ISSW Technology Transfer Practitioner Enablement and Training

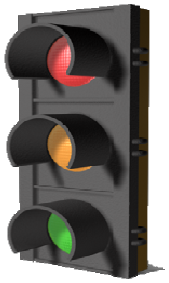
IBM Integration Bus
Workload Management and Policy

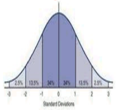
Sharpening ISSW Expertise

WebSphere. software

Part 1: Workload Management

Making it easier to control the processing speed in Integration Bus



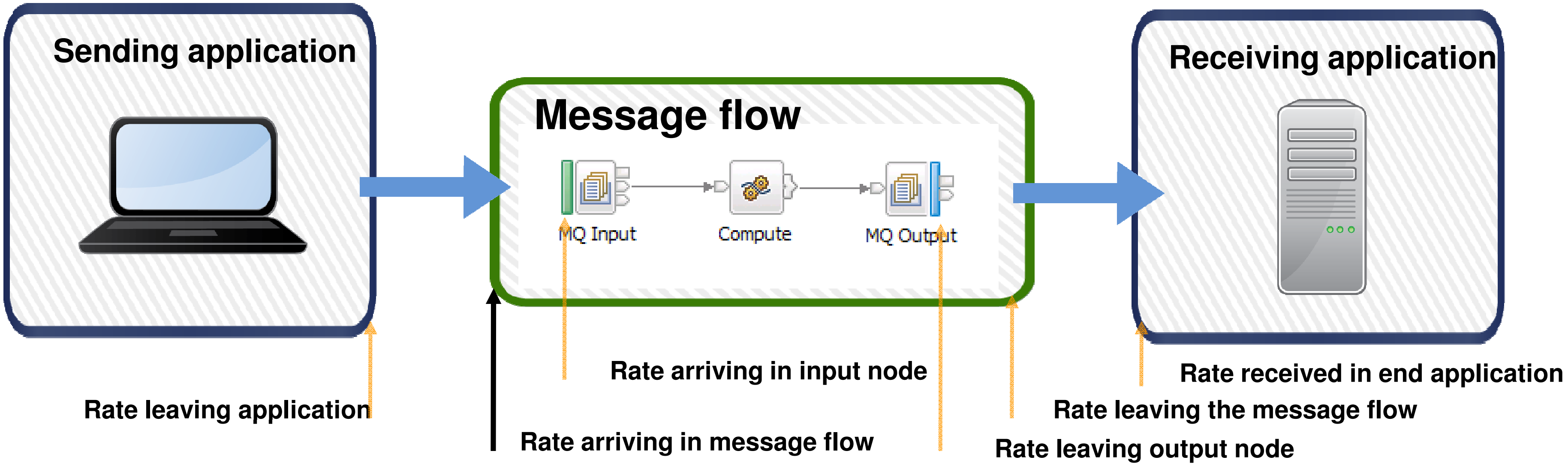


- Message Flow processing rate control
  - Provide intelligent mechanisms to increase and decrease processing speed
  - Policy specifies goals for the processing rate of a message flow
    - DECREASE: Notify, Delay, Redirect.
    - INCREASE: Add (increase threads), All (pre-emptively start all threads).
    - CONTROL: Restarting or reporting unresponsive flows or threads.
  - Learning mode that calculates the best values for policy.
  - Can be set at design time or changed operationally (CMP, command line and Web admin)

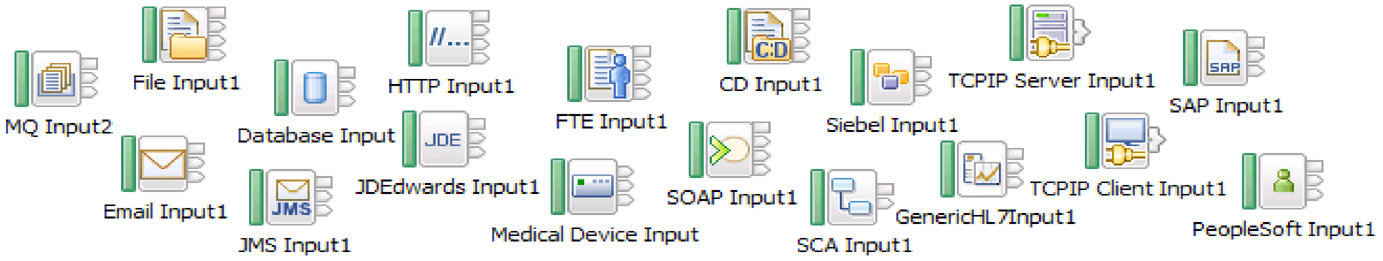
- Allow more diverse Workload Management:
  - Apply policy to different Broker artifacts:
    - Applications, services, individual nodes.
  - Scheduling like between 12:00 am and 6:00 am.
  - Long term SLA like number of messages per day.
  - Quality of service like batch versus low latency.

- Allow Workload Management to be defined in a policy separate from the message flow.
  - Policy can be defined in the bar file on the Message Flow.
  - Or in Broker's new built in Repository.
  - Use common repository component to allow policies to be stored in broker or else where.
  - Policy can be changed and used in broker independently of where it is stored.

Message Flow processing rate control - key rate terms

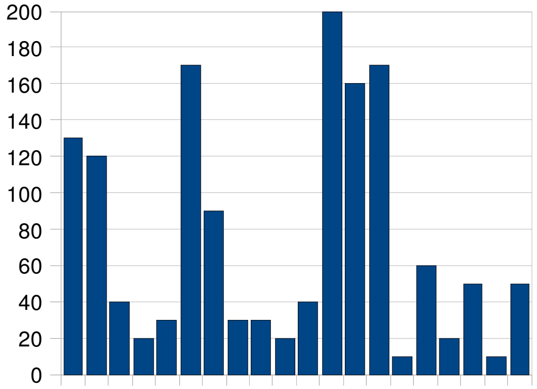


- There are various points in the flow of data from one application to another that the message rate can be measured and controlled.
- The Rate arriving in message flow is used as the controlling rate for the message flow:
  - Effectively limits Rate received in end application.
  - Includes the total rates of all input nodes of any type in the message flow.

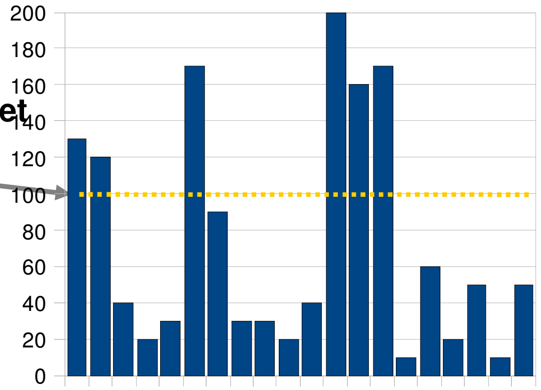


Message Flow processing rate control - Notification

Rate leaving sending application.

