z/OS Intelligent Resource Director

- WLM LPAR CPU Management
- Dynamic Channel-path Management
- Channel Subsystem I/O Priority Queueing

Frank Kyne
Michael Ferguson
Tom Russell
Alvaro Salla
Ken Trowell

ibm.com/redbooks

Redbooks

International Technical Support Organization

z/OS Intelligent Resource Director

August 2001

SG24-5952-00

Take Note! Before using this information and the product it supports, be sure to read the general information in “Special notices” on page 401.

First Edition (August 2001)

This edition applies to Version 1 Release 1 of z/OS, Program Number 5694-A01.

Note: This book is based on a pre-GA version of a product and may not apply when the product becomes generally available. We recommend that you consult the product documentation or follow-on versions of this redbook for more current information.

Comments may be addressed to:
IBM Corporation, International Technical Support Organization
Dept. HYJ Mail Station P099
2455 South Road
Poughkeepsie, NY 12601-5400

When you send information to IBM, you grant IBM a non-exclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

Preface

This IBM Redbook describes the new LPAR Clustering technology, available on the IBM @server zSeries processors, and z/OS. The book is broken into three parts:

- Dynamic CHIPD Management
- I/O Priority Queueing
- CPU Management

Each part has an introduction to the new function, planning information to help you assess and implement the function, and management information to help you monitor, control, and tune the function in your environment.

The book is intended for System Programmers, Capacity Planners, and Configuration Specialists and provides all the information you require to ensure a speedy and successful implementation of the functions at your installation.

The team that wrote this redbook

This redbook was produced by a team of specialists from around the world working at the International Technical Support Organization Poughkeepsie Center.

Frank Kyne is a Senior I/T Specialist at the International Technical SupportOrganization, Poughkeepsie Center. He has been an author of a number of other Parallel Sysplex redbooks. Before joining the ITSO three years ago, Frank worked in IBM Global Services in Ireland as an MVS Systems Programmer.

Michael Ferguson is a Senior I/T Specialist in the IBM Support Centre in Australia. He has 14 years of experience in the OS/390 software field. His areas of expertise include Parallel Sysplex, OS/390, and OPC.

Tom Russell is a Consulting Systems Engineer in Canada. He has 30 years of experience in IBM, supporting MVS and OS/390. During a previous assignment at the ITSO Poughkeepsie, he wrote numerous books on Parallel Sysplex implementation and performance, continuous availability, and the OS/390

Workload Manager. His areas of expertise include online systems design, continuous availability, hardware and software performance, and Parallel Sysplex implementation. Tom holds a degree in Mechanical Engineering from the University of Waterloo.

Alvaro Salla is an independent IT Consultant in Brazil. Prior to this position, Alvaro worked for IBM Brazil for 31 years, involved with S/390. Alvaro has had assignments in Poughkeepsie and in the European education center in La Hulpe. During this time, he has been involved in many residencies and has authored many redbooks.

Thanks to the following people for their contributions to this project:

Bob Haimowitz
International Technical Support Organization, Poughkeepsie Center

Stephen Anania
IBM Poughkeepsie

John Betz
IBM Poughkeepsie

Friedrich Beichter
IBM Germany

Dick Cwiakala
IBM Poughkeepsie

Manfred Gnirss
IBM Germany

Steve Grabarits
IBM Poughkeepsie

Jeff Kubala
IBM Poughkeepsie

Juergen Maergner
IBM Germany

Kenneth Oakes
IBM Poughkeepsie

Bill Rooney
IBM Poughkeepsie

Ruediger Schaeffer
IBM Germany

Hiren Shah
IBM Poughkeepsie

Charlie Shapley
IBM Poughkeepsie

John Staubi
IBM Poughkeepsie

Kenneth Trowell
IBM Poughkeepsie

Gail Whistance
IBM Poughkeepsie

Peter Yocom
IBM Poughkeepsie

Harry Yudenfriend
IBM Poughkeepsie

Special notice

This redbook is intended to help systems programmers and configuration planners to plan for and implement z/OS Intelligent Resource Director. The information in this publication is not intended as the specification of any programming interfaces that are provided by z/OS. See the PUBLICATIONS section of the IBM Programming Announcement for z/OS for more information about what publications are considered to be product documentation.

IBM trademarks

The following terms are trademarks of the International Business Machines Corporation in the United States and/or other countries:

e (logo)® 
IBM ®
CICS
DB2
DFSMShsm
Enterprise Storage Server
Enterprise Systems Architecture/390
ES/9000
ESCON
FICON
MQSeries
MVS/ESA
MVS/XA
OS/390
Parallel Sysplex
PR/SM
RACF
RAMAC
Redbooks
Redbooks Logo 
RMF
S/390
Sysplex Timer
400
Lotus
VM/ESA
VTAM
z/Architecture
z/OS

Comments welcome

Your comments are important to us!

We want our IBM Redbooks to be as helpful as possible. Send us your comments about this or other Redbooks in one of the following ways:

- Use the online **Contact us** review redbook form found at:

  `ibm.com/redbooks`

- Send your comments in an Internet note to:

  `redbook@us.ibm.com`

- Mail your comments to the address on page ii.

Part 1

Introduction to Intelligent Resource Director

1

Introduction to Intelligent Resource Director (IRD)

This IBM Redbook provides the information you require to evaluate, plan for, implement, and manage the new functions known collectively as Intelligent Resource Director.

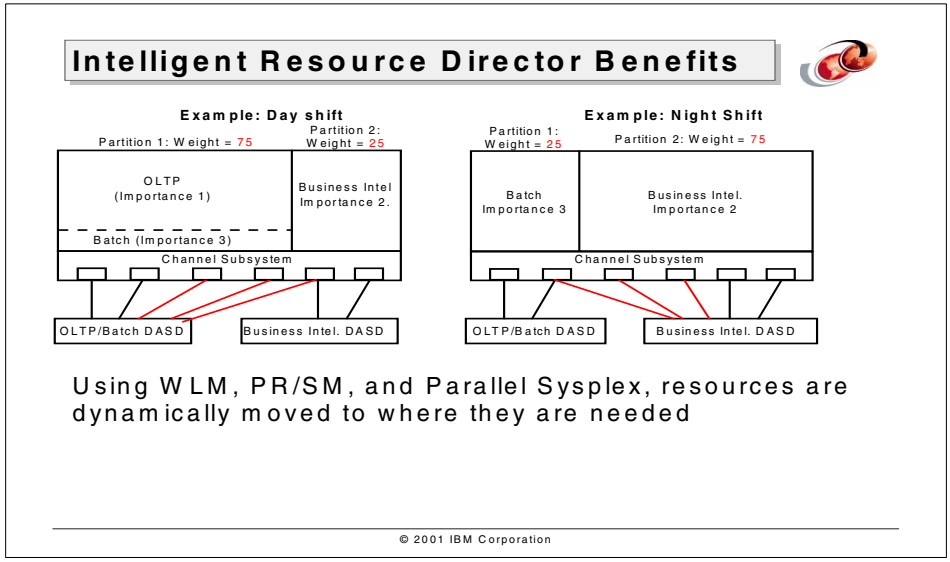
Intelligent Resource Director was announced on October 3, 2000 as one of the new capabilities available on the IBM zSeries range of processors and delivered as part of z/OS.

Intelligent Resource Director might be viewed as Stage 2 of Parallel Sysplex. Stage 1 provided facilities to let you share your data and workload across multiple system images. As a result, applications that supported data sharing could potentially run on any system in the sysplex, thus allowing you to move your workload to where the processing resources were available.

However, not all applications support data sharing, and there are many applications that have not been migrated to data sharing for various reasons. For these applications, IBM has provided Intelligent Resource Director, which basically gives you the ability to move the resource to where the workload is.

Intelligent Resource Director uses facilities in z/OS Workload Manager (WLM), Parallel Sysplex, and PR/SM to help you derive greater value from your z/Series investment. Compared to other platforms, z/OS with WLM already provides benefits from the ability to drive a processor at 100% while still providing acceptable response times for your critical applications. Intelligent Resource Director amplifies this advantage by helping you make sure that all those resources are being utilized by the right workloads, even if the workloads exist in different Logical Partitions (LPs).

The following figure contains a simplified view of what Intelligent Resource Director does for you. The figure shows a Central Processing Complex (CPC) with two LPs. One LP contains an OLTP workload, defined in WLM as being Importance 1, and a batch workload, defined in WLM as being Importance 3. The other LP contains a Business Intelligence workload that is defined to WLM as being Importance 2. Both the batch and the Business Intelligence workloads are capable of using the capacity of the whole CPC if allowed to. To provide the OLTP workload with the resources it requires to meet its goals during the prime shift, Intelligent Resource Director sets the LPAR weight of that LP to 75. The weight of the Business Intelligence LP is set at 25. However, in the evening shift when the OLTP workload has gone away, Intelligent Resource Director will adjust the weights so that the Business Intelligence LP, which is of higher importance



than batch, now gets 75% of the CPC, and the LP containing the batch workload now gets 25%. You will also notice that during the prime shift the OLTP DASD have more channels, whereas in the evening, there are more paths to the Business Intelligence DASD. Intelligent Resource Director has also automatically adjusted the channel configuration to provide more paths to the DASD subsystem serving the more important workload.

Intelligent Resource Director is not actually a product or a system component; rather it is three separate but mutually supportive functions:

- WLM LPAR CPU Management
- Dynamic Channel-path Management (DCM)
- Channel Subsystem I/O Priority Queueing (CSS IOPQ)

This book contains three parts, one for each of these three functions. For each function we provide an introduction, detailed information about how it works, planning information, implementation steps, operational considerations, and some recommendations about monitoring and tuning.

Intelligent Resource Director is implemented by new functions in:

- z/OS (in z/Architecture mode)
- Workload Manager (WLM)
- IBM zSeries 900 and later CPCs

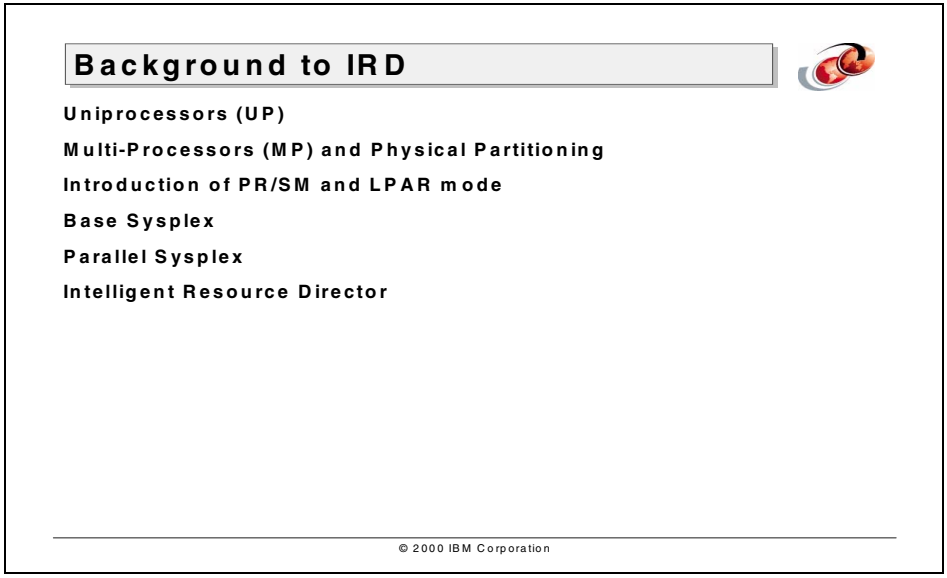
and by using existing function in the following system components:

- Hardware Configuration Dialog (HCD)
- Dynamic I/O Reconfiguration
- I/O Supervisor (IOS)

In this chapter, we talk about Intelligent Resource Director in general and discuss why IBM has developed this capability, and in what ways it can benefit your installation. In the subsequent chapters we discuss each of the three functions in detail. Depending on which of these functions you wish to exploit, each of these parts can be read in isolation. To enable this, you will find that there is a small amount of duplication in different parts of the book: we hope this approach doesn't detract from the readability of this document.

Note: All of the IRD functions require z/OS to be running in z/Architecture mode. In this book, any time we mention z/OS, we always mean z/OS running in z/Architecture mode, unless otherwise specifically noted.

1.1 S/390 - A history lesson



When S/360 was first announced, the available hardware at the time was a single CP processor (we called them CPUs back in those days!), containing less storage and fewer MIPS than the cheapest pocket calculator available today. It was also considerably larger and more expensive! As the business world discovered more and more uses for this new tool, the demand for MIPS outpaced the rate at which CP speed was progressing.

As a result, IBM introduced the ability to add a second CP to the processor. This provided more power, and potentially more availability, since you could conceptually continue processing even if one of your CPs failed. These machines were available either as APs (Attached Processors, where only one CP had an I/O subsystem) and MPs (Multi Processors, where each CP had access to its own I/O subsystem).

In addition to providing more capacity on an MP, the processors, I/O channels, and storage could be physically “partitioned”, meaning that two separate copies of the operating system could be run on the processors, if you so desired. In the days when hardware and software were much less reliable than we are used to today, this provided significant availability benefits because you could now divide your production system in half, and if a system failed, only half your applications would be lost—a major attraction at the time!

The next major hardware advance, in terms of flexibility for running multiple copies of the operating system, was Processor Resource/System Manager (PR/SM) with the LPAR feature, introduced on the IBM 3090 range of processors. PR/SM gave you the ability, even on a CPC with just one CP, to run up to four Logical Partitions (LPs). This meant that you could split your production applications across several system images, maybe have a separate development system, or even a systems programmer's test system, all on a CPC with just a single CP. Such a configuration didn't do much to protect you from a CP failure (if you had just one CP), but it did do a lot to protect you from software failures. It also gave you the ability to create a test environment at a lower cost, thus giving you the capability to ensure that all software was tested before your production applications were run on it.

All the enhancements to this stage were aimed at giving you the ability to break a single large system into a number of smaller, independent system images. However, as applications grew and system management became a concern, some mechanism to provide closer communication with, and control of, the systems was required. To address this need, MVS/ESA Version 4.1 introduced the concept of a "sysplex" (now called a Base sysplex to differentiate it from a Parallel Sysplex). This provided a new MVS component known as the Cross System Coupling Facility (XCF), which allows applications running on multiple images to work more cooperatively without having to suffer a significant overhead or complex programming. An example of an application that exploited this capability is CICS MRO, where CICS regions running on multiple systems in the sysplex can communicate using XCF, thereby providing significantly better performance than with the previous option of using VTAM to communicate with each other. MVS/ESA 4.1 also introduced support for the ability to have a single time source for all the systems in the sysplex (the Sysplex Timer). This laid the foundation for data sharing, which was introduced by the next significant advance: Parallel Sysplex, introduced with MVS/ESA Version 5.1.

Parallel Sysplex provides the single image infrastructure to have multisystem data sharing with integrity, availability, and scalability not possible with earlier data sharing mechanisms. All these benefits are enabled by the introduction of a new external processor/memory known as a Coupling Facility (CF). Coupling Facilities were initially run on dedicated processors (9674s) and have since been enhanced to run in special LPs on the general purpose 9672s and, more recently, the z900 processors as well. There have been many other enhancements to Parallel Sysplex, including, for example, the ability to duplex the DB2 Group Buffer Pools, providing even higher availability and flexibility for those structures.

And that brings us to the present day, with the announcement of the zSeries processors and the z/OS operating system, and the subject of this book: Intelligent Resource Director.

1.2 Why Intelligent Resource Director is the next step

If you look at a typical medium-to-large S/390 configuration, you have a variety of processor types and sizes, generally operating in LPAR mode, and supporting images that run batch, OLTP, Web servers, application development, Enterprise Resource Planning (such as SAP R/3), Business Intelligence, and various other workloads. Within each LP, WLM in Goal mode is responsible for allocating resources such that it helps the most important workloads (as specified by the installation) meet their Service Level Agreement objectives. WLM has been providing this capability since MVS/ESA 5.1, and is generally considered to be very effective at this task.

Moving up a level, you have PR/SM Licensed Internal Code (LIC) with responsibility for allocating the physical CPU resource, based upon an installation-specified set of *weights* for the LPs.

So, we have WLM managing the allocations of the resources that are given to the LP by PR/SM, and we have PR/SM dividing up processing resources among the LPs on the processor. Would it not make sense to have some communication between WLM and PR/SM? WLM knows the relative importance of the work running in each LP, and is ideally placed to decide what the weight of each LP should be, so that PR/SM will give the CPU to whichever LP needs it the most in order to help meet the goals of the business; this is one of the functions delivered by Intelligent Resource Director.

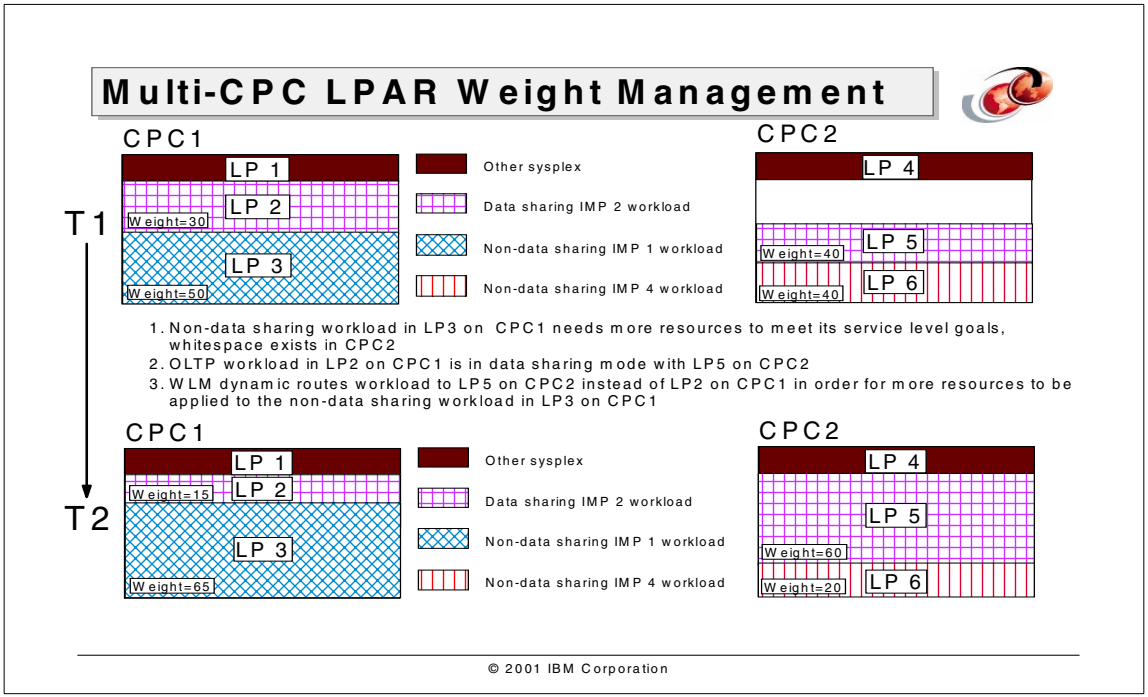
The ongoing value of this depends on the continued use of LPAR mode, so what is the future for LPAR mode? We expect the use of PR/SM to increase rapidly from the already high level of use at the moment, for the following reasons:

- For many customers, the capacity of the latest processors is growing at a faster pace than their workloads. So as time goes on, it becomes more and more feasible to consolidate existing systems onto a smaller number of larger processors.
- One of the reasons that installations currently do not consolidate onto fewer, larger processors is the impact this would have on their software charges. At the moment, you might have DB2 on one processor, IMS on another, and CICS/VSAM on a third. If you were to integrate them onto one large 9672-class processor, you would have to pay licenses for those products based on the total capacity of the new processor. However, the software licensing changes introduced with z/OS and the z900 processors mean that you now pay software license fees based on the size of the LP the software is running in, rather than on the total capacity of the processor. It therefore becomes more feasible to consolidate onto a smaller number of larger processors.

