms)	Average CPU Time/ Invocation (ms)
Request	1.00	4.4	0.8
Reply	1.00	3.9	0.5
BackendReplyApp	1.00	1,001.7	1.1

Integration Bus Explorer & Activity Log



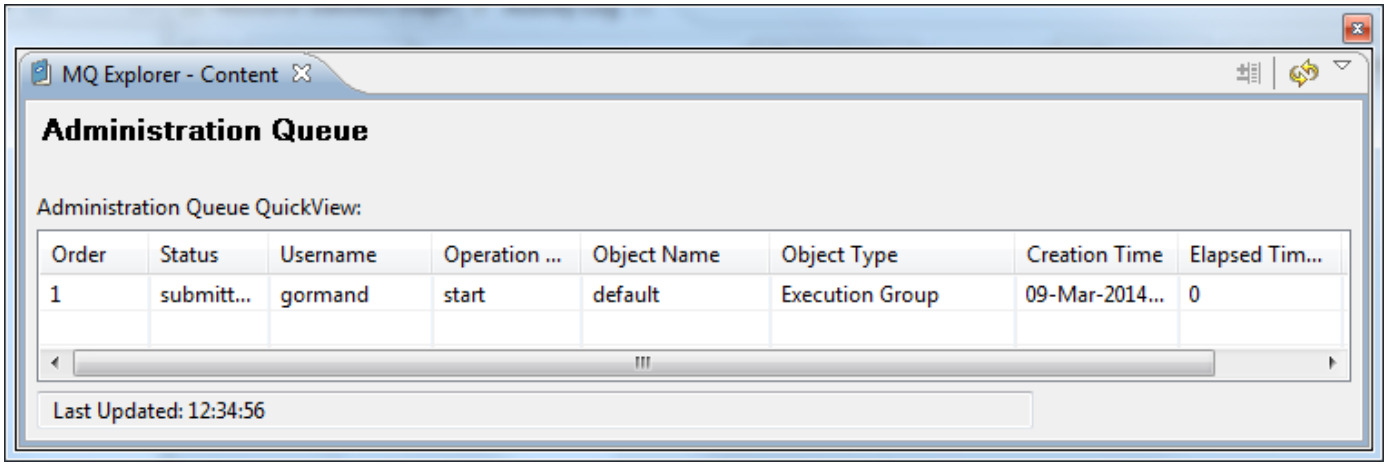
- View activity as it happens using explorer
- Filter by resource managers