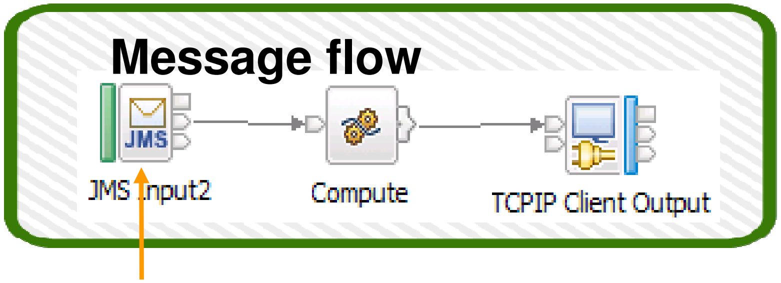


Processing rate in the message flow.



Notification threshold set on the message flow.

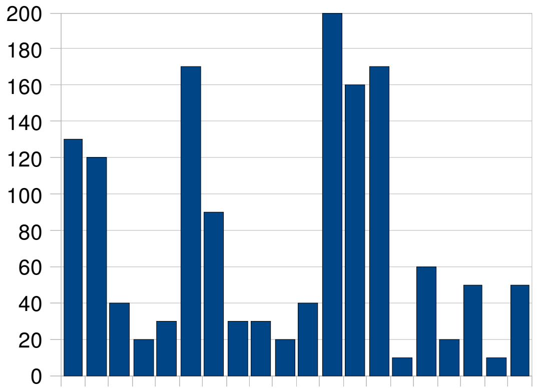
- A sending application generates a higher (or lower) than expected message rate.
- Configure broker to:
  - Send a notification when a threshold is exceeded or dropped below.
  - MQ Pub/Sub mechanism used to decouple notification from consumers.
    - Also write to Activity trace.
    - And write to User trace.
- Action to reduce message rate up to the user:
  - Ramp back sending application.
  - Email administrator to investigate.
  - Stop message flow to protect receiving application.



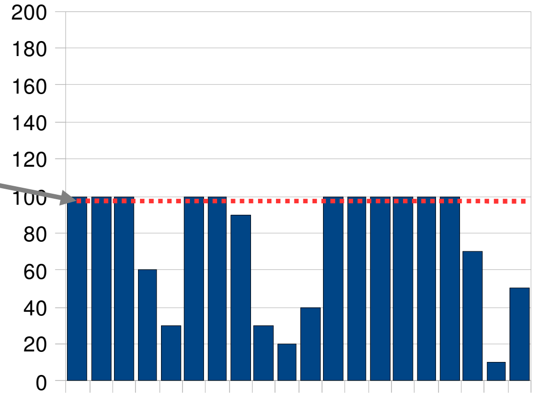
Count messages processed every 20 seconds

Message Flow processing rate control – Delay

Rate leaving sending application.

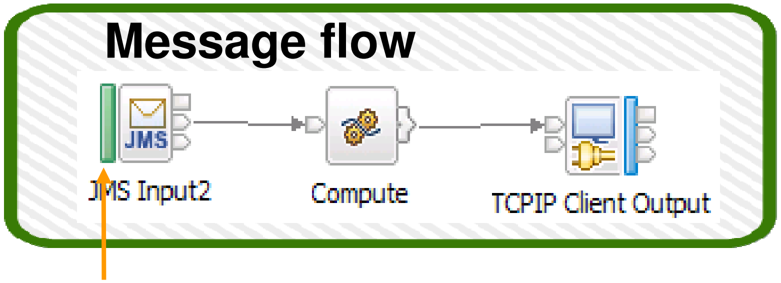


Processing rate in the message flow



Maximum rate set on the message flow.

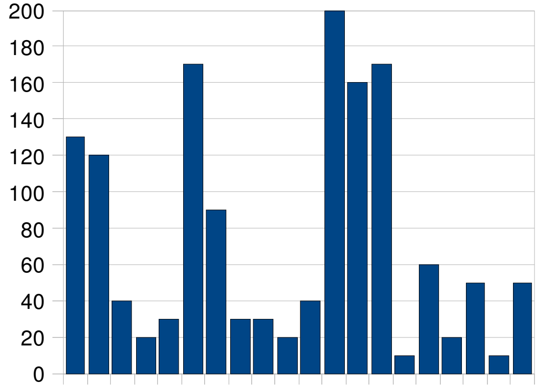
- A sending application generates a “bursty” message rate.
- A receiving application consuming the messages can handle the average rate but not short term fluctuations.
- Configure message flow to:
  - Limit the rate - Calculated on a message by message basis.
  - Delay messages if going too fast.
- **Time is measured before getting message.**
- **Time is measured again before getting next message.**
- **Delay is made if time difference is less than the time required to keep to the maximum rate.**



Delay point before any data is got or received.

Message Flow processing rate control – Redirect

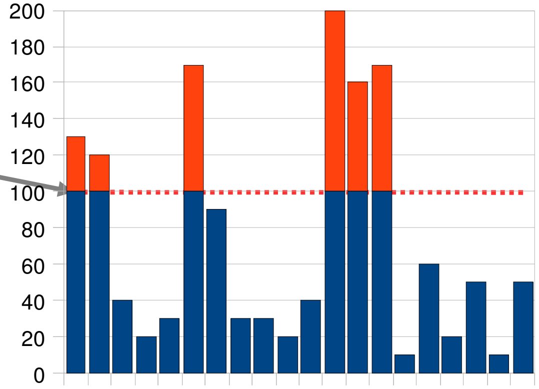
Rate leaving sending application.



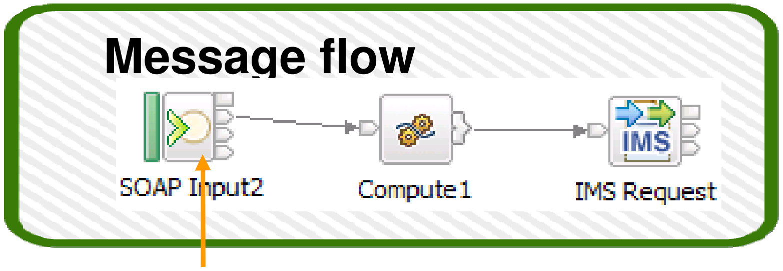
Processing rate in the message flow.

Maximum rate set on the message flow.

Limit action set to Redirect on the message flow.