- The continuing trend toward server consolidation, and the ability to run Linux LPs on the z900 and 9672s, mean that the number of LPs being created to handle these consolidated workloads will continue to increase.
- When PR/SM was first introduced, the maximum number of LPs on a processor was 4. It was then increased to 7, then 10, and is currently 15. It is not unreasonable to expect that, as time goes on, this number will continue to increase, especially as the total capacity of the CPC continues to increase.
- The need for large amounts of “white space” to handle the large workload spikes that are representative of the e-business environment is better handled on a small number of larger processors. For example, if you currently have 5 x 200 MIPS processors, each running about 85% busy, you have 30 MIPS of spare capacity on each processor to handle these unexpected spikes. If you consolidated those 5 processors onto a single 1000 MIPS z900, you would now have 150 MIPS of white space that is available to each LP should it require that capacity when unexpected spikes occur.

OK, you say, now that you have the ability to move resource to where the work is, does this mean that you should forget about that data sharing project you had in mind for next year? Not at all. First of all, data sharing is more about application availability than capacity. No matter how good Intelligent Resource Director is, it is not going to protect a non-data sharing application from an outage of its database manager or the operating system that it is running under. In addition, data sharing (and its associated ability to do workload balancing) and Intelligent Resource Director actually work very well together. The ability to distribute a workload across multiple processors means that a non-data sharing application that is sharing a processor with a data sharing workload can be given additional CPU resource as the data sharing workload is shifted to another physical processor.

This is shown in the figure on the following page, where LP3 on CPC1 contains an important non-data sharing workload (WLM Importance 1) that is constrained by the current weight of the LP (50%). LP2 on CPC1 contains an Importance 2 data sharing workload. This workload is also CPU-constrained, and is using all the CPU (30%) guaranteed to it by its weight. LP1 is running another workload from another sysplex, and is also using all the CPU guaranteed by its weight (20%). CPC1 is therefore 100% busy. On CPC2, LP6 has a weight of 40, is running an importance 4 workload, and is not using all the CPU guaranteed by its weight. LP5 is running the same data sharing workload as LP2 on CPC1. It has a weight of 40, and is currently not using all the CPU guaranteed by its weight. Finally, LP4 is running work from another sysplex, and is using all the CPU guaranteed by its weight (20%).



Prior to the introduction of Intelligent Resource Director, this is more or less how the environment would have remained. The importance 1 workload in LP3 on CPC1 would continue to miss its goal. The importance 2 workload in LP2 on CPC1 would also continue to miss its goal, while the importance 4 workload in LP6 on CPC2 would exceed its goal.

If we introduce Intelligent Resource Director into this environment, the first thing it will do is take some weight from LP2 on CPC1 and give it to LP3. Even though LP2 is missing its goal, WLM will try to help the importance 1 workload over the importance 2 workload. The effect of this change of weights is to give even less CPU to LP2. Fortunately, the workload in LP2 is a data sharing workload, so when it has so little capacity on CPC1, the work tends to migrate to LP5 on CPC2. As LP5 on CPC2 gets busier, WLM in LP5 increases its weight, at the expense of the importance 4 workload in LP6 on CPC2.

The net effect of all these changes is that the Importance 1 work in LP3 will get the CPU capacity it requires to meet its goal: it increased from 50% to 65% of the capacity of CPC1. Similarly, the data sharing workload in LP2 and LP5 will get additional CPU, meaning that it will now achieve its goal. Overall, you are not only driving your CPCs harder; more importantly, the CPCs are spending more time processing the most important workloads.

This discussion just gives you a glimpse at the flexibility and new function provided by just one of the components of Intelligent Resource Director. In the remainder of this book, we talk about each of the three components of Intelligent Resource Director, help you identify the value of that component in your environment, and then help you implement and manage it.

Part 2

WLM LPAR CPU Management

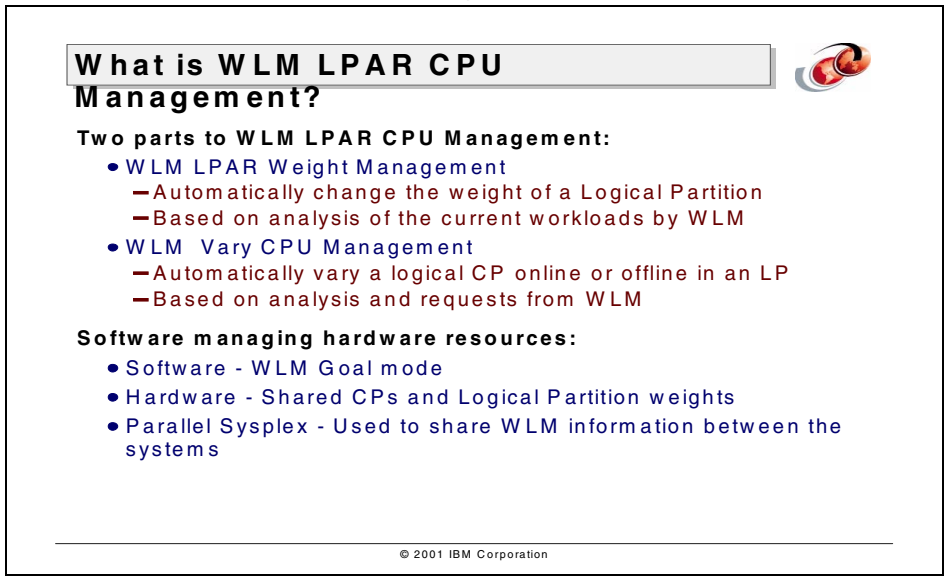
2

Introduction to WLM LPAR CPU Management

WLM LPAR CPU Management is a new capability provided by z/OS. It is available on IBM zSeries 900 and later CPCs when the z/OS LP is running in WLM Goal mode. It is one of the three components of Intelligent Resource Director.

In this chapter, we provide an introduction to this new capability, including information you need to decide if WLM LPAR CPU Management is appropriate for your environment. If you determine that it is, the subsequent chapters in this part provide the information to plan for, implement, and manage it. We recommend that these chapters be read sequentially; however, it should be possible to read each chapter independently should you decide to do so.

2.1 What WLM LPAR CPU Management is



WLM LPAR CPU Management is implemented by z/OS Workload Manager (WLM) Goal mode and IBM zSeries 900 PR/SM LPAR scheduler Licensed Internal Code (LIC).

WLM LPAR CPU Management, as the above chart shows, actually consists of two separate, but complementary, functions:

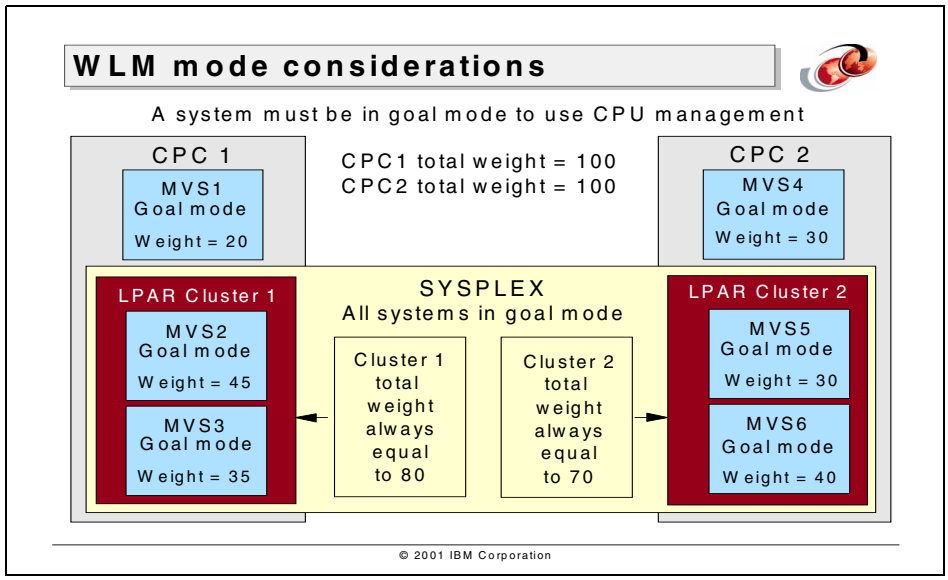
WLM LPAR Weight Management, whose role is to dynamically change the LPAR weight of logical partitions to help a workload that is missing its goal.

WLM LPAR Vary CPU Management, whose role is to dynamically change the number of online logical CPs in a logical partition (LP), to bring the number of logical CPs in line with the capacity required by the LP.

Both functions require that the systems are running in WLM Goal mode. It is also necessary for the systems to be in a Parallel Sysplex, in order to share the WLM information between the systems.

In order to effectively implement WLM LPAR CPU Management, it is important to understand how both WLM Goal mode and LPAR mode work, so the next few pages provide a brief overview of these components.

2.2 Workload Manager advantages

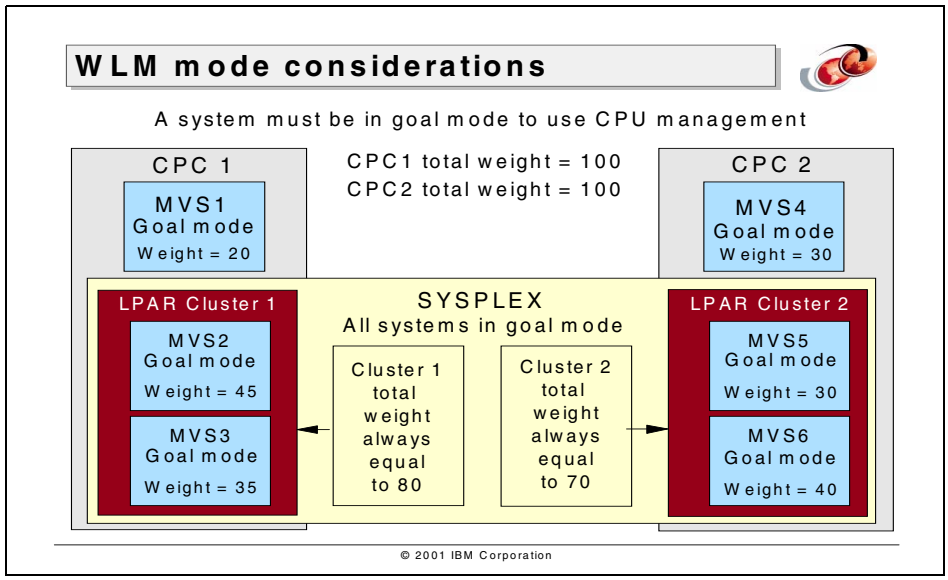


WLM is a z/OS component (it was actually introduced in MVS/ESA V5) responsible for managing the system resources in such a way that the workloads identified as being the most important will achieve their objectives. In fact, WLM is present and responsible for certain tasks, even if the system is in WLM Compatibility mode. In Goal mode, WLM provides the following advantages, when compared to Compatibility mode:

- Simplicity, since goals are assigned in the same terms as in existing Service Level Agreements (instead of having to assign relative dispatching priorities in the IEAIPSxx), and the use of an ISPF/TSO application to define such goals.
- It is more closely linked to the business's needs. Workloads are assigned *goals* (for example, a target average response time) and an *importance*. Importance represents how important it is to the business that that workload meets is goals. In Compatibility mode, all workloads are defined in terms of relative dispatching priority to the other workloads—and these dispatching priorities do not necessarily reflect the business importance of the associated workloads. Also, relative dispatching priorities mean that a workload may keep getting resource whenever it requests it, even if it is over-achieving its goal while other workloads with a lower dispatching priority are missing their goals.

- WLM recognizes new transaction types, such as CICS, IMS DC, DDF, IWEB, Unix System Services (USS), DB2 parallel query, MQSeries, APPC and so on, allowing reporting and goal assignment for all of these workload types.
- It is particularly well suited to a sysplex environment (either basic or Parallel) because WLM in Goal mode has knowledge of the system utilization and workload goal achievement across all the systems in the sysplex. *This cross-system knowledge and control has a much more significant impact in an IRD environment.*
- It provides much better RMF reports, which are closely aligned to the workloads and their specified goals. For example, if you defined a goal that 80% of transactions should complete in 1 second, RMF will report the actual percent of transactions that completed within the target time. The reports also include CICS and IMS internal performance data that is not available from RMF in WLM Compatibility mode.
- Averaged over a full day, WLM Goal mode will generally provide better performance than Compatibility mode. This is because WLM in Goal mode is constantly adjusting to meet the goals of the *current* workload, whereas the IEAIPSxx in Compatibility mode is usually designed for a particular workload mix, but is less effective when the workload mix changes, as it usually does over the course of a day.
- It provides dynamic control of the number of server address spaces, such as:
  - Batch Initiators
  - HTTP servers
- WLM plays a central role in dynamic workload balancing among OS/390 and z/OS images in a Parallel Sysplex with data sharing. WLM Goal mode works with VTAM Generic Resources, Sysplex Distributor for TCP/IP, CICS MRO, and IMS Shared Message Queues to route transactions to the most appropriate system.
- WLM provides more effective use of the Parallel Access Volume feature of the IBM 2105 ESS.
- It provides the ability to decide which image a batch job can run in based on the availability of a real or abstract resource (using the WLM Scheduling Environment feature).
- It provides the possibility of specifying both a minimum amount of CPU resource that a Service Class Period is guaranteed if it needs it, and a maximum amount of CPU resource that a Service Class Period can consume.

2.3 Workload Manager highlights



When operating in Goal mode, WLM has two sets of routines: those that attempt to achieve transaction goals (these strive to make the system as responsive as possible), and those that attempt to maximize the utilization of the resources in the environment (these aim for maximum efficiency).

- Enforcing transaction goals consists of the following:
  - During the Policy Adjustment routines, WLM may change the priority of given tasks, including the dispatching priority, the weight of an LPAR (a WLM LPAR CPU Management function), the number of aliases for a given 2105 device, and the specification of a channel subsystem I/O priority.

    This function employs a donor/receiver approach, where an important workload that is missing its goal will receive additional resources—resources that are taken away from other workloads that are over-achieving their targets or workloads and that are less important (as defined by the installation). One of the objectives of this function is that the workloads within a given importance level will all have a similar Performance Index (PI) (a measure of how closely the workload is meeting its defined goal).
  - Server Address Space (AS) creation and destruction (AS Server Management routines).

  - Providing information to the dynamic workload balancing functions, like VTAM Generic Resources or Sysplex Distributor, to help them decide on the best place to run a transaction.

- Resource adjustment routines

  These are designed to maximize the throughput and efficiency of the system. An example would be the new WLM LPAR Vary CPU Management function, which will vary logical CPs on- and off-line in an attempt to balance required capacity with LPAR overhead.

There are several types of goals, such as:

- Average response time (1 second, for example)
- Percentile response time (80% of the transactions with response time less than 2 seconds)
- Execution Velocity, which is a measure of the amount of time the workload is delayed waiting for a resource that WLM controls

To provide this information, WLM tracks transaction delays, such as:

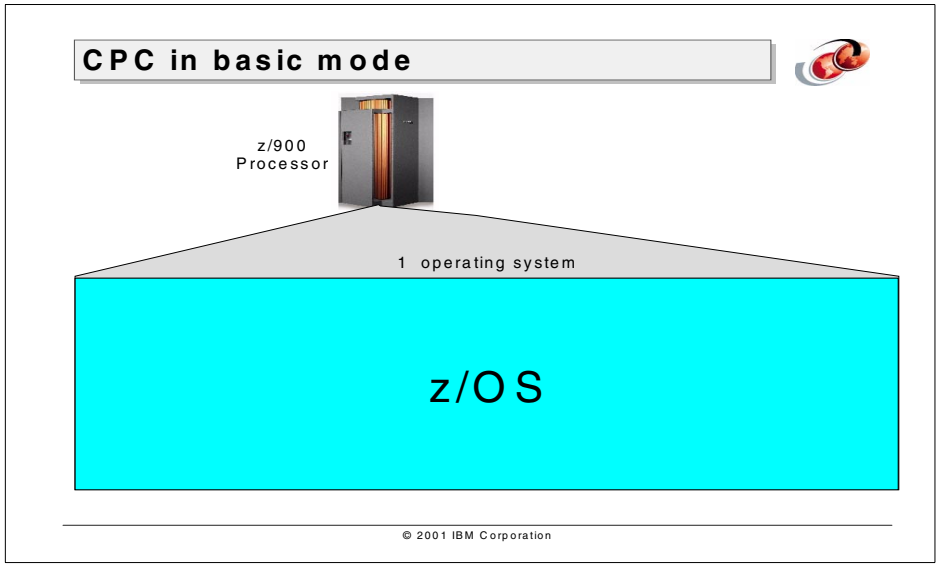
- CPU delay
- Storage delay (page faults and swapping)
- I/O delay
- AS Server queue delay

The following are some reasons why an important goal might not be reached and how WLM adjusts the way it manages system resources to compensate:

- CPU delay - the dispatching priority or LPAR weights are increased.
- Storage delay - storage isolation figures or swapping target multiprogramming level (TMPL) or system think time are raised.
- I/O delay - the I/O priority is increased in the UCB, channel subsystem and 2105 control unit queues, or an additional alias may be assigned to the device for devices that support Parallel Access Volumes.
- AS Server queue delay - a new server address space is created.

