

Have you **Outgrown** Scheduling Environments?

ThruPut Manager[®] and Batch Routing

Routing Overview

With Workload Manager, IBM introduced Scheduling Environments which was an improvement over using the Job Class and System Affinity mechanisms for routing. In the same way, ThruPut Manager's Job Binding is a superior solution to Scheduling Environments.

Have You Outgrown Scheduling Environments?

For simple routing based on few resources and few machines, Scheduling Environments may suffice. However

The benefits of a comprehensive routing scheme:

- *hardware utilization is maximized*
- *software licensing costs are minimized*
- *batch throughput is optimized*

they don't attack the hard problems when you have a dynamic environment with many routing dependencies. Is your routing setup less than optimal because it is difficult to redefine your Resource Elements and Scheduling Environments and re-educate users and

operations? Have you been unable to move workload around for optimal effectiveness because you don't have a routing mechanism that can handle your total batch workload and its various sub-categories? These may all be signs that you are ready to move up to ThruPut Manager's Job Binding solution.

An Architected Approach

The difference is in the robust architecture of Job Binding which addresses all the requirements of a comprehensive batch routing facility. This paper explains the architecture behind Job Binding and Scheduling Environments and the consequences for you in the datacenter and your users.

Why Datacenters Need Routing

In a standard JES2 MAS (Multi-Access Spool) complex, jobs are free to run anywhere. But dependent resources and other conditions are frequently not available on all systems, so a routing mechanism has always been needed.

The list below gives several reasons to direct a job to a particular LPAR. All the reasons represent either a resource or condition that must exist on the machine a job is to execute on. The complexity builds exponentially when multiple resources or conditions are needed by a single job. As you look over the list, ask yourself how often these resources or conditions combine in your workload:

- redistribute your workload due to a new or upgraded machine
- reduce the number of machines licensed for a software product
- cope with outages
- deal with month-end or seasonal increases in workload
- dedicate a workload to a specific LPAR, say "night time engineering batch"
- need a job to run where
 - the correct DBMS or CICS region runs
 - the channel-extender is available
 - the mail server is available.

ThruPut Manager's Job Binding

Job Binding is a logical approach that identifies conditions or resources that the job needs to run successfully, and then ensures that the job runs when and where those conditions or resources exist. The implementation is straightforward for the datacenter and transparent to the user.

Using Binding Agents

Job Binding uses constructs called *Binding Agents*, which are defined by the installation. Each Binding Agent represents a resource or condition that can affect the routing of a job. These Binding Agents are activated on each LPAR where the resource or condition exists, either

8300 Woodbine Avenue
4th Floor
Markham ON Canada
L3R 9Y7

905.940.9404
905.940.5308 (fax)

marketing@mvssol.com
www.mvssol.com

MV'S
solutions inc.

ThruPut
Manager 6

automatically, when ThruPut Manager detects that the resource provider – say a DBMS – is started and ready for work, or by operator command. In Figure 1, the DBMS and Mail Server Binding Agents are activated on LPAR-B.

At submission, Binding Agents are added to each job automatically by ThruPut Manager through installation rules. The user need have no involvement in this process

When a job is submitted on one JESplex and needs to run on another JESplex, then “remote binding agents” direct the job to run on the second JESplex.

Manages Routing Exceptions

ThruPut Manager reconciles its understanding of the status of its agents with JES2’s knowledge of what is running, so that agents associated with a server that is no longer running are automatically deactivated. If a required Binding Agent is inactive or if a job has conflicting Binding Agents, that is when no machine exists with the combination of Binding Agents needed for the job, then the job is placed in a ThruPut Manager hold. Such jobs can be easily monitored and viewed to see why they are not running, by using operator commands to display ThruPut Manager status information. Once the Binding Agent is activated, or the combination of Binding Agents becomes available, the job is automatically released.

With ThruPut Manager’s Job Binding there need be no conflict with a System Affinity or a Scheduling Environment. These can be removed from a job by installation rules and Binding Agents can be added, all without user involvement.

If two Binding Agents are known to be incompatible, the datacenter can create a rule to take appropriate action, including failing the job with appropriate notification.

Binding Agents Maintained Over IPLs

The state of a Binding Agent is maintained in a central Control File and persists over an IPL, meaning that relatively fixed associations, such as those for restricted-license software, do not have to be re-established after an IPL.