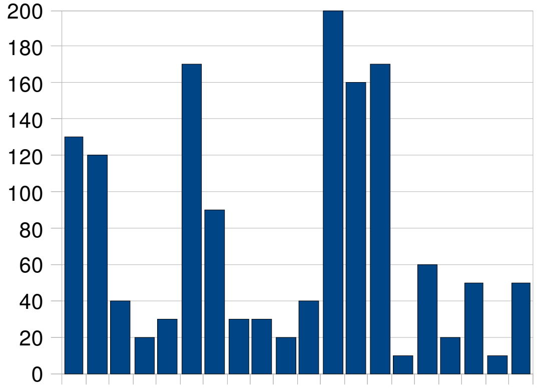
- Messages can NOT be delayed because either:
  - Messages from the sending application can not be queued.
  - Receiving application can not cope with the long term message rate.
- Configure message flow to:
  - Limit the rate - calculated on a message by message basis.
  - Redirect messages if going too fast (send to the failure terminal).
- **Time is measured before propagating message.**
- **Time is measured again before propagating next message.**
- **Redirect to failure is made if time difference is less than the time required to keep to the maximum rate.**



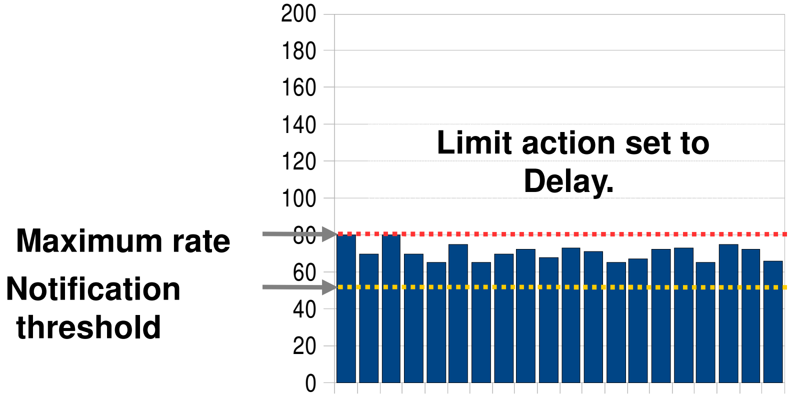
Redirect point before any data is propagated.

Message Flow processing rate control - Combining

Rate leaving sending application.



Processing rate in the message flo



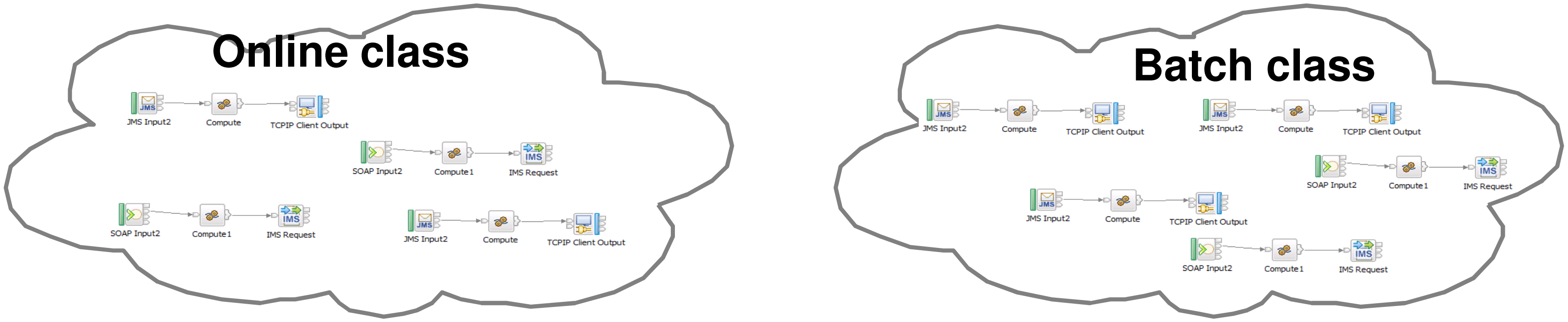
- It is possible to combine notification threshold and the maximum rate.
- In the example above the flow is configured to:
  - Send a notification whenever the rate exceeds 50 messages per second.
  - Use a delay to limit message rate to 80 messages per second.
- Message rate is only one possible condition for controlling workload:
  - CPU usage – both % and CPU time.
  - Time waiting for input messages.
  - Size of data – bytes per second or bytes per message.
- Other actions in addition to Notification, Delay and Redirect:
  - Stop message flow or fail messages.
  - User custom action written in java or c.

Message Flow processing rate control - Increasing the rate

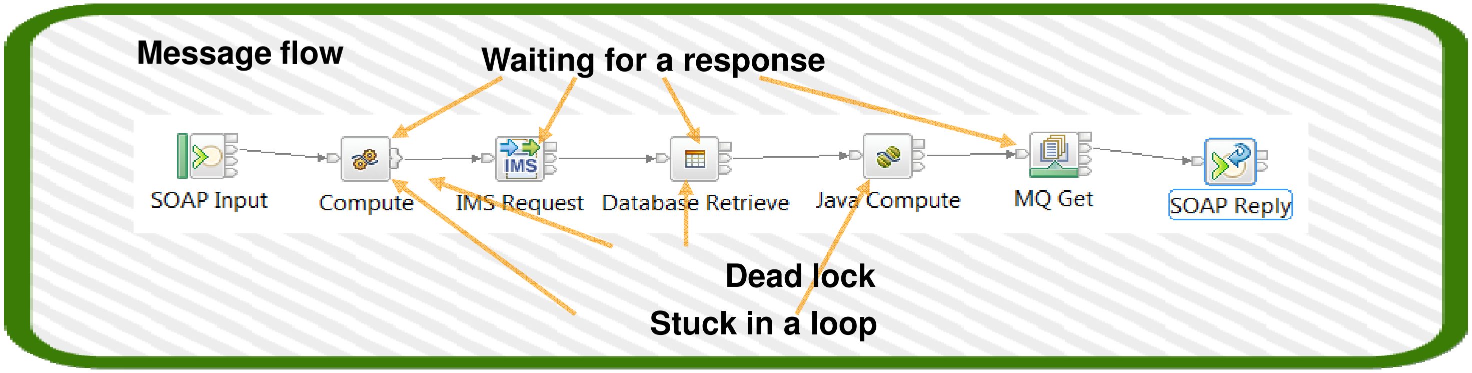
- Increasing rate is not as straight forward as decreasing:
  - The sending application might not be sending messages at higher enough rate.
  - Machine might not have enough resources to meet target rate.
- Notification mechanism can be used both for exceeding threshold and dropping below threshold to trigger an external action – making sending app send more data.
- Restrict other message flow's rates to allow important flows to have more resources.
- Add a mechanism to allow shaping to increase threads in an attempt to increase rate.
  - Dynamic nature might cause strange behaviour in production.
  - More threads is not always better or faster.
- Add a “learning mode” where shaping runs and tries different threading, delaying and redirecting strategies and produces a set of optimum work load parameters like:
  - Additional instances.
  - Maximum rate.

Message Flow processing rate control - Across message flows

- Mechanism so far have concentrated on individual flow goals.
- Need a mechanism to manage workload across lots of flow:
  - Prioritise work in certain flows above other flows.
  - Allow applications which need low latency to be protected from batch processing.
- Introduce classes of workload that can be assigned to flows
  - Batch class – any work that can be postponed to a later time.
  - Online class – any work that must be completed as quickly as possible.
  - Allow user to define their own classes which have relative priority.
- Work load would be sampled periodically (20 seconds) and relative speed of flows in each class would be modified to achieve the class goal.
  - Flows in the Online class are going slowly then delay flows in the Batch class.