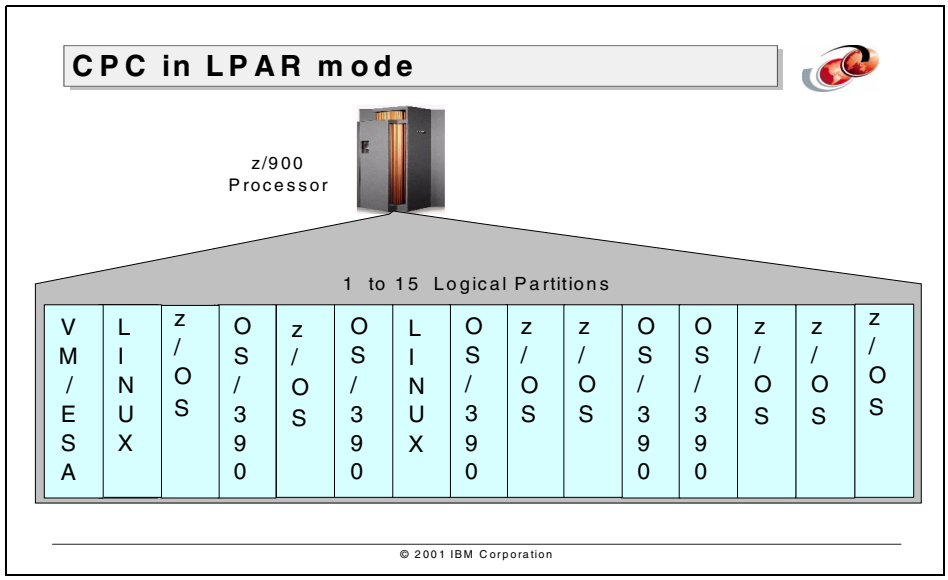
For the sake of clarity, we wish to point out that the System Resource Manager (SRM) component of z/OS still exists, regardless of whether the system is running in Goal or Compatibility mode. However, for the sake of simplicity, we do not differentiate between the two in this book. While a particular action may be carried out by either SRM or WLM, we always use the term WLM.

2.4 LPAR concepts I



When a CPC is in basic mode, all the CPC resources (CPs, storage, and channels) are available to the one operating system. All the physical CPs are used in dedicated mode for the one operating system. Any excess CP resource is wasted since no other system has access to it. There are still situations where CPCs are run in basic mode (for example, if the z/OS system needs to use the entire capacity of the CPC), however, because of the huge capacity of modern CPCs, this is getting less and less common.

2.5 LPAR concepts II



Processor Resource/System Manager (PR/SM) is a standard feature of all S/390 (including 9672 and ES/9000) and z900 CPCs. It consists of two capabilities:

- Multi High Performance Guest Support (MHPGS) in VM

  This allows several preferred guests under VM/ESA (one of them using V=R storage), all with performance very close to that available when running native (that is, not under VM).

- Logical Partitioning (LPAR)

  This allows a CPC to be divided into multiple logical partitions. This capability was designed to assist you in isolating workloads in different z/OS images, so you can run production work separately from test work, or even consolidate multiple servers into a single processor.

LPAR has the following properties:

- Each LP is a set of physical resources (CPU, storage, and channels) controlled by just one independent image of an operating system, such as: z/OS, OS/390, Linux, CFCC, VM, or VSE.
- You can have up to 15 LPs in a CPC.
- Each LP is defined through IOCP/HCD. For example, the IOCP RESOURCE PARTITION = ((LP1,1),(LP2,2)) statement defines two LPs. A Power-on-Reset (POR) operation is required to add or remove LPs.

- LP options, such as the number of logical CPs, the LP weight, whether LPAR capping is to be used for this LP, the LP storage size (and division between central storage and expanded storage), security, and other LP characteristics are defined in the Activation Profiles on the HMC.
- Individual physical CPs can be shared between multiple LPs, or they can be dedicated for use by a single LP.
- Channels can be dedicated, reconfigurable (dedicated to one LP, but able to be switched manually between LPs), or shared (if ESCON or FICON).
- The processor storage used by an LP is dedicated, but can be reconfigured from one LP to another with prior planning.
- Although it is not strictly accurate, most people use the terms LPAR and PR/SM interchangeably. Similarly, many people use the term LPAR when referring to an individual Logical Partition. However, the term LP is technically more accurate.

Physical CPs can be dedicated or shared. If dedicated, the physical CP is permanently assigned to a logical CP of just one LP. The advantage of this is less LPAR overhead. An operating system running on a CPC in basic mode gets marginally better performance than the same CPC running OS/390 as a single LP with dedicated CPs. This is because even with dedicated CPs, LPAR still gets called whenever the LP performs certain operations (such as setting the TOD clock).

If you share CPs between LPs rather than dedicating them to a single LP, there is more LPAR overhead. The LPAR overhead increases in line with the proportion of logical CPs defined in all the active LPs to the number of shared physical CPs.

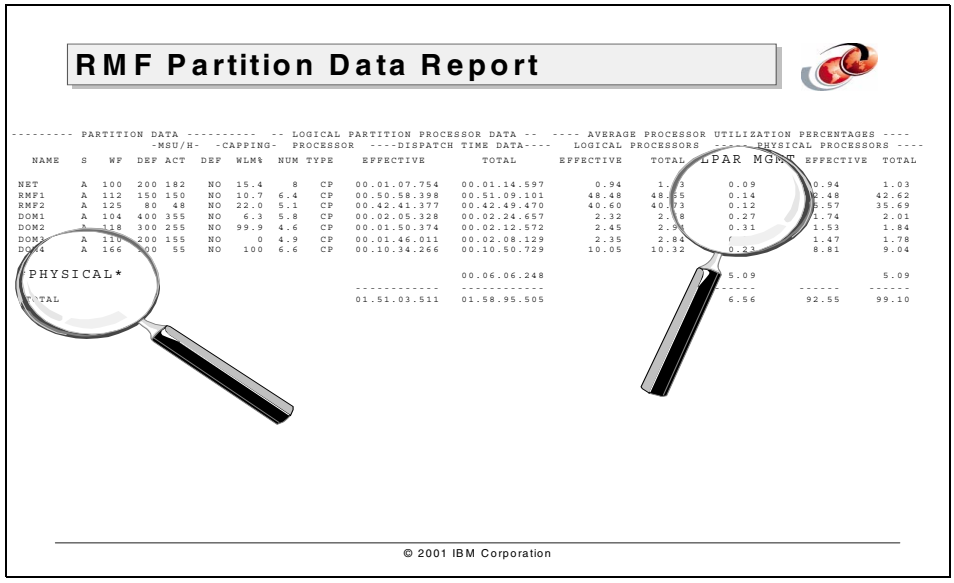
IBM has a tool (LPARCE) which estimates the overall LPAR overhead for various configurations. Your IBM marketing representative can work with you to identify the projected overhead of various configurations and workload mixes. If you are already in LPAR mode, RMF reports this overhead in the LPAR Activity report.

While the use of shared CPs does cause more overhead, this overhead is nearly always more than offset by the ability to have one LP utilize CP capacity that is not required by a sharing LP. Normally, when an operating system that is using a shared CP goes into a wait, it releases the physical CPs, which can then be used by another LP. There are a number of controls available that let you control the distribution of shared CPs between LPs.

It is not possible to have a single LP use both shared and dedicated CPs, with the exception of an LP defined as a Coupling Facility (CF).

One of the significant drivers of the increased number of LPs is server consolidation, where different workloads spread across many small machines may be consolidated in LPs of a larger CPC. LPAR also continues to be used to run different environments such as Systems Programmer test, development, quality assurance, and production in the same CPC.

The RMF Partition Data report in the following figure contains the LPAR overhead. The RMF interval is 15 minutes and there are eight physical CPs in the CPC (not shown in this piece of the report). At lower CPC utilizations, LPAR gets called more frequently, and therefore will appear to have a higher overhead; this is known as the Low Utilization Effect (LUE). As the CPC gets busier, and logical CPs consume more of their allotted time slices, LPAR gets called less frequently, and the LPAR overhead will decrease.



The LPAR overhead has three components:

- Time that LPAR LIC was working for the specific LP

  An example of this time would be preparing the LP logical CP to be dispatched on the physical CP, or emulating functions required by the operating system that are not allowed in LPAR mode (for example, changing the contents of the TOD clock). In reality it is not an overhead, it is very productive time: if you were not sharing the CP, it would sit idle until the owning LP was ready to do some more work. RMF shows this value (for each LP) under the column LPAR MGMT. For example, for LP RMF1, the total

percent of the time that the physical CPs were assigned to this LP was 42.62%, and the amount of time that the CPs were actually doing work for the operating system in that LP was 42.48%, meaning that .14% of the time was spent in LPAR overhead.

- Time that LPAR LIC is working for the collection of LPs

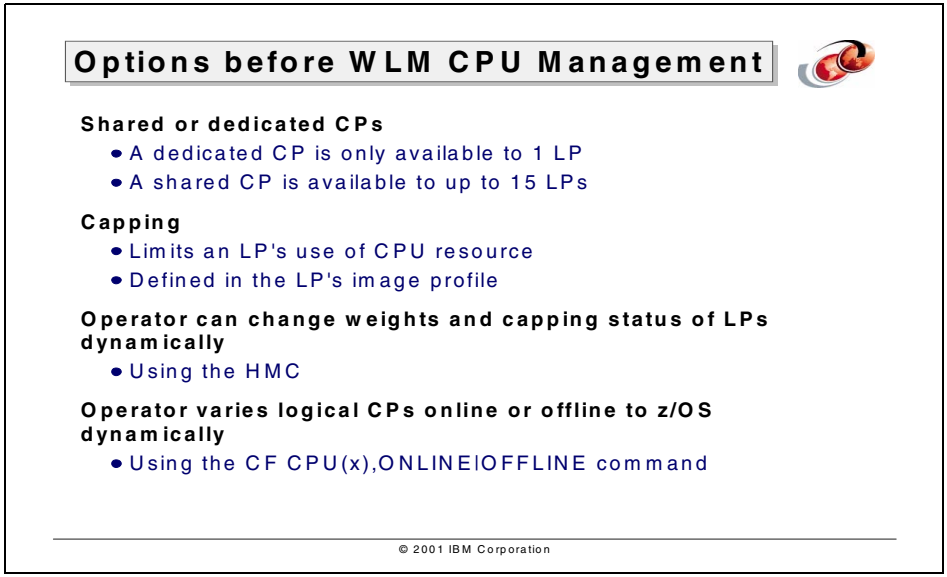
  An example of this time would be LPAR LIC looking for a logical CP to be dispatched. RMF shows this figure globally in the line *PHYSICAL*, under the column LPAR MGMT. In our example, 5.09% of the time was spent in this processing. This number is higher than would normally be considered acceptable; however, the ratio of logical to physical CPs in this configuration was 5 to 1—higher than would be recommended for most production environments. As CPU utilization increases, you would normally expect this number to decrease because LPAR will not get called as frequently.

- The overhead of being in interpretive mode

  It is not shown by RMF and tends to be very small. This overhead exists even in a dedicate LP because the dedicated logical CP runs in interpretive mode. You can measure this time by running your system in basic mode and then in LPAR mode with dedicated CPUs, and comparing the CPU time for a given job.

In our example above, the total measured LPAR “overhead” was 6.56%. This includes the overhead in each LP as well as the global overhead. There is no Rule-Of-Thumb (ROT) for this figure; however, as a general guideline you should start to take a look when it reaches 3% and to worry when it exceeds 7%. In our example, WLM LPAR Vary CPU Management could be used to help reduce this number.

2.6 Options prior to WLM LPAR CPU Management



This shows the options available for controlling the distribution and allocation of CPU resources in an LPAR environment prior to the availability of WLM LPAR CPU Management.

The major decision is whether an LP is going to use shared or dedicated CPs. If you wish to change the type of CPs being used, you must deactivate and reactivate the LP.

For a shared LP (all logical CPs are sharing a set of physical CPs), the *LP weight* plays a key role in controlling the distribution of CP resources between the LPs. The weight specifies a guaranteed amount of CP resource that this LP will get if required. However, if no other LP is using the remaining CP capacity, the LP can consume CP above and beyond the amount guaranteed by its weight.

The operator can use the HMC to change the weight of an LP dynamically, without the need to deactivate the LP.

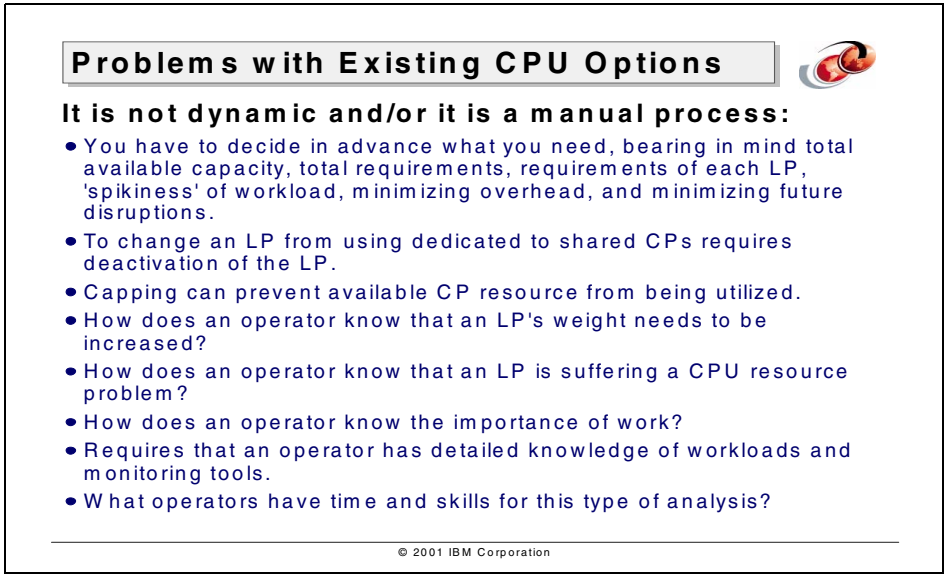
By *capping* an LP, the installation declares that the guarantee established by the weight is also used as a limitation. As a consequence, the logical CPs in the LP cannot exceed the quota determined by its weight, even if there is available CP. You can dynamically enable and disable capping for an active LP without disrupting the LP.

In an environment with an IBM zSeries 900 and z/OS, there are three different types of capping:

- LPAR Capping, as we have just described.
- WLM Goal mode capping (also called resource group capping), where in a given z/OS image (in basic or LPAR mode) WLM can optionally cap some workloads. WLM Goal mode capping has two advantages when compared with LPAR capping:
  - You can specify separate minimum and maximum limits for the workload. With LPAR capping, there is only a single value which acts as both a guaranteed minimum and a capped maximum.
  - The workload granularity. With WLM capping you can cap an individual workload within a z/OS image, whereas with LPAR capping you cap the whole z/OS image.
- Soft-capping, implemented by WLM, which dynamically enables LPAR capping for an LP and is used together with IBM License Manager to limit the capacity available to an LP. This is described in 2.15, “Relationship to IBM License Manager” on page 42.

It is not a trivial task to estimate the “correct” number of logical CPs in an LP. This is discussed in more detail in 3.9, “WLM LPAR Vary CPU Management” on page 80.

2.7 Problems with existing CPU management options



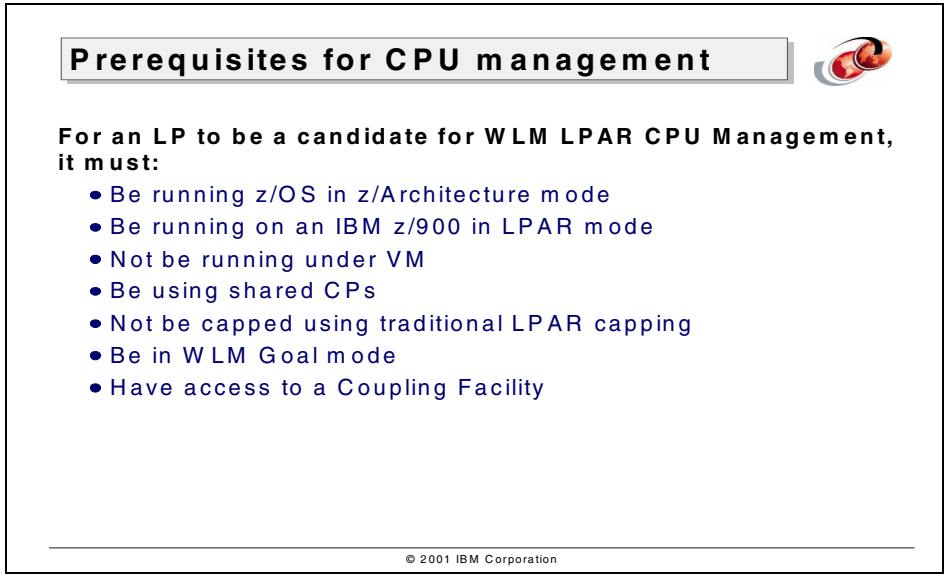
Based on these discussions about traditional CPU management techniques, the chart above summarizes the problems or shortcomings with these techniques.

The major problem has to do with this being a manual process. As a result:

- You have to decide in advance what you need, bearing in mind total available capacity, total requirements, requirements of each LP, 'spikiness' of workload, minimizing overhead, and minimizing future disruptions. Changing an LP from using dedicated to shared CPs requires deactivation of the LP. Also, if you do not define any Reserved CPs for the LP, a deactivation of the LP is also required to increase the maximum number of CPs for the LP.
- Capping can prevent available CP resource from being utilized. Let's say that 50 MIPS sits unused because of LP capping. That may be deemed acceptable. However, imagine if the LPAR overhead increased by 50 MIPS - would that be considered acceptable? In both cases you have paid for 50 MIPS that are not available to your workloads.
- The operator does not know that an LP's weight needs to be changed.
- The operator does not know that an LP is suffering a CPU delay problem.
- The operator may not know the relative importance of work running in each LP.