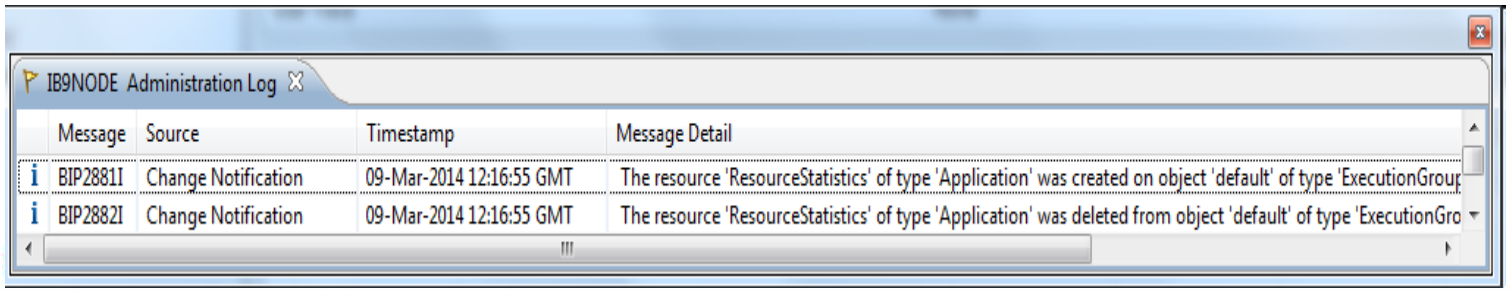


Message ...	Timestamp	RM	MSGFLOW	Message Summary
BIP11506I	15-Jul-2013 12:50:07.000...		JavaComputeTransform...	Committed a local transaction.
BIP11501I	15-Jul-2013 12:50:07.000...		JavaComputeTransform...	Received data from input node 'JCTransformNoXPathInput'.
BIP11506I	15-Jul-2013 12:50:08.000...		JavaComputeTransform...	Committed a local transaction.
BIP11501I	15-Jul-2013 12:50:08.000...		JavaComputeTransform...	Received data from input node 'JCTransformNoXPathInput'.
BIP11506I	15-Jul-2013 12:50:09.000...		JavaComputeTransform...	Committed a local transaction.
BIP11501I	15-Jul-2013 12:50:09.000...		JavaComputeTransform...	Received data from input node 'JCTransformNoXPathInput'.
BIP11506I	15-Jul-2013 12:50:10.000...		JavaComputeTransform...	Committed a local transaction.
BIP11501I	15-Jul-2013 12:50:10.000...		JavaComputeTransform...	Received data from input node 'JCTransformNoXPathInput'.
BIP11506I	15-Jul-2013 12:50:11.000...		JavaComputeTransform...	Committed a local transaction.
BIP11501I	15-Jul-2013 12:50:11.000...		JavaComputeTransform...	Received data from input node 'JCTransformNoXPathInput'.
BIP11506I	15-Jul-2013 12:50:13.000...		JavaComputeTransform...	Committed a local transaction.
BIP11501I	15-Jul-2013 12:50:13.000...		JavaComputeTransform...	Received data from input node 'JCTransformNoXPathInput'.
BIP11506I	15-Jul-2013 12:50:14.000...		JavaComputeTransform...	Committed a local transaction.
BIP11501I	15-Jul-2013 12:50:14.000...		JavaComputeTransform...	Received data from input node 'JCTransformNoXPathInput'.
BIP11506I	15-Jul-2013 12:50:15.000...		JavaComputeTransform...	Committed a local transaction.
BIP11501I	15-Jul-2013 12:50:15.000...		JavaComputeTransform...	Received data from input node 'JCTransformNoXPathInput'.
BIP11506I	15-Jul-2013 12:50:16.000...		JavaComputeTransform...	Committed a local transaction.
BIP11501I	15-Jul-2013 12:50:16.000...		JavaComputeTransform...	Received data from input node 'JCTransformNoXPathInput'.
BIP11506I	15-Jul-2013 12:50:17.000...		JavaComputeTransform...	Committed a local transaction.
BIP11501I	15-Jul-2013 12:50:17.000...		JavaComputeTransform...	Received data from input node 'JCTransformNoXPathInput'.
BIP11506I	15-Jul-2013 12:50:19.000...		JavaComputeTransform...	Committed a local transaction.
BIP11501I	15-Jul-2013 12:50:19.000...		JavaComputeTransform...	Received data from input node 'JCTransformNoXPathInput'.

Administration Queue / Log

- The tools include a lot of information that is useful to the administrator, for example:
  - **Administration queue**: What operational changes are currently pending
  - **Administration log**: What changes have been recently applied to the broker's configuration, and by whom?





Types of trace in Integration Bus

Trace is available for separate components which can be formatted to a file using:

- mqsichangetrace to enable trace
- mqsireadlog (BIPRELG) to read trace
- mqsiformatlog (BIPFMLG) to format

Types of trace available:

- User Trace – for you.
- Service Trace – for IBM Support
- Command Trace – for IBM Support
- CVP (Component Verification) Trace – for all

```
Aug  9  15:44  MQ91BRK.c91e9b41-3101-0000-0080-c0fdd5a003a1.trace.bin.0
Aug  9  17:28  MQ91BRK.c91e9b41-3101-0000-0080-c0fdd5a003a1.trace.bin.1
Aug  9  15:44  MQ91BRK.c91e9b41-3101-0000-0080-c0fdd5a003a1.trace.bin.2
Aug  9  15:44  MQ91BRK.c91e9b41-3101-0000-0080-c0fdd5a003a1.trace.bin.3
Aug  9  16:10  MQ91BRK.c91e9b41-3101-0000-0080-c0fdd5a003a1.userTrace.bin.0
Aug  9  17:28  MQ91BRK.c91e9b41-3101-0000-0080-c0fdd5a003a1.userTrace.bin.1
Aug  9  17:28  MQ91BRK.httplistener.trace.bin.0
Aug  9  17:28  MQ91BRK.httplistener.userTrace.bin.0
Aug  9  16:06  MQ91BRK.mqsichangeflowstats.trace.bin.0
Aug  9  16:06  MQ91BRK.mqsichangeflowstats.userTrace.bin.0
Aug  9  15:44  MQ91BRK.mqsichangetrace.trace.bin.0
Aug  9  15:44  MQ91BRK.mqsichangetrace.userTrace.bin.0
Aug  9  15:33  MQ91BRK.mqsicvp.trace.bin.0
Aug  9  17:28  MQ91BRK.mqsicvp.trace.bin.1
Aug  9  17:28  MQ91BRK.mqsicvp.userTrace.bin.0
Aug  9  15:41  MQ91BRK.mqsireadlog.trace.bin.0
Aug  9  15:41  MQ91BRK.mqsireadlog.userTrace.bin.0
Aug  9  16:11  MQ91BRK.mqsireportflowstats.trace.bin.0
Aug  9  16:11  MQ91BRK.mqsireportflowstats.userTrace.bin.0
Aug  9  17:04  MQ91BRK.mqsistop.trace.bin.0
Aug  9  17:04  MQ91BRK.mqsistop.userTrace.bin.0
Aug  9  17:28  MQ91BRK.service.trace.bin.0
Aug  9  17:28  MQ91BRK.service.userTrace.bin.0
```

User Trace Example

UserTrace tells you exactly what is happening as a message passes through an integration data flow.

See which **nodes** are being called, which **parsers** are being used, and what **ESQL** is executed.


```
CREATE FUNCTION Main() RETURNS BOOLEAN
BEGIN
    --CALL CopyEntireMessage();
    SET OutputRoot.XMLNSC.ID = InputRoot.XMLNSC.Customer.ID;
    RETURN TRUE;
END;
```

```
BIP2632I: Message received and propagated to 'out' terminal of MQ input node 'ParserStats.MQ Input'.
BIP6060I: Node 'ParserStats.MQ Input' used parser type ''Properties'' to process a portion of the incoming message of length '0' bytes beginning at offset '0'.
BIP6061I: Node 'ParserStats.MQ Input' used parser type ''MQMD'' to process a portion of the incoming message of length '364' bytes beginning at offset '0'. The parser type was selec
BIP6061I: Node 'ParserStats.MQ Input' used parser type ''XMLNSC'' to process a portion of the incoming message of length '158' bytes beginning at offset '364'. The parser type was s
BIP2537I: Node 'ParserStats.Compute': Executing statement  ''BEGIN ... END;'' at ('.ParserStats_Compute.Main', '2.2').
BIP2537I: Node 'ParserStats.Compute': Executing statement  ''SET OutputRoot.XMLNSC.ID = InputRoot.XMLNSC.Customer.ID;'' at ('.ParserStats_Compute.Main', '4.3').
BIP2539I: Node 'ImbESQLManager': Evaluating expression ''InputRoot.XMLNSC.Customer.ID'' at ('.ParserStats_Compute.Main', '4.30'). This resolved to ''InputRoot.XMLNSC.Customer.ID''.
BIP2568I: Node 'ParserStats.Compute': Copying sub-tree from ''InputRoot.XMLNSC.Customer.ID'' to ''OutputRoot.XMLNSC.ID''.
BIP2537I: Node 'ParserStats.Compute': Executing statement  ''RETURN TRUE;'' at ('.ParserStats_Compute.Main', '5.3').
BIP4015I: Message propagated to the 'out' terminal of node 'ParserStats.Compute' with the following message trees: ''.
BIP2638I: The MQ output node 'ParserStats.MQ Output' attempted to write a message to queue ''OUT'' connected to queue manager ''''. The MQCC was '0' and the MQRC was '0'.
BIP2622I: Message successfully output by output node 'ParserStats.MQ Output' to queue ''OUT'' on queue manager ''''.
```

SYSLOG / JOBLOG / STDOUT / STDERR

The Integration Bus runtime writes important operational messages to the system log:

- On z/OS, each JOBLOG includes all messages written by processes within the address space. The SYSLOG includes messages from all IIB JOBLOGs.
- On Windows, these messages are written to the event log.
- On Unix and Linux, these messages are written to the syslog.