Message Flow processing rate control – Unresponsive flows



Message flow

Waiting for a response

Dead lock

Stuck in a loop

STOP

- Message flow processing can become unresponsive when:
  - Waiting for a response from an external system.
  - Processing an infinite loop or a calculation that takes a very long time.
  - Deadlocked between two resources.
- How to get out of this situation?
  - Kill the execution group process.

    Only mechanism in the past

  New Run a command to force the stopping of a message flow.

    or REST, Web, CMP, Toolkit.

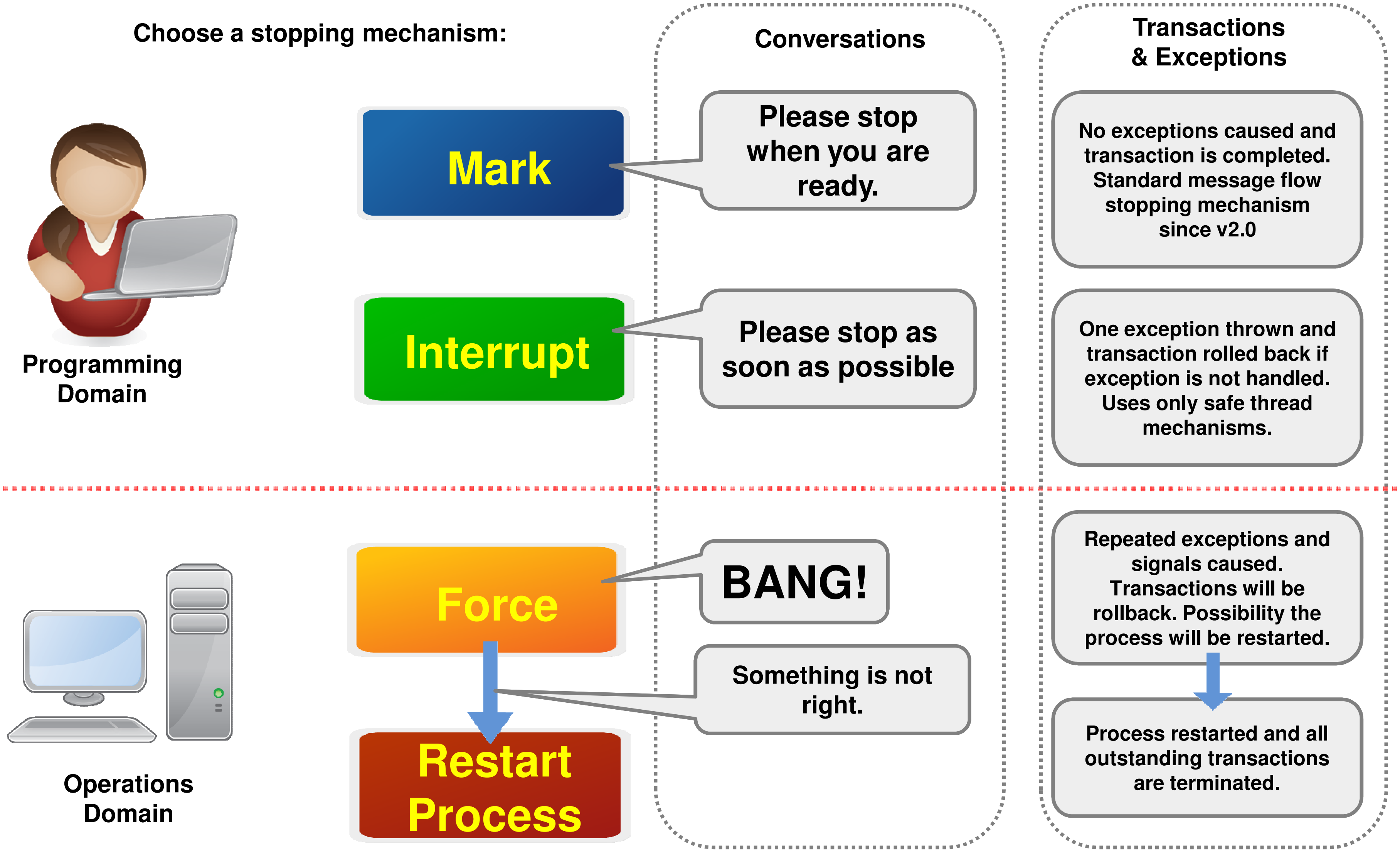
  New Define a policy which specifies the maximum time allowed for processing a message.

    •Publish a report on what the Instance (thread) is doing and has just done.

    •Optionally restart: Execution group, Message flow or Instance (thread).

Message Flow processing rate control – Stopping an Instance

Choose a stopping mechanism:



Conversations

Transactions & Exceptions

Programming Domain

Mark

Please stop when you are ready.

No exceptions caused and transaction is completed. Standard message flow stopping mechanism since v2.0

Interrupt

Please stop as soon as possible

One exception thrown and transaction rolled back if exception is not handled. Uses only safe thread mechanisms.

Operations Domain

Force

BANG!

Repeated exceptions and signals caused. Transactions will be rollback. Possibility the process will be restarted.

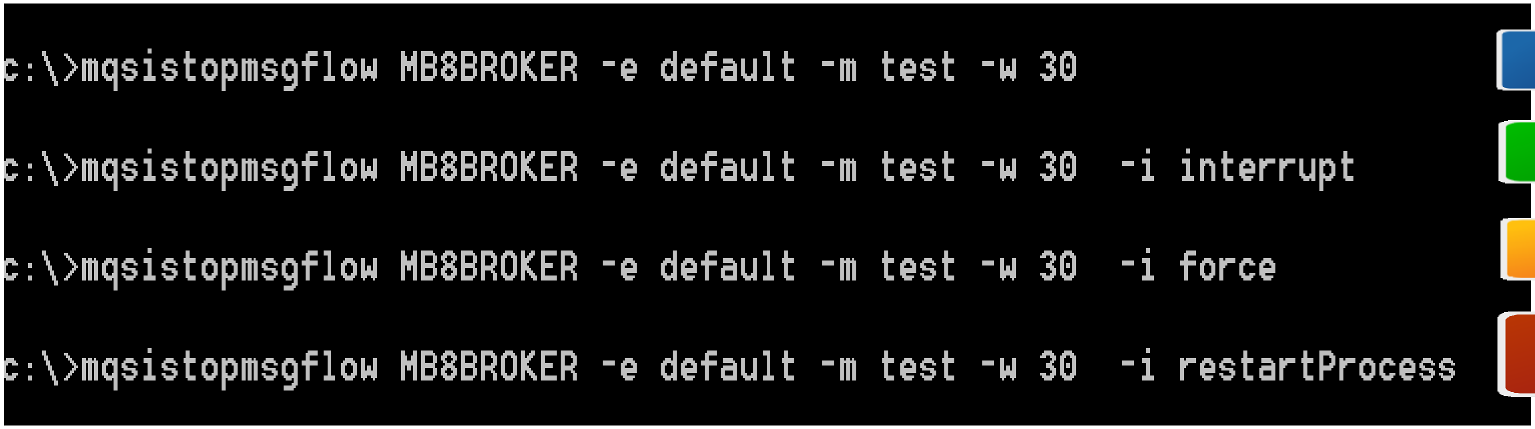
Restart Process

Something is not right.