- Attempting to make dynamic changes requires that an operator has detailed knowledge of workloads and monitoring tools.
- Usually, operators have no time for this type of analysis.

2.8 Prerequisites for WLM CPU Management



WLM LPAR CPU Management is not necessarily available to all LPs. The above chart summarizes the requirement for an LP if it is to be managed by WLM LPAR CPU Management. Specifically, the LP should:

- Be running z/OS 1.1 or later in z/Architecture mode

  An IBM zSeries 900 CPC in LPAR mode can run a number of different operating systems, including VM/ESA and LINUX. However, as WLM LPAR CPU Management is only used by z/OS, we only talk about LPs running z/OS. The only point to note about LPs running different operating systems is that they can share CP resources with z/OS LPARs. The CP resources are distributed to each LP by LPAR LIC based on their current weight. The only effect they have on z/OS is that, if they are not using all of the physical CP resources available to them based on their current weight, z/OS LPs can use this spare capacity. Other operating system's current weights are not altered by WLM LPAR CPU Management.

  Note: At the time of writing, IBM has previewed the ability for WLM LPAR CPU Management to also manage the weights of Linux LPs that are using CPs (as opposed to IFLs). This support will be delivered with z/OS 1.2.

- Be running on an IBM zSeries 900

  This new capability is only available on IBM zSeries 900 CPCs. z/OS running on an IBM 9672 cannot use WLM LPAR CPU Management.

- Not be running under VM

  Because WLM LPAR CPU Management functions by changing LP weights, it cannot be used to manage z/OS systems that are running under VM.

- Be using shared CPs

  If an LP is using a dedicated CP, any weights set for that LP are meaningless: the LP has 100% of the capacity of the dedicated CPs, whether it needs that capacity or not.

- Not be capped using traditional LPAR capping

  Traditional LPAR capping is capping as it exists on the IBM 9672 CPC (also available on an IBM zSeries 900). There is now another form of capping (called soft-capping), which is used with IBM License Manager. IBM License Manager provides the ability to license a product based on the capacity of the LP the product is running in. The product is licensed, and the capacity of the LP can be capped, in terms of Millions of Service Units (MSUs). Once the rolling 4-hour CPU usage of the LP reaches this defined number of service units, LPAR capping of the LP is dynamically invoked by WLM. This subject is discussed in 2.15, “Relationship to IBM License Manager” on page 42. It is mentioned here to distinguish it from traditional LPAR capping. For more information on IBM License Manager refer to the IBM Redbook *z/OS IBM License Manager Installation and Use*.

- Be in WLM Goal mode

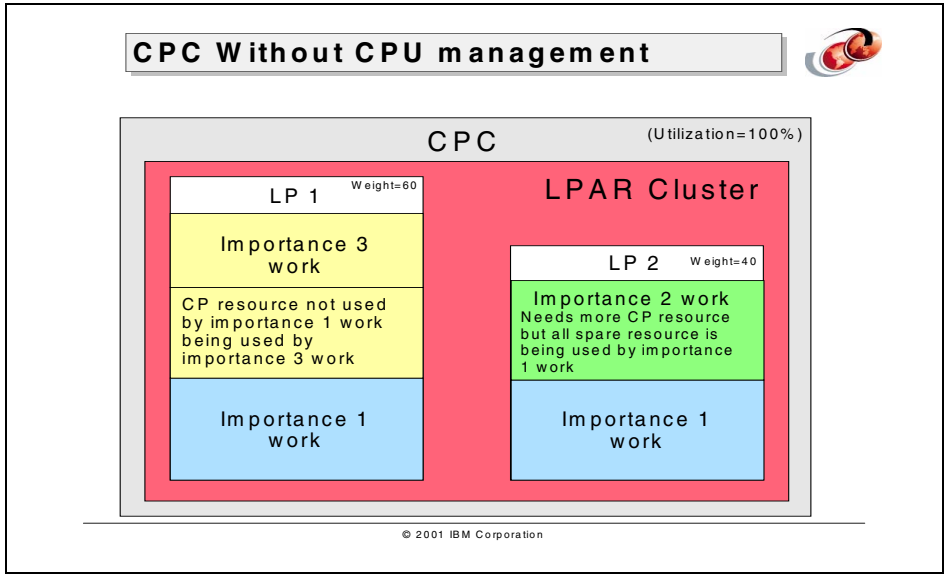
  Both WLM LPAR Weight Management and WLM LPAR Vary CPU Management are invoked during the execution of the WLM Policy Adjustment routine, which only gets invoked when WLM is in Goal mode. WLM Policy Adjustment looks at Service Class Periods (SCPs) and how it can help them better meet their goals. WLM LPAR Weight Management can be invoked to help an SCP that is suffering from CPU delays.

  However, WLM LPAR Vary CPU Management does not look at SCPs at all. It is simply trying to tune the CP resources of a z/OS image by balancing the number of online logical CPs with the actual CP capacity requirements of the LP.

- Have the images in a Parallel Sysplex (LPAR Cluster)

  WLM LPAR CPU Management uses a CF structure to store information that it uses when projecting the impact of a weight adjustment on an LP.

2.9 WLM LPAR Weight Management I



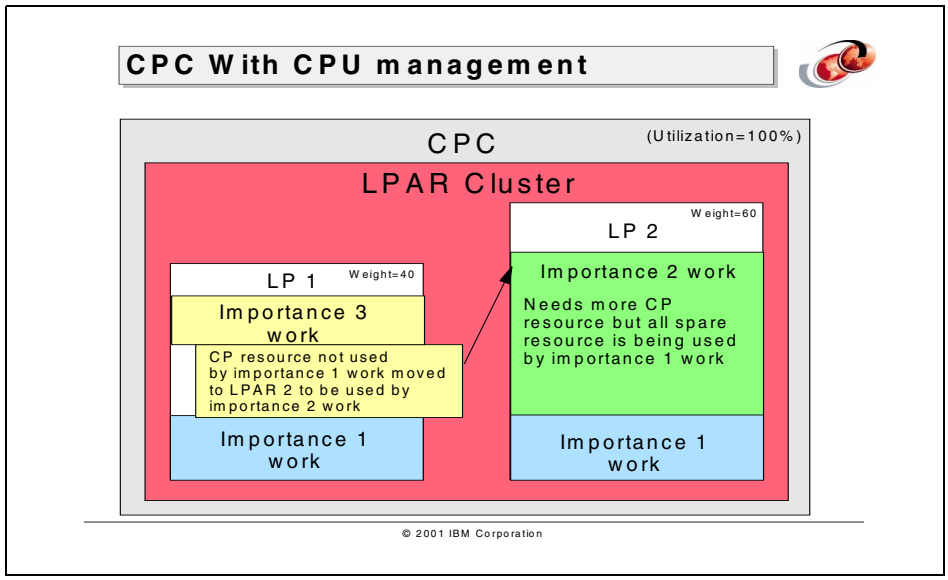
OK, now that you're an expert on PR/SM, we start to discuss the first function of WLM LPAR CPU Management: WLM LPAR Weight Management.

The figure above shows a CPC with two LPs, both in the same Parallel Sysplex. Both LPs are running importance 1 work. Work here means a WLM Service Class Period. LP1 is also running importance 3 work. LP2 is running importance 2 work. As none of these workloads support data sharing, they must run in a specific z/OS image. Both LPs are running at 100% and using all the available CP resource, as dictated by their weights.

The importance 2 work in LP2 is missing its goal due to CPU delays. It cannot take any CP resource from the importance 1 workload in the LP since that requires all the CPU it is using to meet its goals. Meanwhile, in LP1, there is CP resource not required by importance 1 work. It is being used by the importance 3 workload. It is important to note that there is no unused capacity on the CPC.

Without WLM LPAR CPU Management, there is no way to automatically move the CP resource being used by importance 3 work in LP1 to LP2, where the (more important) importance 2 work requires it. It could be done manually, but realistically, no one manages LP weights in this manner.

2.10 WLM LPAR Weight Management II



In the above example, we have implemented WLM LPAR CPU Management. LP1 has CP resource that is not being used by importance 1 work but is used by importance 3 work. LP2 needs more CP resource to enable importance 2 work to meet its goals. All available CP resource on LP2 is being used by importance 1 work to meet its goals.

In this case, WLM LPAR Weight Management is able to move CP resources from the importance 3 work on LP1 to importance 2 work on LP 2 by changing the weights of each LP. This gives LP2 access to more CP resource and decreases the amount of CP resource LP1 has access to. This may cause the importance 3 work in LP1 to miss its goal; however, the more important importance 2 workload in LP2 should now meet its goal.

WLM LPAR Weight Management enables WLM to move the CP resources from one LP (donor) to another (receiver) so that higher importance work can meet its goal. However, before WLM moves CP resource, it ensures that any work of equal or higher importance on the donor LP will not suffer as a result of the change.

This is similar to the notion of Mohammed and the mountain—if you have a workload that cannot be moved to the resource (it does not support data sharing), then you move the resource to the workload.

Customers that have years of experience with the S/390 platform understand the importance of having a balanced configuration (CPU, I/O, and storage) to provide optimal performance and avoid bottlenecks. The simplest topology for reaching this happy state is to place all your workloads in the same z/OS image on a CPC running in basic mode. However, for several reasons, such as software availability, lack of capacity, and workload isolation, this configuration is often not feasible.

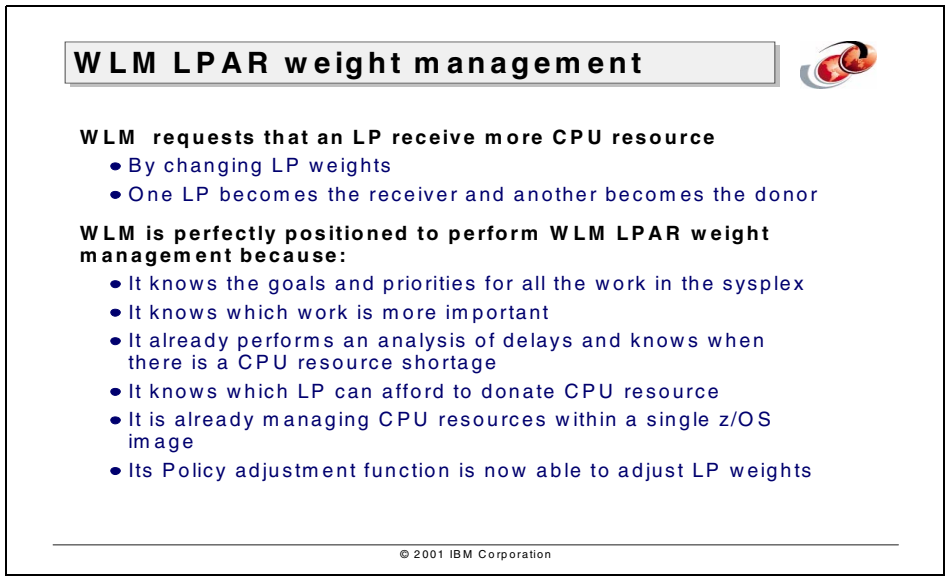
So, given that you probably have your workload spread over a number of z/OS images, many DASD controllers, thousands of DASD devices and hundreds of channels, how do you ensure a balanced configuration (and one that remains balanced over time)? Two of the things you can do are:

1. Implement SMS, to distribute the data set allocation over your controllers and devices. However, as time goes on, data may become inactive but remain on DASD, leading to skews in activity rates across volumes and controllers.
2. Implement Parallel Sysplex data sharing and use WLM Goal mode to help distribute your work across the systems in the sysplex. Unfortunately not all workloads support data sharing and workload balancing, so you will probably still have some work that has an affinity to a particular system image.

This is where the Mohammed and the mountain analogy comes in. If you are less than 100% successful in distributing your active data evenly across the devices, and your transactions among the z/OS images, then the only way to give those workloads the service they require is to move the resource (CPU and channels) to the workload.

WLM in Goal mode knows which workloads are more important and therefore has the knowledge to control which LP should be given the CP resource. The same applies to I/O channel paths, as you will see when we discuss Dynamic Channel-path Management.

2.11 WLM LPAR Weight Management III



You will recall that a major function of WLM Goal mode is to manage resources and workloads in an attempt to meet the goals for *all* workloads in the sysplex. One of the ways it achieves this is by changing the dispatching priority of address spaces in an SCP that is missing its goal because of CPU delays. This is done by the Policy Adjustment routine. Using WLM LPAR Weight Management, this concept is extended to include the possibility of changing the weight of an LP. Consequently, within an LPAR cluster, if the transactions in an SCP are:

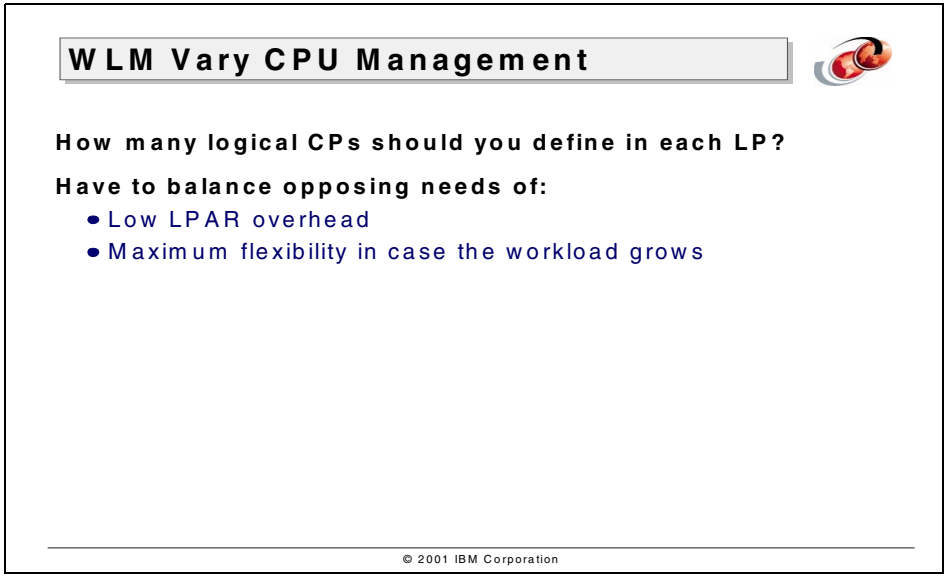
- The most important ones in the LP (LP1).
- Not achieving the goal that was specified for them (that is, the performance index is greater than one).
- Suffering because they are delayed waiting for access to the CPU.
- Unable to take some dispatching priority from another SCP in the same image because the SCPs that are significant users of CPU are more important ones.
- Can take enough weight from a donor system to make a significant improvement in the PI of the work that is missing goal.
- Weight donation does not cause more important work to miss its goal or cause the PIs of SCPs with the same importance to diverge. WLM attempts to have a similar PI for all SCPs that have the same importance. This is an existing WLM characteristic and is not unique to an IRD environment.

If all these conditions are met, WLM will decrease the weight of LP2 (it is the donor) and the weight of LP1 is increased by the same amount (it is the receiver).

WLM in Goal mode is in the best position to make decisions about the values of LP weights in an LPAR cluster environment for the following reasons:

- It knows the goals and priorities for all the work in the sysplex.
- It knows which work is more important.
- It already has the logic to perform an analysis of delays and can determine when there is a CPU resource shortage.
- It already has the logic to project the impact of a planned change on the potential donor.
- It knows which LP can afford to donate CPU resource.
- It is already managing CPU resources within a single z/OS image.
- Its Policy Adjustment function is now able to adjust LP weights.
- In an IRD environment, WLM first decides on the target weight for the LP, *then* decides how many CPs should be online to the LP to enable this weight to be achieved.

2.12 WLM Vary CPU Management



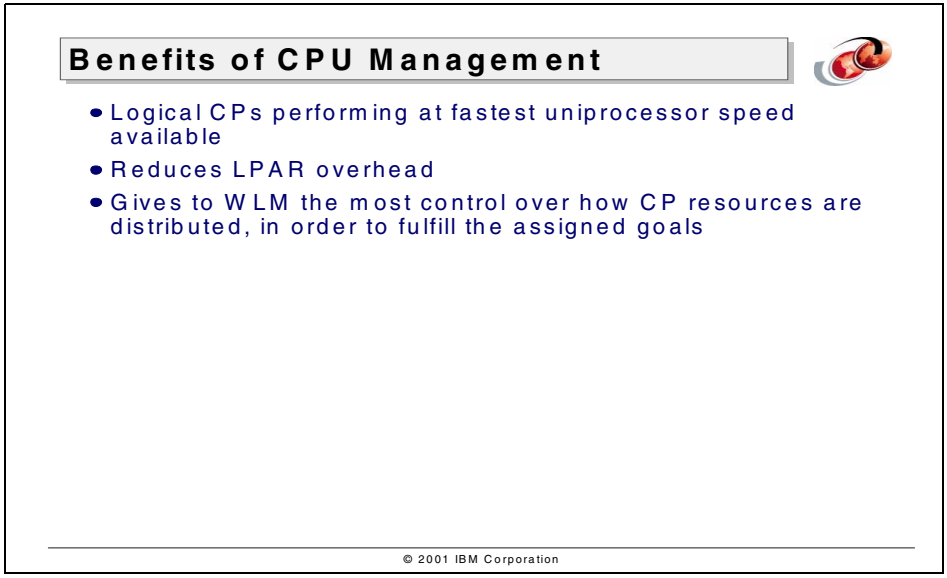
WLM LPAR Vary CPU Management addresses a question that has existed since IBM introduced PR/SM: How many logical CPs should you define in each LP?

For maximum flexibility, and to potentially give each LP access to the full power of the CPC, you would define the number of logical CPs in an LP to be equal to the number of shared physical CPs.

On the other hand, to minimize LPAR overhead, you would give each LP as few logical CPs as possible, while still trying to make sure that each LP will have access to sufficient capacity.

With WLM LPAR Vary CPU Management, you can have the best of both worlds. You can define each LP with the maximum number of logical CPs, and WLM LPAR Vary CPU Management will configure offline logical CPs that are not currently required by the LP. This provides the highest logical CP speed and least LPAR overhead. If WLM decides to increase the weight of an LP, and this would result in more CP resource than can be provided by the current number of online logical CPs, WLM LPAR Vary CPU Management will configure an additional logical CP online.