Binding Agents Apply to TSO Access

Binding Agents also apply to TSO sessions. When TSO users attempt to access licensed software on a system which does not have the appropriate Binding Agents active, they receive a message telling them that their dependency cannot be met. An installation can customize the message they receive and inform the user which system(s) are valid.

Scheduling Environments

IBM provides a logically similar although architecturally very different approach called *Scheduling Environments*, which are built on *Resource Elements*. They do not apply to TSO sessions.

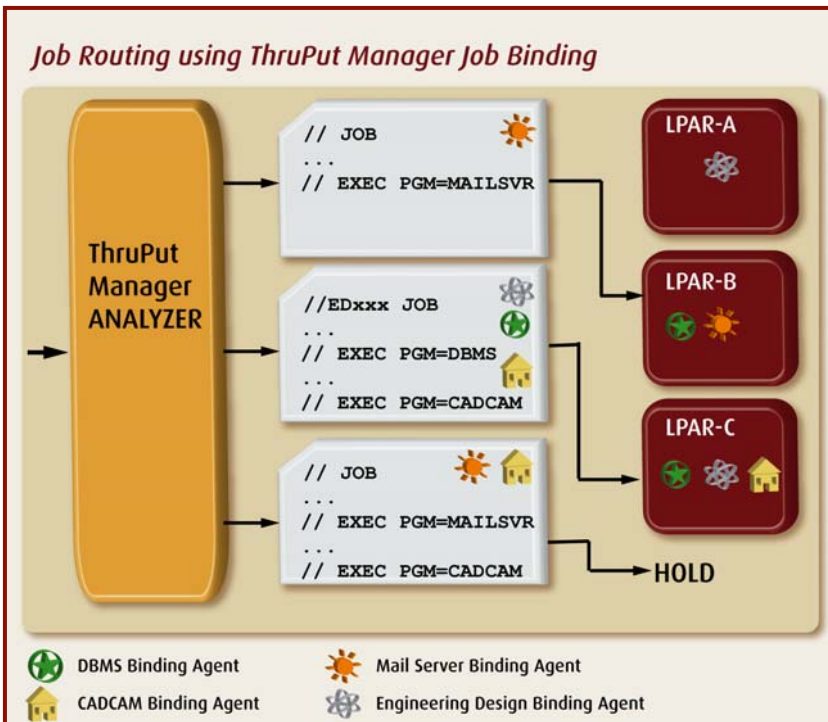


Figure 1: ThruPut Manager analyzes JCL at submission time, detects the resources needed and associates the relevant Binding Agents with the job. In this example, it detects the first job needs to run where the Mail Server is running and associates it with the Mail Server Binding Agent. The job runs on LPAR-B. From the jobname, the second job needs to run on an Engineering Design machine with both CADCAM software and the DBMS, and so runs on LPAR-C. The third job needs both the Mail Server and the CADCAM software and it is placed in a ThruPut Manager hold since no environment exists with the relevant Binding Agents active.

and is not required to make JCL changes to support it. This means that the binding requirements can always be depended on to be accurate.

At runtime, ThruPut Manager resolves the routing requirements of each job by detecting all the Binding Agents associated with a job and ensuring it runs on a system with that combination of conditions and resources.

Using Scheduling Environments

The datacenter identifies a set of *Resource Elements*, each of which represents a condition or resource. The data center staff turn Resource Elements “on” and “off”, for each LPAR. In the example there would be a Resource Element for each of the DBMS, Mail Server, CAD/CAM software, and the Engineering Design condition.

A *Scheduling Environment* is a combination of Resource Elements and defines a distinct routing purpose. A job can be associated with only one Scheduling Environment, so the installation must define a Scheduling Environment for each combination of Resource Elements that may be needed. You will need many more Scheduling Environments than the Resource Elements you started out to manage.

Defining Scheduling Environments is Cumbersome

In the example, the datacenter would define four Scheduling Environment for jobs requiring just a Mail Server, just the DBMS, just the CAD/CAM software, or just the Engineering Design condition. As well the datacenter would define Scheduling Environments for any jobs requiring a combination of either two or three Resource Elements. Finally some jobs might even need all four Resource Elements and they would need their own Scheduling Environment. Of course the data center only needs to define the combinations (of the possible 15) that the jobs could possibly require.

Resource Elements can be moved by operator commands and the jobs with the affected Scheduling Environments will follow accordingly.

Scheduling Environments are JCL Dependent

Scheduling Environments do not provide an automatic solution. A job’s Scheduling Environment is coded in JCL. Any change to the naming of Scheduling Environments, or the mapping of Resource Elements to Scheduling Environments, means the users have to be involved. This incurs the co-ordination effort of negotiating the changes

with the user community and synchronizing the changes with their schedules – not a trivial exercise.