Process restarted and all outstanding transactions are terminated.

Message Flow processing rate control – Stopping flow using a command

- **mqsistopmsgflow** in previous versions uses the Mark mechanism to stop flows.
- Add a new option to allow Interrupt Force o Restart Process instead.
- It is possible to combine commands to escalate stopping the flow. For example:



```
c:\>mqsistopmsgflow MB8BROKER -e default -m test -w 30

c:\>mqsistopmsgflow MB8BROKER -e default -m test -w 30  -i interrupt

c:\>mqsistopmsgflow MB8BROKER -e default -m test -w 30  -i force

c:\>mqsistopmsgflow MB8BROKER -e default -m test -w 30  -i restartProcess
```

Mark

Interrupt

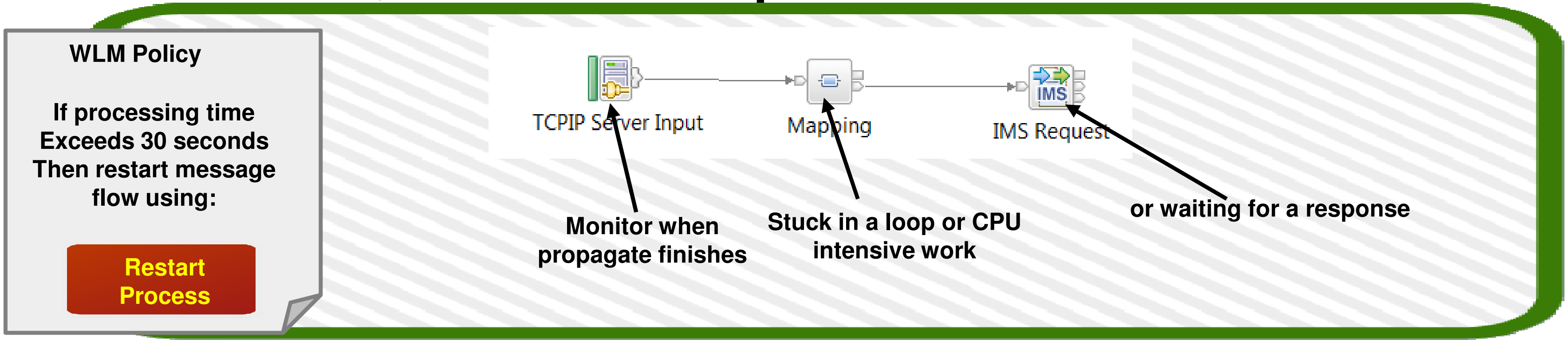
Force

Restart Process

- Also exposed through the following APIs:
  - CMP.
  - REST.
  - Web Administration Interface.
  - IBM Integration Toolkit.

Message Flow processing rate control – Detecting an unresponsive flow

Message flow



WLM Policy

If processing time Exceeds 30 seconds Then restart message flow using:

Restart Process

Monitor when propagate finishes

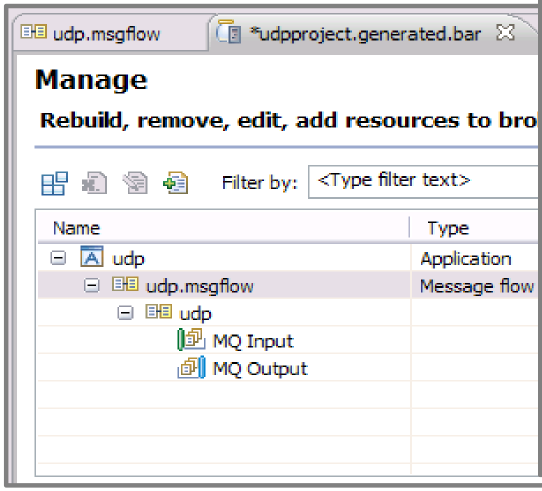
Stuck in a loop or CPU intensive work

or waiting for a response

- WLM policy attached to a flow to specify a maximum processing time for any instance (thread) after it has left an input node.
- Monitored using a separate thread that publishes a notification when a time out occurs.
- Additional corrective action can be configured:
  - No action, leave it to user to perform an action based on notification.
  - Restart message flow: Mark or Interrupt or Force or Restart Process
  - Stop message flow: Mark or Interrupt or Force or Restart Process
  - Restart message flow instance (thread): Mark or Interrupt or Force

Message Flow processing rate control - Workload properties that can be set.

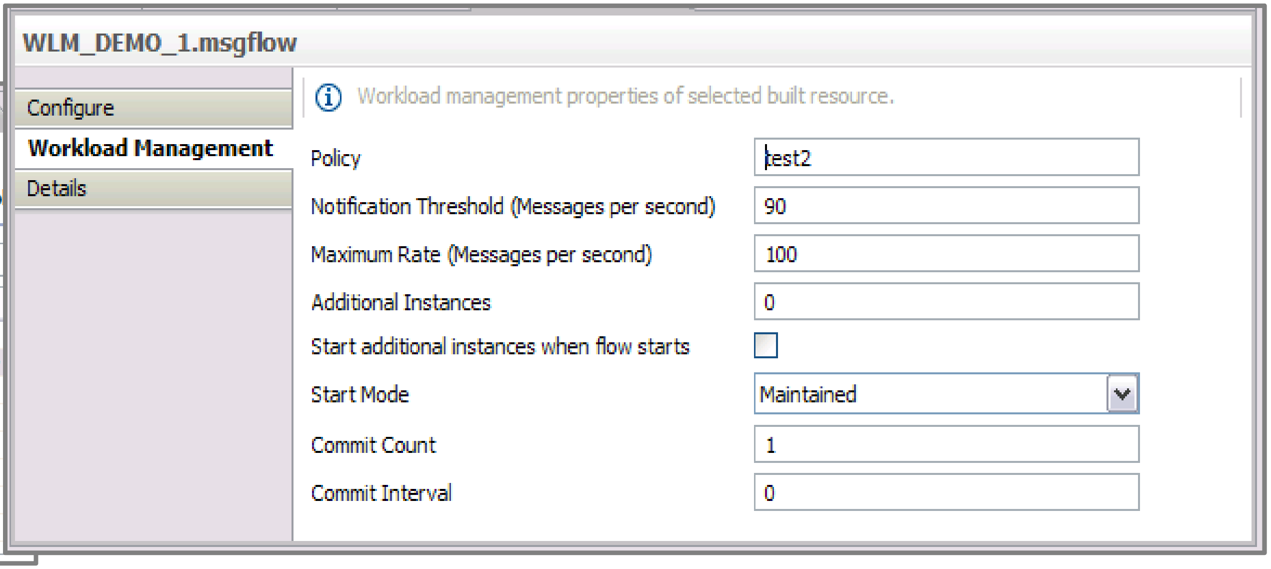
- Notification threshold (Messages per second)
  - The rate at which notification will be published when either exceeded or dropped below.
  - Default: Infinite.
- Maximum rate (Messages per second)
  - The maximum rate a flow will process messages.
  - Default: Infinite.
- Limit action.
  - Delay or Redirect.
  - Default: Delay.