2.13 Value of WLM LPAR CPU Management

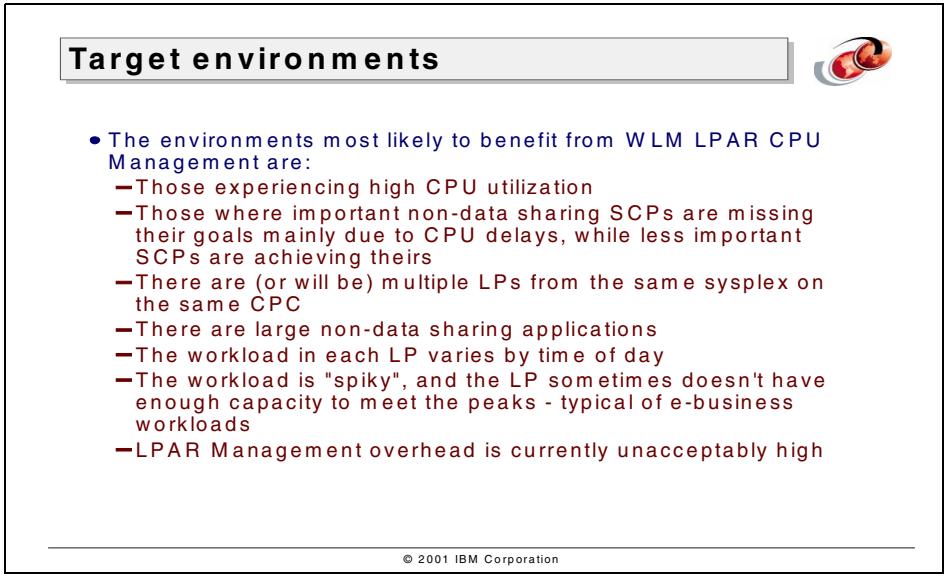


The value of WLM LPAR CPU Management can be summarized by saying that it provides improved value from the installed CP resources, and it provides additional flexibility while at the same time reducing overhead.

The WLM LPAR Weight Management function attempts to ensure that important workloads will not be delayed by lack of CPU if there are other, lower importance, workloads that are using that resource *anywhere in the LPAR Cluster*.

The WLM LPAR Vary CPU Management function is aimed more at maximizing the efficient use of the available CPU resources. However, in the process of doing this it gives you the ability to create a configuration designed for maximum flexibility, without having to suffer the cost that normally accompanies such a configuration. Also, by minimizing the number of logical CPs, it increases the effective speed of each one.

2.14 When do you need WLM LPAR CPU Management?



The chart above lists the environments that are likely to benefit from WLM LPAR CPU Management. Basically, if you have some workload that does not support workload balancing (and therefore cannot move to where the resource is), and your CPCs run at or near 100% for some of the time, and you have more than one LP from a sysplex on the CPC, then you are likely to benefit from WLM LPAR CPU Management.

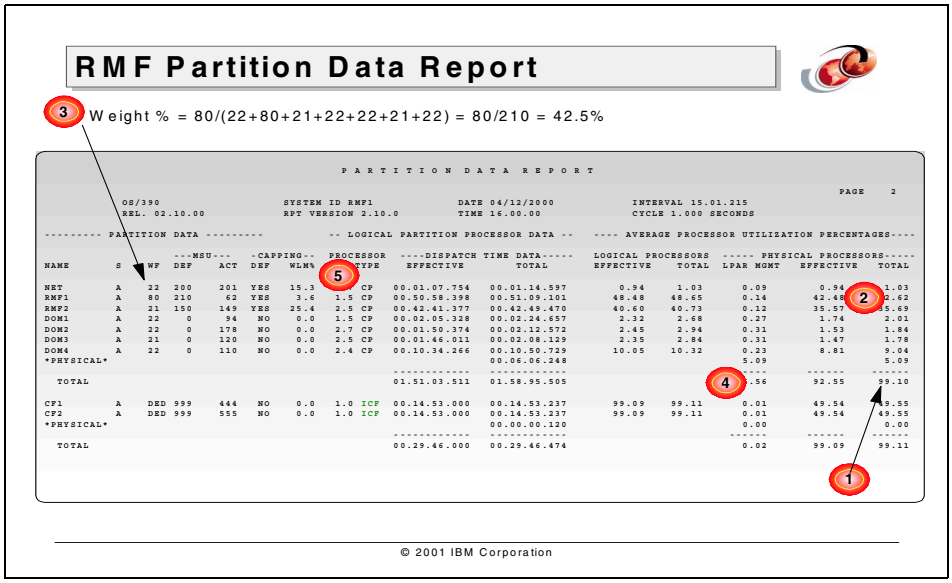
Obviously you have to meet the prerequisites of being on an IBM zSeries 900, you must be running z/OS, and you must be in WLM Goal mode. Remember that IBM has announced that z/OS 1.2 will be the last release to support WLM Compatibility mode, so everyone will have to be in WLM Goal mode soon anyway.

You can use your current RMF reports to obtain some of the information to help you determine if WLM LPAR CPU Management will be beneficial in your environment. The information to check for is:

1. CP utilization in the CPC close to 100%.

   Look in the RMF Partition Data Report in the column “AVERAGE PROCESSOR UTILIZATION PERCENTAGES (PHYSICAL PROCESSORS - TOTAL)” in the last line (TOTAL) (Field ‘1’ in the sample report on the next page). If this number is close to 100% (remember this is the average over an interval that is normally 15 minutes), then continue to item 2.

.



2. All the non-idle LPs in the LPAR Cluster are running at close to or greater than the capacity guaranteed by their weights.

   Look in the Partition Data Report in the column “AVERAGE PROCESSOR UTILIZATION PERCENTAGES (PHYSICAL PROCESSORS - TOTAL)” in the lines for the LP(s) you are interested in (field 2 in the figure above).

   If the value in this field is close to the percentage of the CPC guaranteed by the weight for the LP (add the weights for all the non-CF LPs and divide the weight for the LP you are interested in by this value as shown by field 3 in the above figure), that means that this LP is using all the CP that it is being given and could probably use more if it had a higher weight value.

3. High LPAR overhead (above 7%).

   The LPAR overhead is reported in the RMF Partition Data Report in the column “AVERAGE PROCESSOR UTILIZATION PERCENTAGES (PHYSICAL PROCESSORS - LPAR MGMT)” in the last line (TOTAL, field 4 in the figure above).

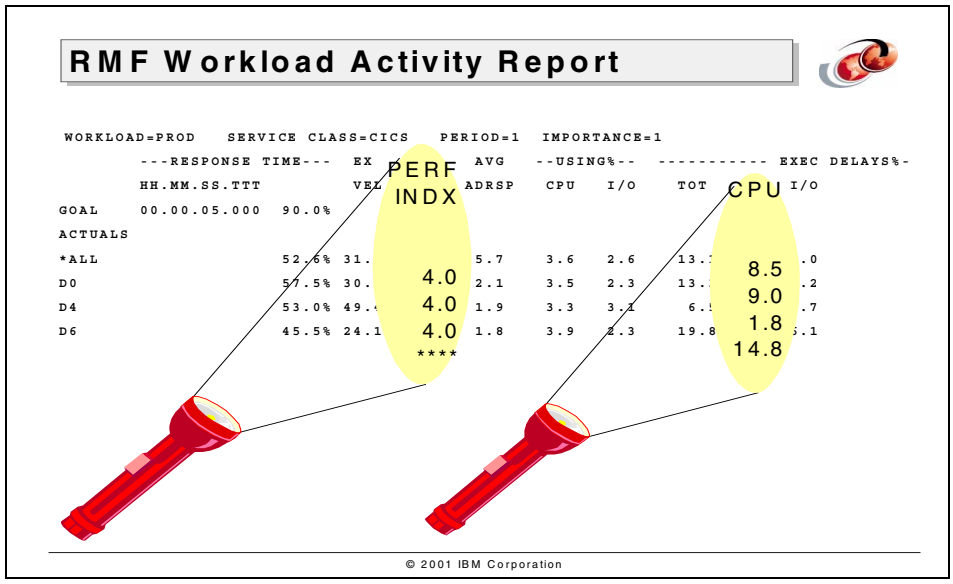
4. Insufficient number of logical CPs in an LP.

   In the RMF Partition Data Report, compare the capacity used by the LP with the capacity available to it based on the number of logical CPs in the LP. The capacity actually consumed is reported in the column “AVERAGE PROCESSOR UTILIZATION PERCENTAGES (PHYSICAL PROCESSORS - TOTAL)” in the line for your LP. Multiply this value by the number of physical CPs to get the equivalent number of physical CPs consumed by your LP.

Now compare this with the number of logical CPs defined for the LP—this is shown in the column PROC NUM, marked with a 5 in the figure. If the two values are close, then you should consider defining more logical CPs for this LP. WLM LPAR Vary CPU Management lets you define more logical CPs than the LP needs, without having to be concerned about the resulting LPAR overhead.

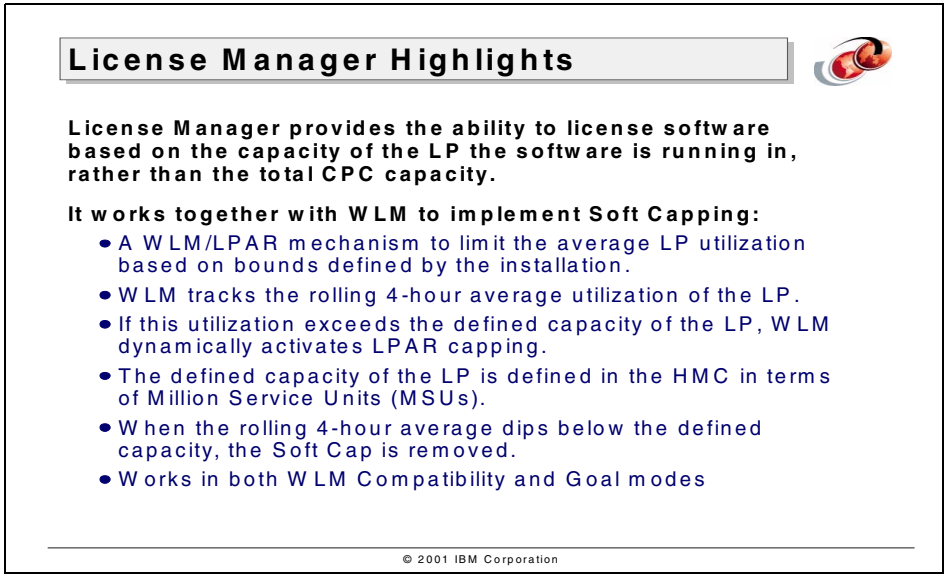
5. Important SCPs are missing their goals.

   Check the system PIs for all workloads in each LP that will be in the LPAR Cluster. If important SCPs are missing their goals (mainly due to CP delays) and other less important workloads (that are significant CPU users) are achieving their goals, the LP containing those less important SCPs is a potential donor. An extract from a sample RMF Workload Activity Report is shown in the figure below.



In this example, the Service class called CICS has an importance of 1 and is being delayed by having to wait for access to the CPU for a significant amount of the time. So this is an SCP that could potentially benefit from WLM LPAR Weight Management—the next step is to check the Workload Activity report for lower-importance SCPs that are significant CPU consumer and are running in other LPs in the same LPAR cluster.

2.15 Relationship to IBM License Manager



Prior to the announcement of the LPAR-level software charging introduced as part of Workload Charging, most S/390 software, both from IBM and other vendors, was priced based on the size of the physical machine on which it ran. The license for a product that runs on a 100 MIPS CPC is considerably less expensive than a license for the same product running on a 1000 MIPS CPC. Generally speaking, the price is independent of the amount of work actually done by the product. As a result, many customers run many small to medium-sized CPCs rather than a smaller number of larger ones. This generally results in a more complex configuration, and excludes the installation from many of the benefits provided by running multiple LPs under PR/SM on a larger processor.

Workload Charging introduces the capability to pay software licenses for some products based on the size of the LP the product is running on, rather than on the total capacity of the CPC. This capability is available on IBM zSeries 900 CPCs when all the MVS-based LPs are running z/OS 1.1 or later. The capability is enabled by a new z/OS component called IBM License Manager and new functions in WLM. This is discussed in detail in the IBM Redbook *z/OS IBM License Manager Installation and Use*, but basically you define a capacity for the LP (in terms of Millions of Service Units (MSUs)) and WLM ensures that the capacity used by the LP on a rolling 4-hour average does not exceed this amount. The amount of capacity that you define for the LP is called its *defined capacity*, and it is this capacity that is used by IBM License Manager to determine the license capacity required for any LP that runs in that LP.

It is important to understand that LP-level software charging and Intelligent Resource Director are independent of each other. You might choose to use Intelligent Resource Director but not the LPAR-level software charging capability, or LPAR-level software charging but not Intelligent Resource Director, or you can use both of them.

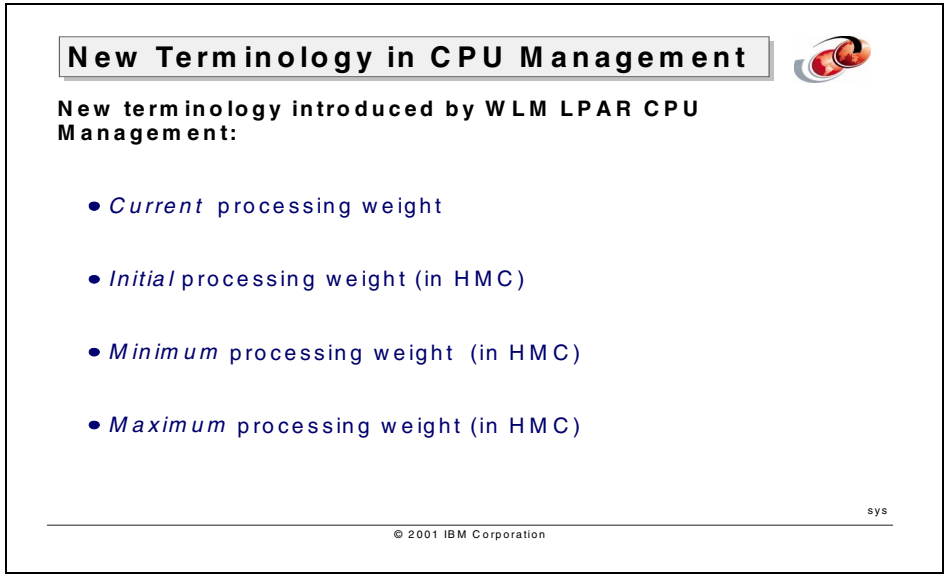
The functions of LPAR-level software charging may seem to be mutually exclusive with WLM LPAR CPU Management, but in fact the two work together. Remember that WLM enforces the capacity *based on a rolling 4-hour average*. So even if the defined capacity for a given LP is only 30% of the total CPC capacity, it is still possible for that LP to use much more than this capacity for short periods, as long as the rolling 4-hour average has not exceeded the defined capacity for that LP. As long as the LP is not soft-capped, the amount of CPU allocated to the LP will be determined by WLM based on the normal WLM LPAR Weight Management criteria. So, WLM will still adjust the weight for the LPs in the LPAR Cluster, based on the importance of the work running in those LPs, and the weight assigned to an LP at any given time may well exceed the share of the CPC that is indicated by the defined capacity for the LP.

If the 4-hour average consumption of the LP does reach the defined capacity, WLM works with LPAR LIC to dynamically turn on LPAR capping for the LP to bring it back down to its defined capacity. As long as the LP is being capped in this manner, WLM will not adjust the weight of the LP. Once the average consumption drops below the defined capacity amount, the LP is once again eligible for WLM LPAR Weight Management. It is important to point out that you should *not* expect to run for prolonged periods with an LP being soft-capped: if you find this is happening consistently, the certificates and the defined capacity of the LP should be adjusted to reduce the amount of time the LP is soft-capped.

It is important to understand that WLM will *not* adjust the defined capacity for an LP—only the installation can adjust the defined capacity for an LP. This means that if you are utilizing Workload Charging, you should not expect IRD to make significant changes in the weight of an LP for a prolonged period of time.

For example, if you want LP A to have most of the weight for all of the day shift, and LP B to have most of the weight for the whole night shift, this will not work, unless you set the defined capacity for each LP to be equal to its maximum rolling 4-hour average. You need to make a decision between maximum flexibility, whereby you set the defined capacities to match the maximum 4-hour average, and minimizing software costs, whereby you really don't want the defined capacities of all the LPs in the LPAR Cluster to exceed the share of the CPC used by those LPs at any given time. This is discussed further in 4.10, "IBM License Manager considerations" on page 121.

2.16 New terminology for WLM LPAR CPU Management



The IBM zSeries 900 CPC and WLM LPAR CPU Management introduce new terms for LPAR weights. The following are used when discussing LPAR weights in this chapter:

- Current processing weight

  This refers to an LP's weight at any given point in time when the system is running. This is the weight that WLM LPAR Weight Management and WLM LPAR Vary CPU Management use when performing their primary functions.

- Initial processing weight

  This weight becomes the LP's current weight immediately following an IPL. If you switch an LP from WLM Goal mode to Compatibility mode, its current weight will revert to this value. Similarly, if you turn off WLM LPAR Weight Management for an LP, the weight of the LP will revert to this value.

- Minimum processing weight

  This is the minimum value that WLM LPAR Weight Management can assign as an LP's current weight. This value is defined by the installation, but it is recommended that you default this value leaving WLM free to automatically set a lower limit on weight that provides enough capacity for critical system work such as global enqueues.

- Maximum processing weight

  This is the maximum value that WLM LPAR Weight Management can assign as an LP's current weight. This value is defined by the installation.

3

How WLM LPAR CPU Management works

If you have gotten this far in this part of the book you obviously believe that WLM LPAR CPU Management can be beneficial in your environment. While it is not necessary to understand exactly how WLM LPAR CPU Management works in order to implement it, we have provided this information for those customers that like to have this level of information. If you feel that you do not need this detailed information at this time, you can skip this chapter and go directly to Chapter 4, “Planning for WLM LPAR CPU Management” on page 101.

This chapter explains the different parts of WLM LPAR CPU Management and how each part works. In order to understand this, we first talk about the functions that have been available to date. WLM LPAR CPU Management is really an extension to these existing functions and provides much closer interaction between the hardware (LPAR LIC) and the software (WLM). This extension has one important benefit over the existing functions; it works dynamically across systems to provide the resources where they are needed. This is of great benefit because it continues along the path of automatically adjusting the allocation of hardware resources in order to achieve the customer-specified workload goals.

