Introduction

Once message flow coding and test are complete, you are in a position to configure the production environment. A key question at this point is how many copies of each of the message flows should be run. Run too few and the required message throughput rate will not be met. Run too many and the system could become flooded causing an unnecessary overhead as the operating system attempts to manage many concurrent units of work. If the execution groups have very large memory requirements, this can lead to excessive paging that in the most extreme cases can render the machine useless as the operating system attempts to manage the competing needs of the different message flows running in their respective execution groups. This is clearly not a good position to be in and can easily be avoided by understanding the processing characteristics of the message flow.

This article helps you avoid this situation by providing an understanding of the principle of how a message flow runs in an execution group and helps you understand the processing characteristics of your own message flows.

In a complex environment, there will be many message flows running within a single WebSphere® Business Integration Message Broker. In such a case, it is ideal to understand the characteristics of all message flows.Â Where this is not feasible or practical it should be possible to divide the message flows into a limited number of categories where each message flow in the category has broadly similar processing characteristics.Â The number of copies of the message flow should be allocated on the characteristics of the category.Â To do this, you will need to study in detail at least one flow per category.

This article assumes that you are familiar with the major components and configuration of WebSphere Business Integration Message Broker V5.Â It also assumes that the ESQL code and node usage within the message flows are optimized for performance.

The output of this testing should be a message rate that is achievable with each of the message flows. The next stage is to compare the rate at which messages can be processed with the message flow against the required message rate.

The rate at which message flows need to process messages often varies considerably.Â Some message flow may only be used at month end for example and then only need to process hundreds or low thousands of messages over a day or two.Â Other message flows may have to process messages every day and must be capable of processing tens, hundreds, or even thousands of messages per second.
Those message flows with low throughput requirements probably need no further study or planning ifÂ a small number of instances of the message flow will be able to process the low volume of messages in sufficient time.
For those message flows with a message throughput requirement that is significantly greater than the rate which was achieved with one copy of the message flow, more involved analysis is required to make optimal use of the resources that you have.
Without this you could easily over allocate resources, such as additional brokers and machines.
The analysis required is not complex and can be extremely useful in understanding the characteristics of your message flow and system as a whole.Â What you need to determine is whether the message flow is CPU or I/O bound.Â You also need to determine whether by tuning the I/O configuration greater message throughput is possible with a single copy of the message flow.Â Once you know this, it will then be possible to finally determine how many copies of the message flow and how much processing power is required to achieve the required message rate.Â In this next section, we examine how to determine whether your message flow is CPU or I/O bound.

- Custom Performance Analysis using the Microsoft Performance Data Helper for details on how the Windows Performance Monitor may be used to measure system performance).
- iostat and vmstat on UNIX systems.
- Resource Measurement Facility (RMF) or Spool Display and Search Facility (SDSF) on z/OS.

Use the tool appropriate to the platform on which the WebSphere® Business Integration Message Broker is running.Â The aim of the testing is to determine how much of one CPU a single copy of the message flow is able to use while processing messages.
When the message flow is the only active work in the system, the system CPU utilization will tell you how much CPU is being consumed by the message flow.Â When there is other work active in the system, you will need to subtract the sum of that work to

determine the amount of CPU consumed by the message flow.Â Alternatively, there may be CPU consumption reporting at the process or address space level.

To complete the analysis, you need to understand how many CPU processors there are in the system on which the measurements are taking place. This is so that you know what CPU utilization represents one processors worth (and so one message flows worth) of processing.Â Â On a four processor machine, each processor provides 25 percent of the available CPU.Â On an eight processor machine, each processor provides 12.5 percent of the available CPU.

If the execution group is responsible for a system CPU utilization of 25 percent on a four-processor system, the message flow is fully utilizing one processor and so is CPU bound.Â This is the maximum rate that the single message flow will be able to process messages on that machine.Â A single copy of a message flow will not be able to use more than one processor (assuming messages are only placed on one input queue). Additional copies of the message flow would be able to use additional processors.

If the message flow was only responsible for a system, CPU utilization of 15 percent on a four-processor system, Â for example, then it is very likely that message processing is I/O bound (assuming a plentiful supply of messages on the input queue).Â The 15 percent utilization is significantly below the 25 percent that could be expected.Â At this point, you need to examine I/O activity since this is the most likely reason for low CPU utilization. Look for disks with poor response times or high utilizations.

The most likely sources of I/O delays are:

- **The queue manager log when processing persistent messages.**
  The solution to this is to increase the buffer sizes and log extent size of the queue manager log.Â Also, consider using low latency disks on which the log is located such as one with a nonvolatile write cache.Â This can significantly improve the rate at which persistent messages can be processed.
- **The database log when there is database Insert/Delete/Update activity within the message flow.**
  The same considerations apply as for the queue manager log.
- **Poor response time from an application that has been called synchronously.**
  This may be on the same or a different machine. The solution to this problem is to investigate the behavior of the other application and make whatever performance improvements are possible.Â This may include running multiple copies of the application or increasing the hardware

resources of the machine on which the application. runs.
Â Â

- **Poor database buffer tuning.**
  If the message flow reads data from a very large table in a database a large amount of I/O may have to be performed by the database to retrieve the required row.Â The solution to this is to tune the database.

At this point, you should know whether a particular message flow is CPU bound or I/O bound.Â If it is I/O bound, you should have taken the necessary steps to reduce the extent of the I/O contention.Â It will not always be possible to change an I/O bound message flow into a CPU bound one but in many cases it is possible to reduce the extent to which it is I/O bound.Â Reducing the I/O delays will lead to an increase in message throughput so there is every incentive to go through this exercise.

Once you have determined the rate that is achievable with a single copy of the message flow, start testing with an increasing number of copies.Â An initial estimate of the number of copies required for a CPU bound message flow is given by the result of (required message rate / the rate possible with one copy of the message flow). This is also likely to be the number of processors required as each copy of the message flow is capable of fully utilizing one processor. Where the number of processors required exceeds the capacity of one machine, you will need to plan for a larger machine or use multiple machines to achieve the target message rate.

If the message flow is I/O bound, it is likely that you will need to run more copies than in the CPU bound case. Start with the same number of copies of the message flow as for the CPU bound case but expect to have to increase the number of copies required.Â How many more copies are required will depend on the extent to which the message flow is I/O bound.Â Continue to increase the number of copies and monitor both CPU utilization and message rate until the target message rate is achieved or the processors are fully utilized.Â If the target message rate has not been achieved, you will need to plan for a larger machine or use multiple machines to achieve the target message rate.Â If during this testing CPU consumption rises without an increase in message rate, further investigation for bottlenecks and tuning may be necessary to maximize the message rate.Â Bear in mind that it may not be possible to totally eliminate I/O delays.

At this point, you should have determined how many copies of the message flow are required to achieve the target message rate.Â This may exceed the capacity of one machine dependent on the complexity of the message flow and the speed of the processors. The number of copies required for CPU bound and I/O bound message flows is likely to be noticeably different.