Manage

Rebuild, remove, edit, add resources to bro

Name	Type
udp	Application
udp.msgflow	Message flow
udp	
MQ Input	
MQ Output	



Workload management properties of selected built resource.

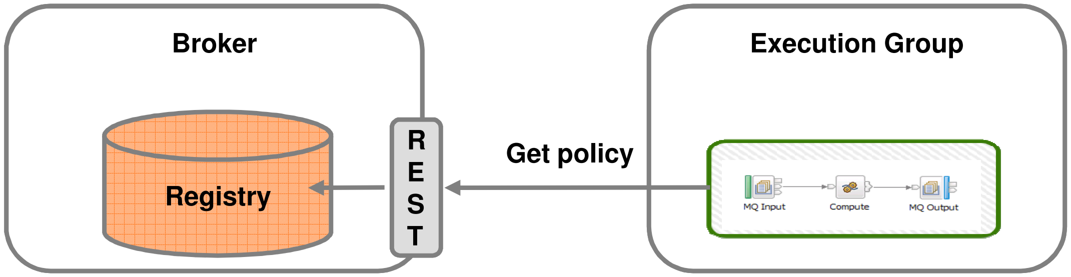
Policy	test2
Notification Threshold (Messages per second)	90
Maximum Rate (Messages per second)	100
Additional Instances	0
Start additional instances when flow starts	□
Start Mode	Maintained
Commit Count	1
Commit Interval	0

Part 2: Policy

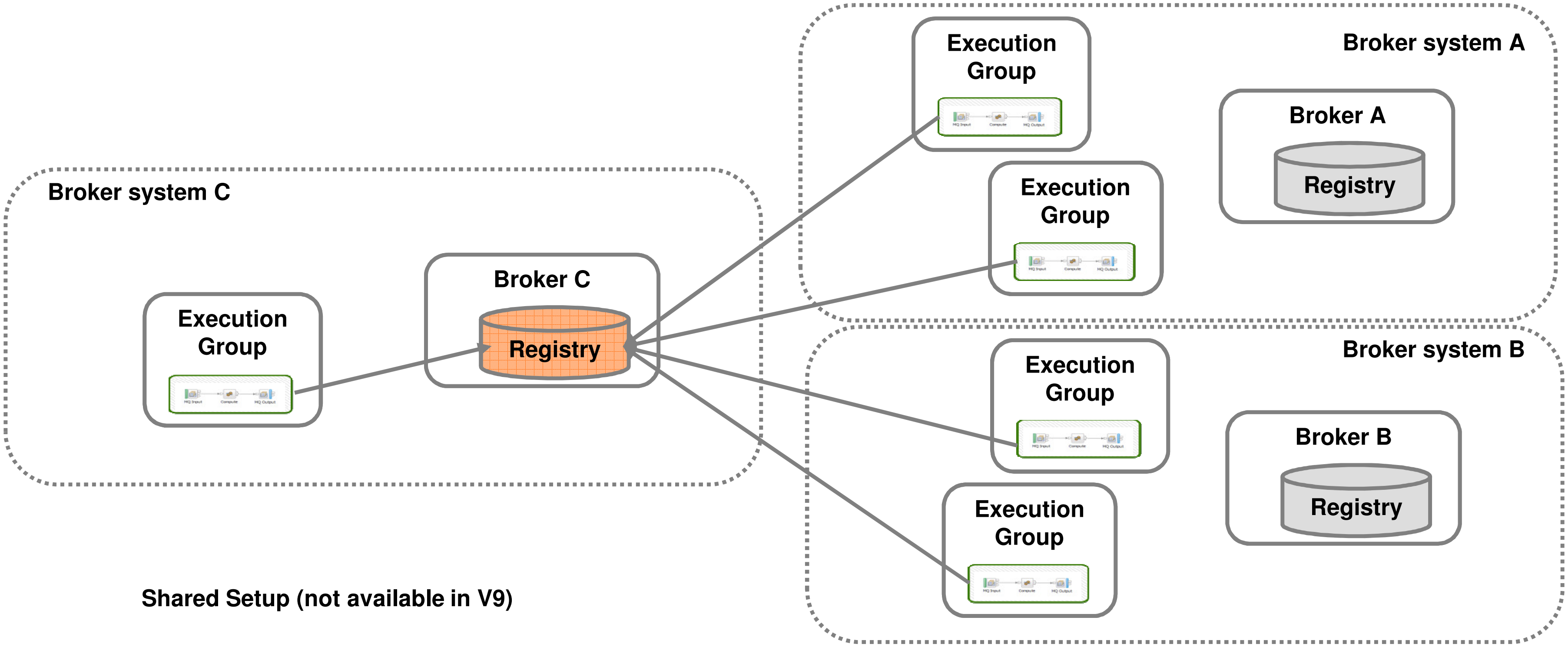
Policy - Storing and using policies in a broker

- Define policy as: any configuration that can be changed after initial developm
  - Connection details, Transformation, Routing, Security.
  - And Workload Management.
- Integration Bus has had two main mechanisms:
  - Built in configurable services:
    - Very broker orientated.
    - Local to the broker using it and no sharing across systems.
    - Simple to setup and use.
  - External repository using WSRR:
    - General purpose tool.
    - Always separate from message broker and shareable between systems.
    - More complex to install and setup.
- Version 9 introduces a new built in registry:
  - Registry can now be local to broker or housed in a remote broker (not V9).
  - Allows the registry to be come part of the integration solution without extra parts.
  - Uses message broker tools to interact with Registry.
  - Based on standards like OSLC and linked data to make it flexible for the future.