Just to remind you, an LPAR Cluster, which is the scope within which WLM LPAR CPU Management works, is the set of z/OS LPs (operating in z/Architecture mode) on the same IBM zSeries 900 CPC that are members of the same Parallel Sysplex. All the functions that we discuss for WLM LPAR CPU Management only operate on the LPs within an LPAR Cluster.

There are two parts to WLM LPAR CPU Management:

1. WLM LPAR Weight Management
2. WLM Vary CPU Management

We describe each of these separately as each operates somewhat independently of the other. Also, it is possible (although not recommended) to use just one of these functions. However, they are invoked in a specific order so that one complements the other. That is, WLM LPAR Weight Management is invoked first, followed by WLM LPAR Vary CPU Management, both during the WLM Policy Adjustment routine. This is because WLM LPAR Weight Management might change the current weights of LPs so that the appropriate amount of physical CP resource is available to each LP in the LPAR cluster.

This is followed by WLM LPAR Vary CPU Management, which varies logical CPs online or offline to z/OS. This ensures that the number of online logical CPs is appropriate for the CP resource being assigned to this LP by its current weight.

For example, a decision by WLM LPAR Weight Management to change the current weight of an LP may cause an additional logical CP to be varied online or offline by WLM LPAR Vary CPU Management.

Throughout this book, we use the following WLM internal values in the examples. These values are only artificial values to make the examples easier to understand. Some of them are the actual values used by WLM at the time of writing, however, these values may (and probably will) change over time in response to changes in hardware and software or customer experiences. Such changes will take place without notice.

- Weight adjustment: When WLM LPAR Weight Management decides to change the weight of an LP, it adjusts the receiver LP and the donor LP by a percentage of the average weight of all the LPs in the LPAR Cluster. In our examples, we assume that the percentage is fixed at 5%; however, in practice the adjustment depends on the workloads in the LPs and how they are performing against their WLM goals.
- WLM interval: This is the frequency with which the WLM Policy Adjustment routine runs on each system. Note that the Policy Adjustment routines are not coordinated across the LPs. The routine will more than likely run at a different time on each LP. We use a frequency of one Policy Adjustment cycle every 10 seconds in our examples.

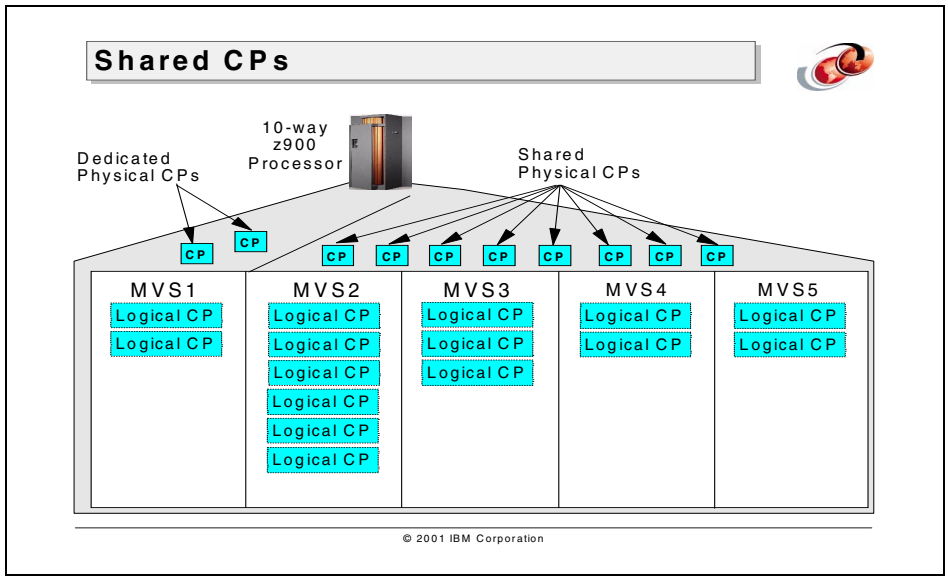
- Time between changes in weights and varying offline logical CPs: When WLM changes the weight of an LP, it is important that the change is given a chance to have an effect before another change is made. We use a value of 1 minute between each time WLM will change the weight of an LP, or vary a CP offline to an LP. In practice, the interval between adjustments is variable, and depends on a number of factors, including the PI of the Service Class Period that is being helped.
- When WLM LPAR Vary CPU Management is deciding how many CPs should be online to an LP, it tries to ensure there is always a buffer between the amount of resource used or guaranteed at the moment, and the number of logical CPs that are online. The idea is that if there is a sudden increase in the workload of the LP, WLM wants to make sure that the LP will not be restricted by the number of logical CPs available to the LP. Similarly, when WLM LPAR Vary CPU Management is deciding if it should take a logical CP offline, it wants to be conservative in this change to ensure that it does not overreact to a temporary decrease in the workload in the LP.

In our examples, we say that WLM will always keep a buffer of one more logical CP worth of capacity online than is currently required. When varying a logical CP offline, we use a buffer value of two logical CPs. In practice, the size of the buffer depends on the number of CPs that are currently online—it will tend to have a larger buffer when there is a smaller number of CPs online.

Note: We want to stress again that these numbers are used simply to make the examples easier to follow. By the time you read this, WLM might be using different values for all these variables. And the actual numbers are not that important anyway; what is important is the principle of how WLM LPAR CPU Management works.

In order to set the scene for the new functions provided in WLM LPAR CPU Management, we first describe how LPAR actually works in a shared CP environment. After that, we progress into how WLM LPAR CPU Management itself actually works.

3.1 Shared logical CPs example



Because LPAR management of the shared CP environment is central to both components of WLM LPAR CPU Management, we provide a fairly detailed description of it in this section. Also, because this information is not provided in any other current documentation, it is important that you can get this information from somewhere.

Our example will be based on the configuration in the above figure. We have a 10-CP IBM zSeries 900. We use the term “physical CP” to refer to the actual CPs that exist on the CPC. We use the term “logical CP” to refer to the CPs each operating system has available on which to dispatch work. The number of logical CPs in an LP must be less than or equal to the number of physical CPs.

In the example, two CPs are dedicated to LP MVS1. The two dedicated CPs are for use exclusively by MVS1. For this LP then, the number of physical CPs is equal to the number of logical CPs. The remaining eight CPs are shared between the LPs: MVS2, MVS3, MVS4, and MVS5. Each of these LPs can use any of the shared physical CPs, with a maximum at any one time equal to the number of online logical CPs in that LP. The number of logical CPs per LP is:

- Six logical CPs in LP MVS2
- Three logical CPs in LP MVS3
- Two logical CPs in LP MVS4
- Two logical CPs in LP MVS5

The number of physical CPs does not have to be equal to the number of logical CPs in an LP when sharing CPs.

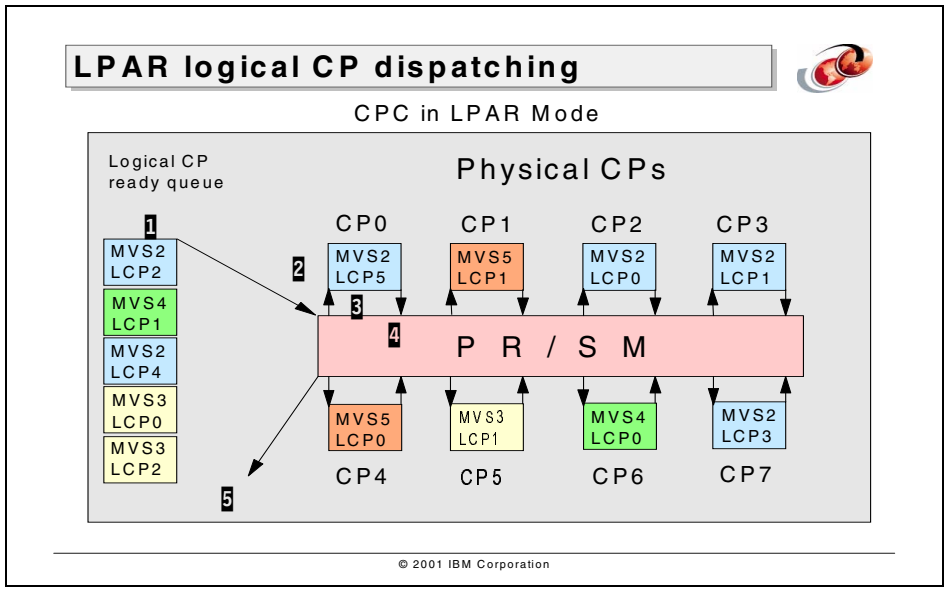
An operator could vary logical CPs online and offline, as if they were physical CPs. This can be done through the z/OS CONFIG command.

An LP cannot have more logical CPs online than the number defined for the LP in the HMC. Note, however, that Capacity Upgrade on Demand (CUoD) provides the ability to define *reserved* CPs for an LP. These reserved CPs represent future CPs that may be installed non-disruptively on the CPC, and can then be non-disruptively added to the LP.

The HMC Image Profile contains the initial number of CPs that will be online when the LP is IPLed, and it can also contain a value in the Reserved field. We strongly recommend specifying a Reserved value that will allow you to move to the maximum physical capacity of the CPC non-disruptively. There is no overhead or cost involved in doing this, and you gain the ability to non-disruptively add logical CPs to the LP. When the actual upgrade is applied, the operator needs to CONFIG the new CPs online the first time. After that, WLM LPAR Vary CPU Management will manage them in the same manner as all the other logical CPs in the LP.

In order to determine the correct number of logical CPs in an LP, you should take the following variables into consideration: LPAR overhead; the level of parallelism you want; correct use of the assigned weight; the number of physical CPs used by the LP; the desired logical CP speed. Sometimes the correct value in the night shift is not the best for the mid-morning workload, and other times it varies from one second to another. Refer to 3.9.1, “WLM LPAR Vary CPU Management concepts” on page 82 for more information on this topic.

3.2 LPAR dispatching and shared CPs



Something that many people do not fully understand is how LPAR dispatching of logical CPs on physical CPs works. As this affects some of the formulas we will be discussing, we provide some information here on how this process works.

Note: Before we start, we want to reiterate that a *dedicated CP* can only be used by the LP to which it was assigned. No other LP can use this CP. As a result, dedicated CPs are not managed by WLM LPAR CPU Management and therefore are not discussed further.

The code that provides the LPAR dispatching function is called LPAR Scheduler Licensed Internal Code (LIC). LPAR LIC logic executes on all of the physical CPs. LPAR LIC dispatches a logical CP on a physical CP (the one the LPAR LIC is currently running on) by issuing the Start Interpretive Execution (SIE) instruction with the logical CP represented by a state control block, as a parameter. This causes the operating system code or application code in the LP to execute on the physical CP, through the logical CP. The logical CP is dispatched on the physical CP by copying the LP's logical CP status (PSW, registers, and so forth) from HSA to the corresponding actual entities.

This function can even provide a different set of instructions for each LP, depending the architecture defined for the LP. For example, the one physical CP could be used to dispatch OS/390, CFCC, and Linux LPs one after the other. When the logical CP is intercepted, the logical CP status is saved in HSA and the LPAR LIC is automatically dispatched on the physical CP again. This code then chooses another logical CP to dispatch, and the whole process starts again.

Conceptually, this process is not that different from the way the z/OS dispatcher works. For example:

- The LPAR Scheduler LIC corresponds to z/OS dispatcher code.
- Each logical CP corresponds to an z/OS dispatchable unit (task, or service request).
- The logical CP state information stored in the HSA corresponds to the TCB or SRB.
- The SIE instruction corresponds to the LPSW (Load PSW) instruction.
- An intercept of a logical CP corresponds to an interrupt for a z/OS dispatchable unit. Information about the possible intercepts for a logical CP is provided in 3.3, “Reasons for intercepts” on page 55.

Coming back to our example, let us follow the flow of execution, where we have eight shared physical CPs.

Every logical CP represents a dispatchable unit to LPAR LIC. In our example, MVS2 has 6 logical CPs, MVS3 has 3 logical CPs, MVS4 has 2 logical CPs, and MVS5 also has 2 logical CPs - this gives us a total of 13 logical CPs, so LPAR LIC has up to 13 dispatchable units to manage in our environment. LPAR LIC is responsible for dispatching logical CPs on physical CPs.

When a logical CP is ready to run (not in wait), it is placed on the logical CP ready queue. This queue is ordered by a priority that is based on the LP weight and the number of logical CPs, as declared by the installation. This is discussed in more detail in 3.8, “WLM LPAR Weight Management” on page 72.

So how does a logical CP move from being on the ready queue to executing on a physical CP, and where is the processing performed that does this? As mentioned before, LPAR LIC, sometimes referred to as the LPAR scheduler, is responsible for dispatching a logical CP on a physical CP. LPAR LIC executes on each physical CP. When it is ready to dispatch a logical CP on a physical CP, LPAR LIC issues a SIE instruction, which switches the LPAR LIC code from the physical CP and replaces it with the code running on the logical CP.

In the figure on page 52, the steps that occur in dispatching a logical CP are as follows:

1 - The next logical CP to be dispatched is chosen from the logical CP ready queue based on the logical CP weight.

2 - LPAR LIC dispatches the selected logical CP (LCP5 of MVS LP) on a physical CP in the CPC (CP0, in our example).

3 - The z/OS dispatchable unit running on that logical processor (MVS2 logical CP5) begins to execute on physical CP0. It executes until its time slice (generally between 12.5 and 25 milliseconds) expires, or it enters a wait, or it is intercepted for some reason.

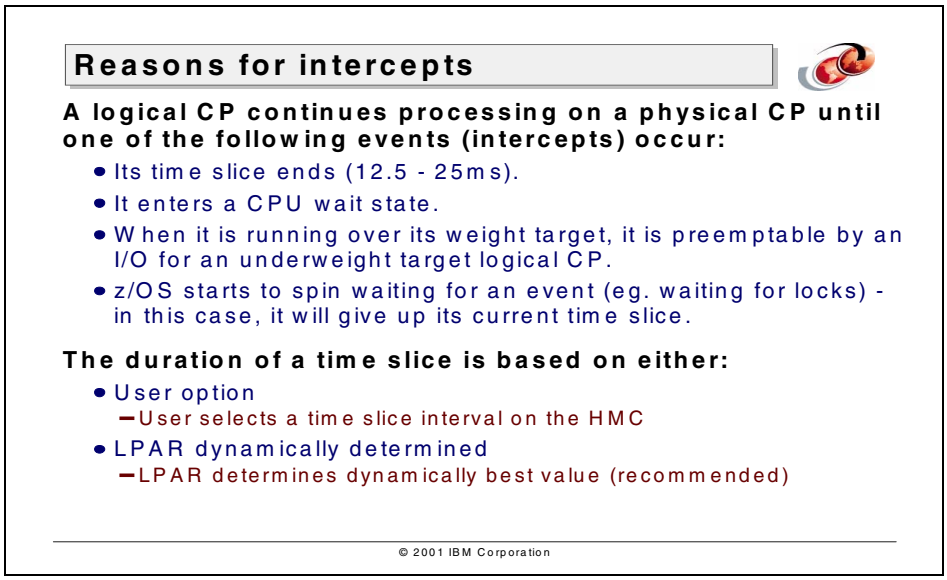
4 - In our example, the logical CP keeps running until it uses all its time slice. At this point the logical CP5 environment is saved and control is passed back to LPAR LIC, which starts executing on physical CP0 again.

5 - LPAR LIC determines why the logical CP ended execution and requeues the logical CP accordingly. If it is ready with work, it is requeued on the logical CP ready queue and step 1 begins again.

This process occurs on each physical CP. In our example with 8 physical CPs shared, LPAR LIC code executes in each of them.

This explains how a logical CP is dispatched from the logical CP ready queue to a physical CP. We now move on to describing time slices and how logical CPs are prioritized.

3.3 Reasons for intercepts



Every time that a physical CP is taken away from a logical CP, we have an intercept. The causes for an intercept are:

- End of the time slice. A *time slice* is the length of time that a logical CP is dispatched on a physical CP. The use of time slicing ensures that a task in a loop can't cause severe performance problems in other LPs, by monopolizing a physical CP.

  The default duration for a time slice is limited to between 12.5 and 25 ms. The user can override this and specify their own duration for a time slice. This is not recommended as it is unlikely that the user's specification is more efficient. The formula used for the default time slice is:

  $$\frac{(25\ \text{ms} * \text{number of physical CPs})}{\text{total number of logical CPs not in stopped state}}$$

- When it is running over its weight target, a logical CP is preemptable by an I/O for an underweight target logical CP. Refer to 3.5, "LPAR weights" on page 59 to get more information on this.
- z/OS is starting a spin loop and voluntarily gives up its current time slice. z/OS knows that it is functioning in an LP. This is so that it can give control of a physical CP back to LPAR LIC in certain circumstances (a spin loop, for example). Generally this happens when it is not doing any productive work.

This is an example of an event-driven intercept. Refer to 3.4, “LPAR event-driven dispatching” on page 57, for more information.

However, in general operations, z/OS behaves as if it is processing on a dedicated CPC where the number of physical CPs is equal to the number of defined logical CPs. One example of this is the determination of the SRM constant. Even though z/OS knows the number of physical CPs that are actually installed in the CPC, WLM uses an SRM constant (to convert CPU seconds to CPU service units) that is based on the number of logical CPs in this LP.

For consistency this is the best approach; however, it does not take into consideration that the MP effect is related to the number of physical CPs and not to the number of logical CPs. The SRM constant varies when the operator changes the number of logical CPs online, through the CONFIG command.

- The operating system places the logical CP in a wait (because of a no work-ready situation). This is another case of an event-driven intercept.