Some datacenters mitigate the user reliance exposure by having an exit add the correct Scheduling Environment to each job.

While this solution contains

ThruPut Manager’s Job Binding mechanism is easier, safer and more powerful than home-grown exits can ever be.

the problem to the datacenter, every change to the names of the Scheduling Environments, the Resource Elements or their relationship means cracking open the code. We know that any code change, regardless of how simple or how talented the programmers, must go through a test and review cycle before being allowed to run in a production environment, much less control the production environment. Again this is not a trivial process. Such exits can be very complex, and systems Programmers with the skill to develop and maintain them are becoming rarer.

Routing Exceptions Not Managed

In the case of conflicting routing controls, the resolution mechanisms are non-existent:

- If a job has a Scheduling Environment and a System Affinity coded, they may point to different LPARs. There are no standard tools available to resolve this problem.
- When a Scheduling Environment is not available, the job is simply not scheduled. It can be difficult for operations to identify why a job is not running

How Robust Is Your Routing Mechanism?

- ☑ Can you balance your workload independent of your user’s involvement?
- ☑ Does your routing mechanism work without relying on exits?
- ☑ Can you make logical specifications without worrying what combinations jobs may require them in?
- ☑ Can you change the logical to physical mapping with
 - no JCL changes?
 - no publishing of new standards?
 - no training of new standards?
 - no implementation timetables to be coordinated?
- ☑ Can you re-route your workload with a few simple operator commands?
- ☑ Do your logical to physical mappings survive across an IPL?

Comparing ThruPut Manager's Job Binding and Scheduling Environments

Criteria	ThruPut Manager	Scheduling Environments
Independent of Job Class?	Yes	Yes
Independent of Users following JCL standards?	Yes	No, Scheduling Environment may be set in the JCL
Independent of Exits?	Yes	No, Scheduling Environment may be set in an exit
Uses indirect references?	Yes, through Binding Agents	Yes, through Resource Elements
Supports multiple indirect references?	Yes, many Binding Agents can be associated with one job	No, only one Scheduling Environment can be associated with each job
Automatically detects (software) resource availability?	Yes	No
Solution grows linearly as complexity grows?	Yes	No
Supports routing to another node?	Yes	No
Easy to change target LPAR for a workload, or to redirect workload during an outage?	Yes	Usually, though can't cross JESplex boundary
Trivial to introduce a new resource combination?	Yes	No
Supports TSO	Yes	No
Feedback when a job cannot be routed?	Yes	No
Viable as a comprehensive routing solution for your entire workload?	Yes, Job Binding was designed for this	No, Scheduling Environments are too simplistic

- When the job should execute on a different JESplex, there is no automatic mechanism to redirect.

In the example, the third job would be stuck in limbo, with no message or warning.

Scheduling Environments Do Not Persist Over an IPL

The state of Scheduling Environments is lost over an IPL and must be re-established by setting the appropriate Resource Elements to the desired state for each system. Automated operator scripts must be maintained, may be out of date and must be started after the IPL process is completed.

Conclusion

The problems inherent with Scheduling Environments make them suitable, at best, for situations involving limited routing purposes. ThruPut Manager is the superior solution for any datacenter with even modest routing needs. Since it was designed for batch, there are no workarounds or unpleasant side effects to deal with. The Job Binding architecture is a fully automated mechanism which can manage an unlimited number of routing dependencies, in a user-transparent manner. The rule language is easier, safer and more powerful than any home-grown exit can ever be. Job Binding is robust enough to be a routing mechanism for all your batch needs.

ThruPut Manager does the job simply and elegantly, allowing you to reap the benefits of a comprehensive and realistic routing scheme: hardware utilization is maximized, software licensing costs are minimized and batch throughput is optimized, all in an operational- and user-friendly manner.

This document assumes the reader is familiar with ThruPut Manager. Only certain highlights of the product have been discussed here. For further information, please contact us as noted on the first page.

ThruPut Manager is a registered trademark of MVS Solutions Inc. The names of the optional components of ThruPut Manager used in this document are trademarks of MVS Solutions Inc. Other trademarks and registered trademarks used in this document are the property of their respective owners and are to be regarded as appearing with the appropriate ™ or © symbol.

© MVS Solutions Inc. 2005. All rights reserved.

512d