Policy - Setting up the Registry



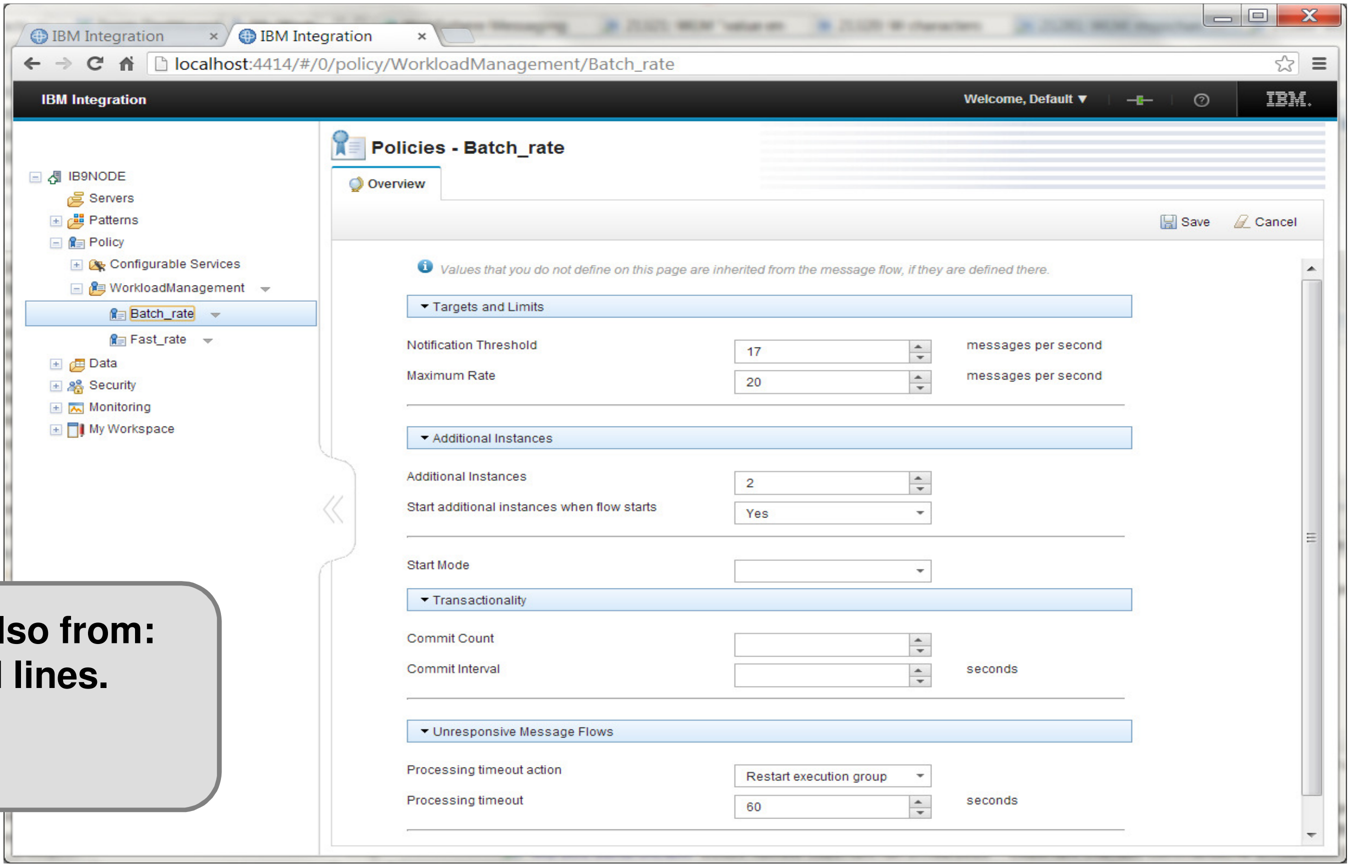
Out of the Box Setup



Shared Setup (not available in V9)

Policy - Managing the polices using the Web Administration Interface

- Enhancements to the Web Administration Interface to allow creation, update, retrieve and delete.



Policies - Batch_rate

Values that you do not define on this page are inherited from the message flow, if they are defined there.

▾ Targets and Limits

Notification Threshold: 17 messages per second

Maximum Rate: 20 messages per second

▾ Additional Instances

Additional Instances: 2

Start additional instances when flow starts: Yes

Start Mode:

▾ Transactionality

Commit Count:

Commit Interval: seconds

▾ Unresponsive Message Flows

Processing timeout action: Restart execution group

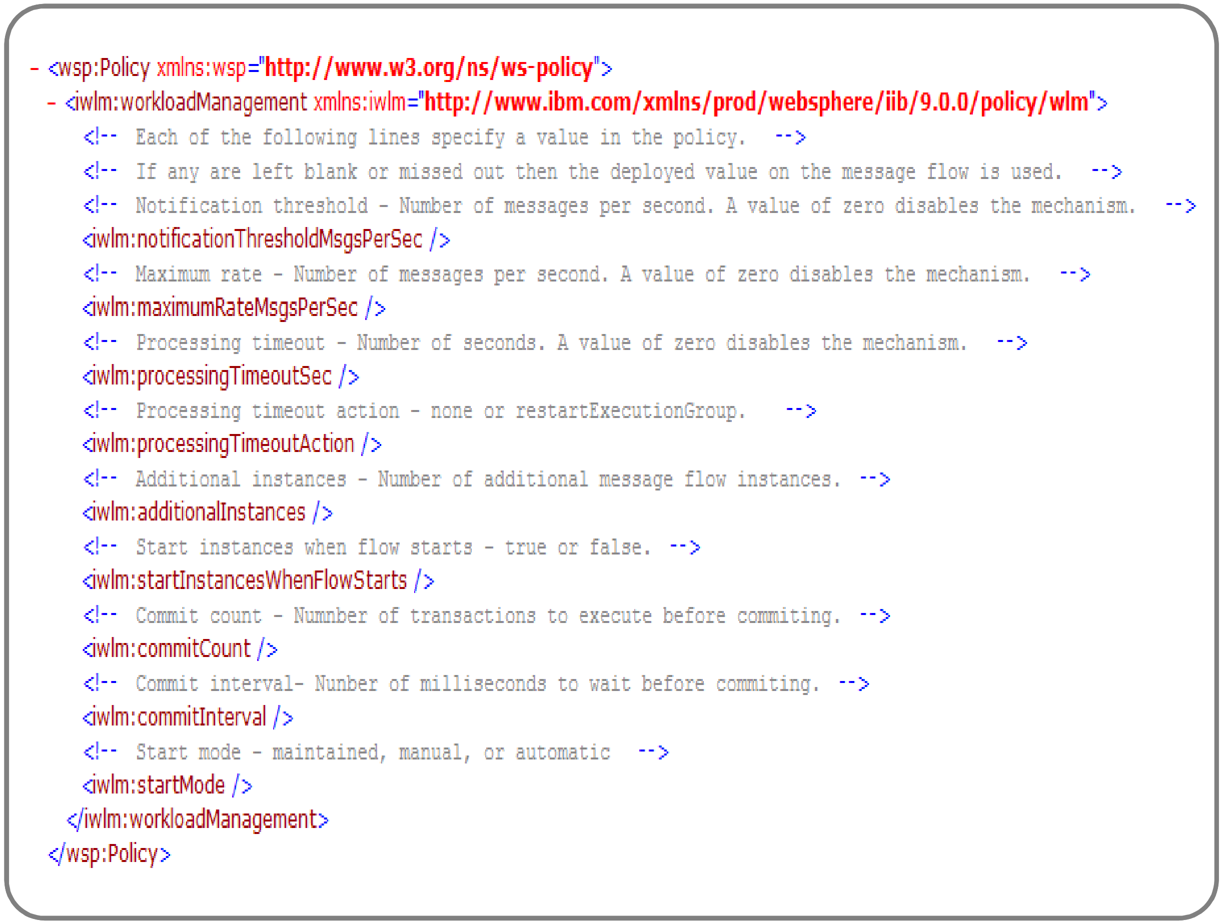
Processing timeout: 60 seconds

- **Full support also from:**
  - **Command lines.**
  - **REST API.**
  - **CMP API**