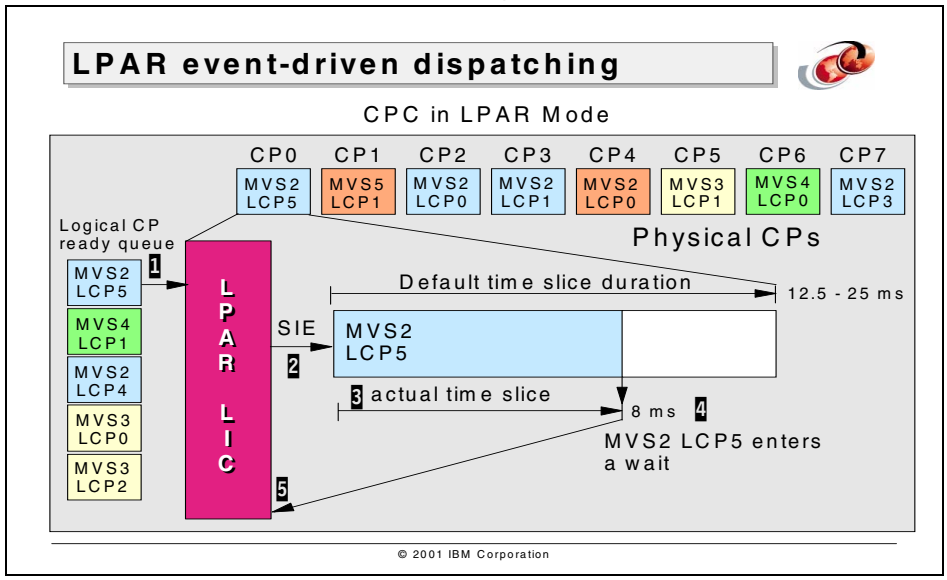
  Refer to 3.4, “LPAR event-driven dispatching” on page 57, to get more information about the event-driven function of LPAR.

The following case is *not* considered an intercept: If the operating system wants to execute some action not allowed in LPAR mode, such as changing the TOD clock (its contents are global and not local), the LPAR LIC gains control to simulate the action and immediately returns control to the same logical CP. The simulation provided by LPAR consists of keeping a TOD offset in the descriptor block of the logical CP, so that when z/OS asks for the time, the contents of the real TOD are adjusted by the amount contained in the offset.

Although we recommend against performing these actions, the installation can:

- Set the value of the time slice value in HMC. Rather than letting PR/SM dynamically determine the time slice, you can set a specific value, as low as 1 millisecond. Refer to 3.4, “LPAR event-driven dispatching” on page 57, for more information.
- Inhibit event-driving dispatching. Refer to 3.4, “LPAR event-driven dispatching” on page 57, for more information.

3.4 LPAR event-driven dispatching



Event-driven dispatching is the ability of LPAR to take a physical CP away from a logical CP before its time slice ends. This is generally because it is not doing any productive work. All cases of intercepts, except where the time slice ends, are examples of event-driven intercepts. Setting up your LPs as being event-driven (this is the default) is highly recommended because it ensures the most efficient use of CP resources.

The opposite of event-driven is time-driven, where the only time a logical CP will be intercepted is at the end of the time slice. Although this is not recommended, it can still be specified by the installation.

In the figure above, the steps showing event-driven dispatching are:

1 - LPAR LIC executing on physical CP0 selects LP MVS2 logical CP5 to dispatch on physical CP0.

2 - LPAR LIC dispatches MVS2 logical CP5 on CP0 through the SIE instruction.

3 - The z/OS dispatchable unit on MVS2 logical CP5 is executing on physical CP0. If it were to use all its time slice, it would execute for between 12.5 and 25 milliseconds (ms). In this example, it does not use all of its time slice.

4 - MVS2 logical CP5 enters a valid wait after 8 ms. This is detected by the LPAR LIC and the time slice is ended at 8 ms.

5 - MVS2 logical CP5 is returned to be requeued in a wait queue (waiting for an interrupt) and another ready logical CP is chosen for dispatch on physical CP0.

The user has two options in the HMC to affect the event-driven dispatching function:

- Time slice value, which can be:
  - Dynamically determined by the system (default). This is selected by checking the box titled **Dynamically determined by the system**.

    This indicates that the CPC is to use defaults. That is, event-driven dispatching is turned on and the time slice duration is determined by the system. It is highly recommended to use this default.
  - Determined by the user. This is selected by checking the box titled **Determined by the user**.

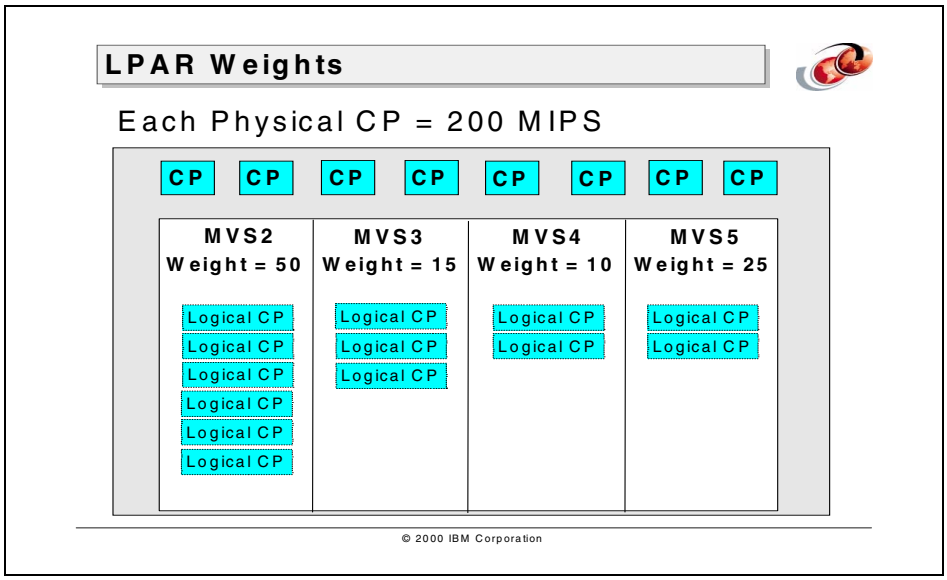
    In this case, you have to select your own time slice duration for use with event-driven dispatch. LPAR still uses event-driven dispatching, but overrides the default time slice duration with the value the user has specified here.
- Switch off event-driven dispatching by checking the box titled **Do not end the time slice if a partition enters a wait state**.

  If you switch off event-driven dispatching, you must specify a value for the dispatching interval. You could use the normal default values of between 12.5 and 25 milliseconds, or select values of your own. Once again, we do not recommend using this option - there are very few, if any, configurations that will perform better with event-driven dispatching turned off.

The RMF LPAR Activity report shows the status of event-driven dispatching (Wait Completion = NO means that event-driven dispatching is being used) and whether the time slice value is determined dynamically or has been set by the user. This part of the LPAR Activity report is as follows:

```
LPAR ACTIVITY REPORT


WAIT COMPLETION                          NO

DISPATCH INTERVAL                   DYNAMIC
```

3.5 LPAR weights



LPAR weights are used to control the distribution of shared CPs between LPs. Therefore, LPs with dedicated CPs do not use LPAR weights.

LPAR weights determine the guaranteed (minimum) amount of physical CP resource an LP should receive (if needed). This guaranteed figure may also become a maximum when either:

- All the LPs are using all of their guaranteed amount (for example, if all LPs were completely CPU-bound).
- The LP is capped using traditional LPAR capping.

An LP may use less than the guarantee if it does not have much work to do. Similarly, it can use more than its weight if the other LPs are not using their guaranteed amount.

LPAR LIC uses *weights* and the *number of logical CPs* to decide the priority of logical CPs in the logical CP ready queue. The following formulas are used by LPAR LIC in the process on controlling the dispatching of logical CPs:

- WEIGHT(LPx)% = 100 * WEIGHT LPx / SUM_of_ACTIVE LPs WEIGHTs
- TARGET(LPx) = WEIGHT(LPx)% * (# of NON_DEDICATE_PHYS_CPs)
- TARGET(LCPx)% = TARGET(LPx) / (# of LCPs_in_LPx) * 100

Following are the definitions of these terms:

WEIGHT(LPx)% This indicates the percentage of the total shared physical CP capacity that will be guaranteed to this LP. This percentage will vary depending on the weights of all the active LPs. In the example, the MVS2 value is 50%, assuming all the LPs are active.

TARGET(LPx) This indicates, in units of shared physical CPs, how much CP resources are guaranteed to the LP. This figure cannot be greater than the number of logical CPs in the LP. This is because you cannot use more physical CPs than the number of logical CPs you have defined—each logical CP can be dispatched on only one physical CP at a time. So, even if there are eight physical CPs available, an LP that has been defined with only four logical CPs can only ever use four of the physical CPs at one time.

If you specify a weight that guarantees you more capacity than can be delivered by the specified number of logical CPs, the additional unusable weight will be distributed among the other LPs.

In our example, MVS2's WEIGHT(LPx)% is 50% of eight physical CPs, meaning that this LP should be given the capacity of four physical CPs.

TARGET(LCPx)% This takes the TARGET(LPx) value (that is, the number of physical CPs of capacity) and divides that by the number of logical CPs defined for the LP. The result determines the percentage of a physical CP that should be given to each logical CP. This is turn determines the effective speed of each logical CP. In our example, the MVS2 value is 4 / 6 * 100, that is, each MVS2 logical CP will be guaranteed 66% of the capacity of a physical CP.

Over time, the average utilization of each logical CP is compared to this value. If Target is less than Current, then the logical CP is taking more CP resources than the guarantee and its priority in the ready queue is decreased. It does *not* mean that it is prohibited from consuming CP, just that it will tend to sit lower in the queue than other logical CPs that have used less than their guaranteed share of the CP resource. Also, these logical CPs are going to be preemptable by an I/O interrupt for a logical CP that is behind its target.

If Target is greater than Current, then the logical CP is taking fewer CP resources than the guarantee and its priority in the ready queue is increased. This means that it has a better chance of being dispatched on a physical CP. Also, these logical CPs are not going to be preempted by an I/O interrupt for another logical CP.

The Current logical CP utilization is shown by RMF in the CPU Activity report (LPAR Busy%) and in the LPAR Activity report (Logical Processors Total). This figure includes the CPU time used by LPAR LIC within the LP.

Now that we have described how logical CP dispatching works, and have explained the use of the logical CP ready queue, it is timely to discuss what is the optimum number of logical CPs for an LP.

In our example configuration, MVS2 has six logical CPs, each of which must get a share of the four physical CPs, as guaranteed by TARGET(LPx). This means that the operating system in MVS2 can run six units of work simultaneously. But it also means that each logical CP does not get a complete physical CP of service (it gets 66%, as we showed previously).

If we change the number of logical CPs that MVS2 has to four, then each logical CP is approximately equal to a physical CP, and will appear to have a higher effective speed. The total amount of CP service the LPAR receives does not change.

When a logical CP does not equal a physical CP of service, the MIPS per logical CP appears to be less than the MIPS of the physical CP. In our example, MVS5 is the only LP where a logical CP equals a physical CP. The apparent processor speed of each LP is shown in the following table:

LPAR Name	Weight	Logical CPs	Percent of physical CPs	Physical CPs per LP	MIPS per logical CP	Total MIPS per LP
MVS2	50	6	50%	4	133	800
MVS3	15	3	15%	1.2	80	240
MVS4	10	2	10%	0.8	80	160
MVS5	25	2	25%	2	200	400
Totals	100	13	100%	8	NA	1600

Each logical CP receives a number of time slices on a physical CP. The MIPS a logical CP delivers depends on how many time slices it receives over a period of time in comparison to the number of time slices available during this time. The number of time slices the logical CP receives depends on the priority in the ready queue, which depends on the LPs's weight and its number of logical CPs.

In our example, each logical CP in MVS2 gets a smaller number of time slices than each logical CP on MVS5. This means that more concurrent units of work can be dispatched in MVS2 than in MVS5, because MVS2 has six logical CPs. However, (assuming that MVS2 and MVS5 are operating at the limit implied by their weights) each logical CP on MVS2 appears to be a slower CP compared to MVS5 because its four physical CPs of service are divided between six logical CPs. MVS5's weight gives it two physical CPs of service divided between its two logical CPs. Therefore, MVS5s CPs appear to be faster than those of MVS2. It's like the old question: "Is it better to have a CPC with a large number of slower CPs, or one with a small number of fast ones?"

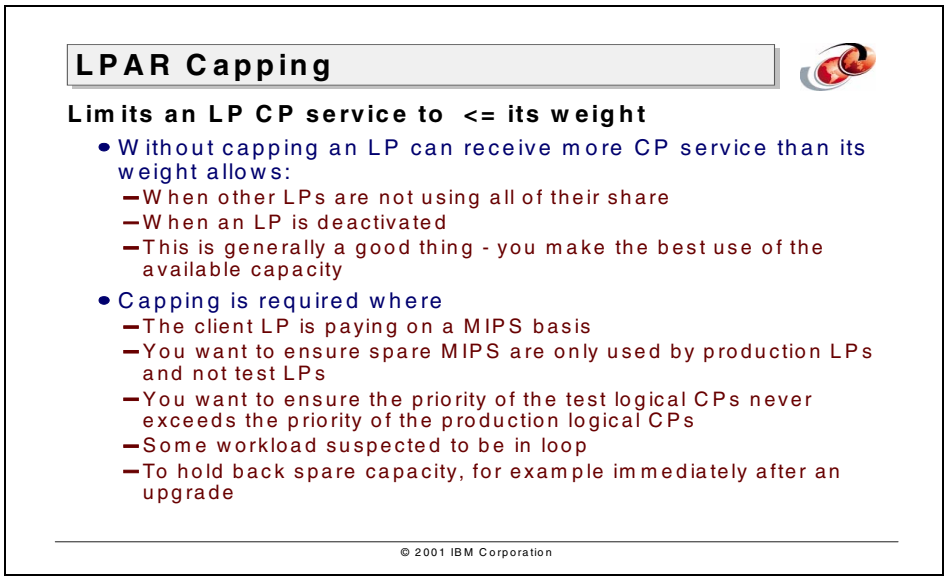
To measure your CP latent demand (the amount of time that your shared logical CPs were ready but could not get the physical CP), you subtract MVS BUSY TIME % - LPAR BUSY TIME%, as shown in the RMF CPU Activity report:

```
OS/390                  SYSTEM ID RMF5              DATE 03/21/00
CPU 9672 VERSION  86    MODEL  R66

 CPU     VF       VF AFFINITY   LPAR BUSY    MVS BUSY   CPU SERIAL
 NUMBER  ONLINE   PERCENTAGE    TIME PERC    TIME PERC    NUMBER

 0       ---       ****         68.49          82.51       027790
 1       ---       ****         67.86          81.65       127790
 2       ---       ****         66.49          79.92       227790
 3       ---       ****         64.58          76.75       327790
 4       ---       ****         62.38          74.35       427790
 5       ---       ****         60.11          71.99       527790
 TOTAL/AVERAGE     ****         64.99          77.36
```

In this example, for 12.37% (77.36 - 64.99) of the RMF interval, this LP had logical CPs that were ready and waiting to execute, but they could not get the physical CPs. If this LP is important to you, you may want to increase the weight, and in some cases increase the number of logical CPs, as well.

It is recommended to use three-digit weight values, as this allows more granular control. Also, for an LP that is part of a sysplex, you should *never* specify a weight that gives that LP less than 5% of the total capacity of the CPC. Experience has shown that LPs with such small weights can have difficulty keeping up with the processing that is required to remain part of the sysplex, which eventually leads to sysplex disruptions.

3.6 LPAR capping



Normally, an LP can get more CP resources than the amount guaranteed by its Target(LPx)—in 3.5, “LPAR weights” on page 59, we discuss how to calculate the Target(LPx) value. Usually, this is a good thing: if there is spare CP resource that is not required by another LP, it makes sense to use it for an LP that needs more CP resource. This can happen when the other LPs are not using all their share or are not active—remember that when an LP is deactivated, the Target(LPx) of the remaining active LPs is re-calculated.

If, for some reason, you want to prevent an LP from ever being able to use more than its Target(LPx), even if there is spare CP resource available, you would use the LPAR capping feature. LPAR capping is implemented by LPAR LIC by observing the CPU resource consumed by a logical CP in a capped LP, and acting if the utilization starts to exceed the logical CPs Target(LCPx). At very frequent intervals (every few seconds or so), LPAR LIC compares the Target(LCPx) of each logical CP of a capped LP with the amount of CP resource it has actually consumed over the last interval. Depending on the result of this comparison, LPAR LIC decides for what percentage of the time in the next interval the logical CP should not be given access to a physical CP.

LPAR capping is a function used to ensure that an LP’s use of the physical CPs cannot exceed the amount specified in its Target(LPx). LPAR capping is set on the processor page for the LP in the HMC Image Profile.

The following are common reasons why people cap LPs. While these may be common, we do not necessarily agree that capping is a good way to achieve your objective in all these cases:

- The LP is being paid for on a MIPS basis.

  With many businesses outsourcing their computing environments, capping becomes an easy way to ensure the customer only gets the CP resource they pay for. However, the capping is implemented by a peak value, a limit that you cannot go beyond, and not as an average.

- You want to isolate test system usage from production usage.

  A test system is often placed on the same CPC as production systems. As with many production systems, their utilization depends on the time of day. Some systems may be used for online processing, while other systems may be used for overnight batch processing. Any spare CP resource is required for the production systems. By capping the test system, you ensure that any available CP resource, above that guaranteed to the test LP, will be available for use by the production systems.

  While this is a popular usage of capping, in our opinion, if you specify the right weights for the test and production LPs, LPAR LIC will ensure that the production LP gets the capacity that has been guaranteed by its weight. Capping the test LP does not let the production LP get more capacity if both LPs are 100% busy—all it does is stop the test LP from using unused cycles above its Target(LPx).

- You want to be able to stop a CF LP from consuming all available CP resources.

  As you probably know, the Coupling Facility Control Code operates in a continuous loop, testing for new messages on its CF links. As a result, a CF LP will normally consume its full share of the CPC, even when there is no real work to be done. If the CF LP is using operating system CPs, as opposed to ICF CPs, this can impact the production LP, stopping that LP from exceeding its Target(LPx).

  However, the solution in this case is not to cap the test CF, but instead to use the Dynamic Dispatching feature in the CFCC. With this, the CFCC does not loop when there is no work. This should only be used for a test CF. Generally speaking, production CFs should use an ICF or a standalone CF, rather than using operating system CPs. The use of Dynamic CF Dispatching generally results in CF response times that are not acceptable for a production CF.

- Some workloads are suspected to be in a loop.

  If you have an application program that is in the habit of looping, but you do not want to cancel it right away (you are not sure) and you do not want this LP to consume more than its share of CP resources, you may use LPAR capping to limit the CP consumption of this LP.