STDOUT/STDERR may also be written to:

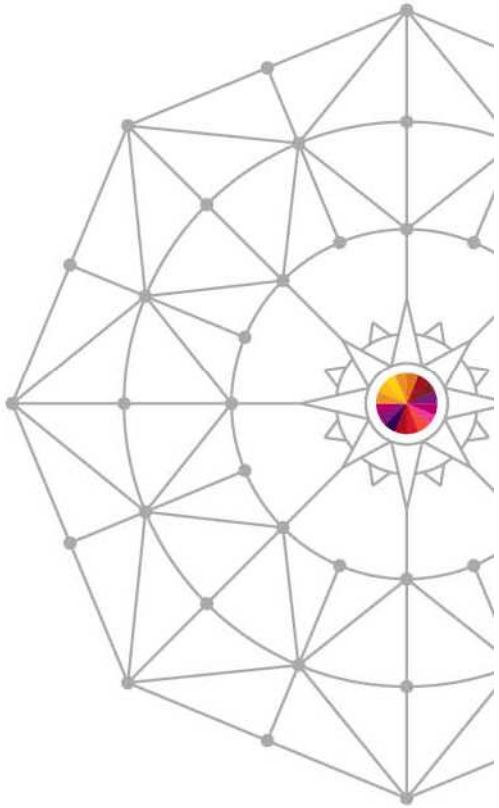
- On z/OS, each JOBLOG includes any STDOUT/STDERR written by processes within the address space.
- On Windows, the **console.txt** file in %MQSI_REGISTRY%\components\<node>\<EG UUID>
- On Unix/Linux, the **stdout** and **stderr** files in $MQSI_REGISTRY/components/<node>/<EG UUID>

```
  Display  Filter  View  Print  Options  Help
 ----------------------------------------------------
 SDSF JOB DATA SET DISPLAY - JOB MQ05BRK  (STC52908)
 COMMAND INPUT ===>
 PREFIX=MQ05BRK  DEST=(ALL)  OWNER=*  SYSNAME=*
 NP   DDNAME   StepName ProcStep DSID Owner    C Dest
      JESMSGLG JES2                2 MQ05BRK  C
      JESJCL   JES2                3 MQ05BRK  C
      JESYSMSG JES2                4 MQ05BRK  C
      ENVFILE  MQ05BRK           104 MQ05BRK  C
      DSNAOINI MQ05BRK           105 MQ05BRK  C
      STDOUT   MQ05BRK           106 MQ05BRK  C
      STDOUT   MQ05BRK           108 MQ05BRK  C
 _    STDERR   MQ05BRK           109 MQ05BRK  C
```

This was session 15025 - The rest of the week ……

	Monday	Tuesday	Wednesday	Thursday	Friday
08:00			What's Available in MQ and Broker for High Availability and Disaster Recovery?	Best Practices in Enhancing our Security with WebSphere MQ	MQ & CICS Workload Balancing in a 'Plexed' World
09:30				What's Wrong with MQ?	
11:00	The Dark Side of Monitoring MQ - SMF 115 and 116 Record Reading and Interpretation			IIIB - Internals of IBM Integration Bus	
12:15				Hands-on Labs for MQ - Take Your Pick!	
01:30		What's New in the MQ Family	MQ on z/OS – Vivisection	MQ Clustering - The Basics, Advances and What's New	
03:00	Introduction to MQ		WebSphere MQ CHINIT Internals	Using IBM WebSphere Application Server and IBM WebSphere MQ Together	
04:30	First Steps with IBM Integration Bus: Application Integration in the new world	What's New in IBM Integration Bus & WebSphere Message Broker	MQ & DB2 – MQ Verbs in DB2 & InfoSphere Data Replication (Q Replication) Performance	MQ Parallel Sysplex Exploitation, Getting the Best Availability From MQ on z/OS by Using Shared Queues	

SHARE
Technology · Connections · Results



Complete your session evaluations online at www.SHARE.org/AnaheimEval

SHARE in Anaheim