- **Policies can be attached and detached from message flows.**

Policy – using commands

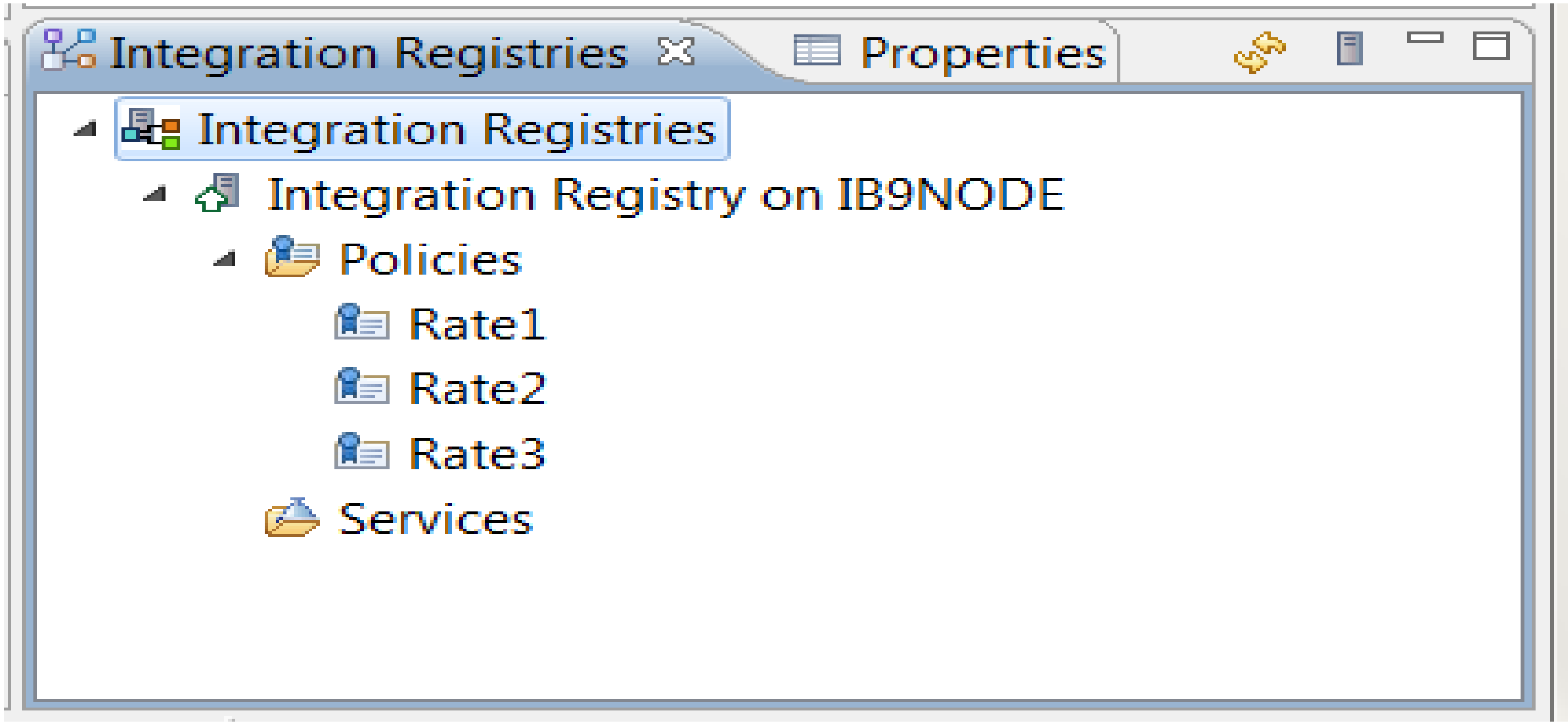
- Commands to create, report, update and delete policies:
  - mqsicreatepolicy
  - mqsireportpolicy
  - mqsichangepolicy
  - Mqsideletepolicy
- Commands use an xml policy document to create and update policies
- **Commands can also be used to attach and detach policies from a message flow:**
  - **mqsiattachpolicy**
  - **mqsidetachpolicy**



```
- <wsp:Policy xmlns:wsp="http://www.w3.org/ns/ws-policy">
  - <iwlm:workloadManagement xmlns:iwlm="http://www.ibm.com/xmlns/prod/websphere/iib/9.0.0/policy/wlm">
      <!--  Each of the following lines specify a value in the policy.   -->
      <!--  If any are left blank or missed out then the deployed value on the message flow is used.   -->
      <!--  Notification threshold - Number of messages per second. A value of zero disables the mechanism.   -->
      <iwlm:notificationThresholdMsgsPerSec />
      <!--  Maximum rate - Number of messages per second. A value of zero disables the mechanism.   -->
      <iwlm:maximumRateMsgsPerSec />
      <!--  Processing timeout - Number of seconds. A value of zero disables the mechanism.   -->
      <iwlm:processingTimeoutSec />
      <!--  Processing timeout action - none or restartExecutionGroup.    -->
      <iwlm:processingTimeoutAction />
      <!--  Additional instances - Number of additional message flow instances.  -->
      <iwlm:additionalInstances />
      <!--  Start instances when flow starts - true or false.  -->
      <iwlm:startInstancesWhenFlowStarts />
      <!--  Commit count - Numnber of transactions to execute before commiting.  -->
      <iwlm:commitCount />
      <!--  Commit interval- Nunber of milliseconds to wait before commiting.  -->
      <iwlm:commitInterval />
      <!--  Start mode - maintained, manual, or automatic   -->
      <iwlm:startMode />
    </iwlm:workloadManagement>
  </wsp:Policy>
```

Policy – using the toolkit

- Toolkit can be used to look at available policies:



- Integration Registries
  - Integration Registry on IB9NODE
    - Policies
      - Rate1
      - Rate2
      - Rate3
    - Services

- **In the future, add comprehensive tools for authoring and deploying polices from the toolkit.**

IBM Integration Bus Version 9

V9

- *Not every thing mentioned will be in Version 9 ...*
- What is included:
  - Notification threshold
  - Maximum rate with delay
  - Processing timeout with options to restart execution group
  - Workload management policy for new and old properties
  - Built in policy repository for local broker use
  - All tools required to use and manage repository:
    - Web Administration Interface
    - Command lines
    - Programmable interface (REST and java)s

Summary

- **IBM Integration Bus V9 has intelligent and simple to use Workload management**

- **Defined using policy separately from the artifacts using it**

- **Key area for future development with a continued focus on:**
  - **Controlling processing**
  - **Using policy**
  - **Displaying what a flow is doing**

Thank
You