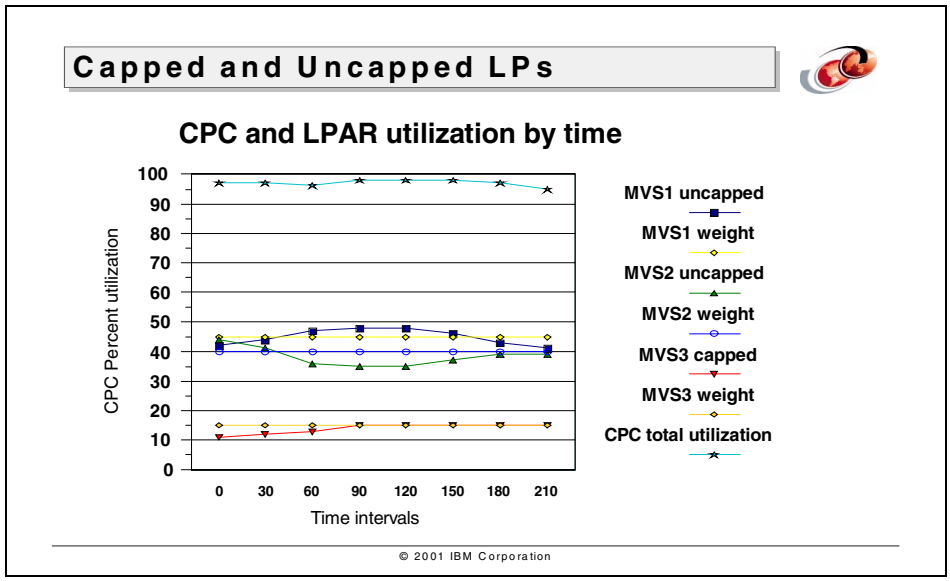
However, there are better ways to control this. The first, obviously, is to fix the application to stop it looping. The other way is to use WLM to cap the service class containing the looping program. Refer to 2.2, “Workload Manager advantages” on page 17, for a discussion about WLM capping. One of the nice things about WLM capping compared to LPAR capping is that WLM capping has finer granularity—you cap just the offending service class, rather than the whole LP.

- Following an upgrade, some installations use capping to avoid a sudden significant improvement in response time, which will then evaporate as the additional capacity gets used up.

  This is a political question rather than a technical one, and therefore is not something that we can comment on.

WLM Goal mode implements capping in its workloads as well, with the added flexibility that are two numbers in WLM; one for protection (smaller), and another for capping (larger), instead of just one as in LPAR capping.

3.6.1 LPAR capped vs. uncapped



The example above shows what happens when there is spare CP resource and the CPC contains LPs which are capped and uncapped. In the example there are 3 LPs:

- MVS1 has a weight of 45 and is uncapped.
- MVS2 has a weight of 40 and is uncapped.
- MVS3 has a weight of 15 and is capped.

The total weights equal 100, therefore each 1 unit of weight is equal to 1% of CPC resource. For example, a weight of 45 equals 45% of the total CPC.

Each LP's weight is shown as a separate straight line in the chart. The total CPC utilization is shown as a separate line close to the top of the chart. This line never exceeds 98%, therefore indicating there is always some spare CPC resource.

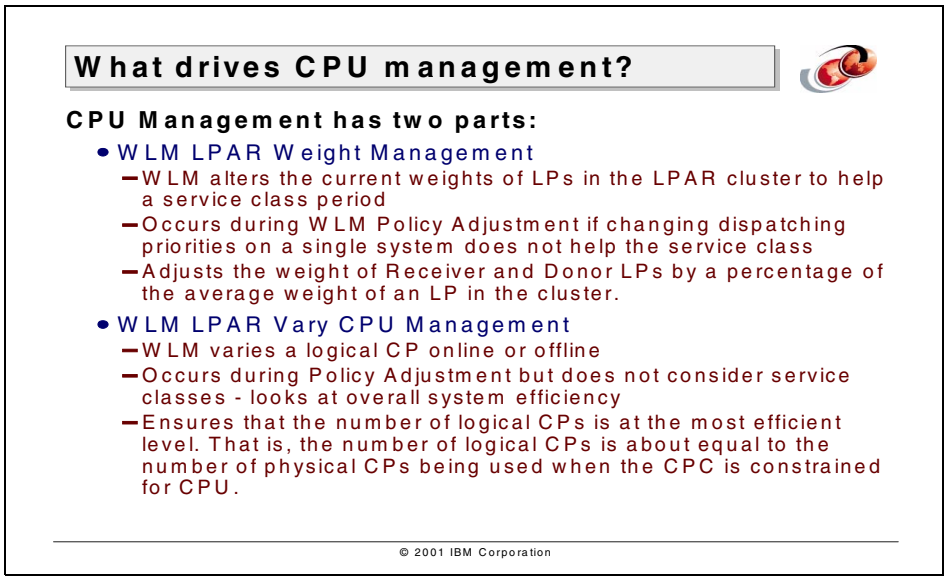
At one time or another, each LP requires less than its guaranteed share of CP resources. Therefore, this spare capacity can be used by either of the uncapped LPs. When an LP uses more than its weight, its utilization line is above its weight line. This occurs for:

- MVS1 from time 60 to 150
- MVS2 from time 0 to 30

For MVS3, which is capped, its utilization line never extends above its weight line. It reaches its weight line and remains there even though there is spare CP resource, as shown by the total CP utilization line being below 100%. A performance impact would be noticed on MVS3 if it required more than its 15% of CP resources.

This completes the review of LPAR shared CP handling. You should now have sufficient knowledge of how LPAR works to be able to understand how the enhancements in WLM LPAR CPU Management are implemented.

3.7 What drives WLM LPAR CPU Management decisions



As has been the goal of WLM for many years, the motivation behind WLM LPAR CPU Management is to provide the resources that transactions need to achieve their goals.

As you are aware, WLM LPAR CPU Management is divided into two parts:

- WLM LPAR Weight Management

  This involves extending the receiver/donor logic for CP delays to CP resource being used by other LPs in the LPAR Cluster. Up until now, WLM could only move CP resource from one SCP to another within the same operating system image, by adjusting the dispatching priority of the receiver and the donor SCPs. When using WLM LPAR Weight Management, WLM can now alter the current weights of LPs within the LPAR Cluster, thereby moving CP resource from one LP to another.

  In the past, LPAR weights could only be changed manually. While it was technically possible to constantly adjust the weights in line with the work in each LP, an operator would first have to determine that an important SCP was missing its goal because the LP did not have sufficient CP resource, and then choose an LP that could afford to give up some weight. This was not practical, and there are probably no installations in the world that manage at this level.

- WLM LPAR Vary CPU Management

  This function does not look at service class periods when making adjustments; rather it attempts to ensure that the available resources are used efficiently. It does this by ensuring that the number of online logical CPs is optimal, based on the number of physical CPs required to do the work.

  This function applies to each LP individually and is controlled via a new keyword (VARYCPU) in the IEAOPTxx Parmlib member. Should you want to enable it for some LPs, but not for others, you must have at least two IEAOPTxx members (one for the systems where it is enabled, and one for the systems where it is not enabled).

  Although WLM LPAR Weight Management and WLM LPAR Vary CPU Management are independent of each other, there is a connection between the two. If WLM LPAR Weight Management changes the weight of an LP, it may be necessary to increase the number of logical CPs so that the LP can actually consume the amount of physical CP guaranteed by the new weight.

  We mentioned this previously, but it is worth stressing that an LP that is capped using LPAR capping cannot take part in WLM LPAR CPU Management. In fact, on the HMC, the buttons for WLM Management and LPAR capping are mutually exclusive—if one is selected, then the other cannot be selected.

The benefits of Vary WLM LPAR CPU Management are:

- Logical CPs are performing at the fastest CP speed available.

  As discussed in 3.5, “LPAR weights” on page 59, as the ratio of logical CPs to physical CPs decreases, the apparent speed of each logical CP increases. For example, if the LP is getting the equivalent of four physical CPs of service and has eight logical CPs online, then each logical CP only gets half of an equivalent physical CP. For example, if a CP delivers 200 MIPS, half of it delivers 100 MIPS. By bringing the number of logical CPs closer to the required number of physical CPs, WLM LPAR Vary CPU Management increases the effective speed of each logical CP. This is not an issue when an LP is running at low utilizations, but becomes more important as an LPs CPU utilization increases towards 100%.

- LPAR overhead is reduced

  There is an LPAR overhead (more work to do) for managing a logical CP. The higher the ratio of logical CPs to physical CPs, the higher the LPAR overhead. By reducing the number of logical CPs in the CPC, WLM LPAR Vary CPU Management reduces the LPAR overhead. Again, this is more important at higher utilizations. If the CPC is only 20% busy, you don’t really care how much CPU is being consumed by LPAR management.

- The LP has improved flexibility and potentially increased capacity.

  Prior to WLM LPAR Vary CPU Management, you had to balance the opposing requirements of minimizing LPAR overhead with providing enough logical CPs to give the LP the capacity it requires. With WLM LPAR Vary CPU Management, you can potentially define each LP with the maximum number of logical CPs (for maximum flexibility), without having to be concerned about the LPAR overhead (because WLM LPAR Vary CPU Management will vary off logical CPs that are not required).

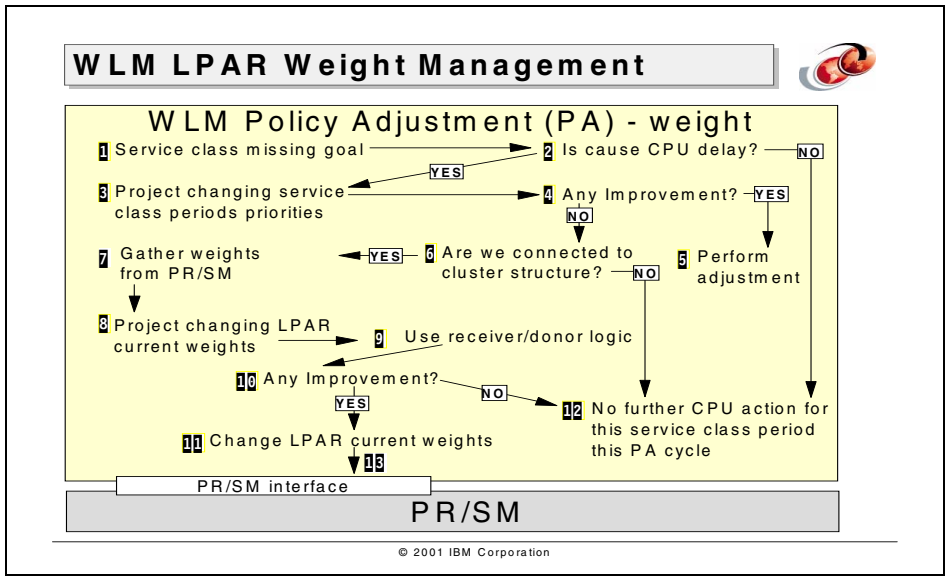
The above benefits are seen on a constrained CPC. On an under-utilized CPC, WLM keeps additional logical CPs online to each LP in the cluster so the workload can take advantage of increased multiprocessing.

Actually changing the LP weights or varying CPs online or offline is simple from an operator perspective, but what is difficult is identifying when the changes are required and whether the changes have had a beneficial effect. By comparison, WLM LPAR CPU Management:

- Identifies what changes are needed and when
- Projects the likely impact on both the work it is trying to help and the work it will be taking the resources from
- Performs the changes
- Analyzes the results to ensure the changes have been effective

There is also the question of the speed at which an operator can perform these actions. WLM may perform these actions every Policy Adjustment interval, which is in the order of several times a minute. It is not possible for an operator to perform all the tasks in this time.

3.8 WLM LPAR Weight Management



WLM LPAR Weight Management helps SCPs meet their goals by moving CP resource from one LP to another by changing the LP's current weights. It extends the existing Policy Adjustment CPU delay routines to include the possibility of moving CP resource to the LP that needs it.

The figure above shows the logic that is used by WLM LPAR Weight Management. The steps are as follows:

1 Select a candidate receiver.

As part of the WLM Policy Adjustment routine, WLM first selects an SCP to be helped. This SCP is known as the *candidate receiver*. Generally, it is the SCP with the highest importance and the highest sysplex performance index. Note that the execution of the Policy Adjustment routine is not synchronized in the WLM instances running in a Parallel Sysplex.

The policy adjustment loop uses the sysplex PI at the beginning of each interval to decide whether there is an adjustment it can take to improve the sysplex PI for the most important service class period. The policy adjustment loop is then performed a second time, this time focusing on the local SCP PI. Sysplex PI evaluation remains the top priority, however, the local PI for a given importance level is also checked before the sysplex PI for the next lower importance, and so forth.

The sequence of checking is:

1. Importance level 1 sysplex PI
2. Importance level 1 local PI
3. Importance level 2 sysplex PI
4. Importance level 2 local PI
5. Importance level 3 sysplex PI
6. And so on.

When WLM identifies an SCP that can be helped by some action on this system, we move to next step.

2 Find the candidate receiver's bottleneck.

Once an SCP has been chosen, WLM must determine what type of delay is causing the problem:

- Is it CPU delay?
- Is it a Storage delay?
- Is it an I/O delay?
- Is the lack of a server address space causing the delay?

If the bottleneck is not a CPU delay, then go to **12**.

If the CPU delay is the major contributor, WLM tries to find (in this system) an SCP to help the receiver. This SCP is known as the *candidate donor*, and must have a PI below 1.0 or be Discretionary (in which case the PI is always determined to be equal to 0.81) and a heavy CPU user. If a donor is found, go to **3**. If not, go to **6**.

3 Find the candidate donor.

The SCP that WLM takes resources from is known as the candidate donor. WLM then projects changing the dispatching priority of the address spaces and enclaves in the SCPs to see if this helps the SCP suffering the CPU delay, but does not hurt the donor SCP, if it is of higher importance. If the donor importance is lower, then it can be hurt. If the donor importance is the same, the donor SCP can also be hurt as long as the change is projected to move the performance indices closer together.

WLM uses resource plots to project a receiver benefit and a donor cost. This calculation derives the projected PIs for each of the SCPs. The net change (the comparison between current PIs and projected PIs) must be favorable before a resource adjustment takes place.

4 Take the decision.

After projecting the PIs, WLM checks for sufficient improvement in the receiver SCP. If there is sufficient improvement, the candidates are declared to be the actual receiver and donor and execution continues with 5. If there is insufficient improvement, go to 6.

5 Change dispatching priorities.

WLM applies the dispatching priority changes to all the address spaces in the SCPs (the donor's dispatching priority is decreased, and the receiver's dispatching priority is increased). And that completes the processing of the receiver SCP for this policy adjustment cycle.

6 Check for LPAR Cluster.

If WLM Policy Adjustment reaches this step, it means that the SCP receiver is suffering because of CPU delay, but no SCP donor was found in this z/OS image. In this case, WLM checks to see if this z/OS image is connected to the WLM LPAR Cluster structure. If it is, go to 7. If not, there is nothing we can do to help this SCP during this Policy Adjustment cycle, so go to 12.

7 Get current weight values from PR/SM.

If the z/OS image is connected to the cluster structure, WLM speaks to PR/SM to get the weights and system names associated with all the LPs in that CPC. The information returned includes:

- Whether the LP is in the same LPAR Cluster.
- Whether the LP has been enabled to use WLM Management.
- Weight information for each LP, including the current weight, the minimum processing weight and the maximum processing weight. This information is assigned by you in the Image Profile on the HMC.

8 Project impact of changing LPAR weights.

Based on the current weights of the LPs in the LPAR cluster, WLM now projects changing the LP current weights to see if this helps the SCP suffering the CPU delay.

To make this projection, WLM must know what work is running in the other LPs in the LPAR cluster, the local PIs for the associated SCPs, and other detailed information about the work in each LP. This information is obtained from the WLM LPAR Cluster CF structure.

9 Verify elapsed time since last change.

Before proceeding, WLM first checks to ensure that some amount of time (we are using one minute in our examples) has passed since the last weight change in the LPAR Cluster. WLM does not proceed if *any* LP in this cluster had its current weight changed within this time. The reason for this checking is to avoid a see-saw effect where too many changes are made to fix a problem, and WLM then has to back off some of those changes. Leaving some amount of time between adjustments gives each change a chance to take effect before a further change is made. There is a situation where the interval can be shortened however—if the SCP being helped is more important than the last SCP helped, or if it is the same importance but with a higher PI than the last SCP helped. This ensures all images in the cluster get a chance to help the most needy work in the cluster.

If it is less than the specified time since the last change, WLM does not proceed. Providing the time has elapsed, the next part of this process is to use the receiver/donor logic in WLM to project the impact on the receiver and donor LPs.

10 Find a candidate weight donor (an SCP and its LP).

WLM determines if increasing the receiver LP's current weight provides any improvement for the receiver. If it does, it then assesses each LP in the LPAR Cluster as a possible donor until it finds one. The LP donor must have an SCP that is a heavy CPU user. In addition, the SCP must meet the donor criteria described in 3 above. It is interesting to note that the other SCPs with work running in the receiver LP may also get a benefit from the change—the additional CP resource is not dedicated to the receiver SCP.

Another part of this process is ensuring that the receiver's increased current weight does not exceed its maximum processing weight and that the donor's new current weight does not fall below its minimum processing weight. Both of these values can bee specified by the installation in the Image Profile (although we recommend leaving them blank so WLM calculates a default value). WLM does not perform weight management (as a receiver) when the LP's current weight has reached the maximum specified on the HMC.

One other point is that a percentage of the average LPAR Cluster weight may be a relatively small percent of the current weight of the receiver, but a much larger percentage of the weight of the donor if the receiver's weight is much larger than the donor. For example, if there are three LPs in the LPAR Cluster—LP1 with a weight of 150, LP2 with a weight of 130, and LP3 with a weight of 20—moving 5% of the average weight (5 weight units) from LP3 to LP1 would only be about a 3% increase for LP1, but a 25% decrease for LP3.

This is not something we need to be overly concerned about as WLM always ensures the donor is left with enough weight units so as not to impact the sysplex PI for SCPs of equal or higher importance to the receiver. However, the sysplex PI of lower importance work may (and probably will) be impacted.

Importance of sensible WLM goals:

Prior to WLM LPAR CPU Management, the WLM policy, with a few exceptions, only affected the relative importance of workloads within a single system. So, if you had separate development and production systems, it didn't really matter if the development DB2 had the same importance as the production DB2. The importance of the DB2 SCP was only relevant compared to the other workloads in the same system. Therefore, it was OK that DB2 was always more important than batch: if the development DB2 had a higher WLM importance than production batch, this made no difference.

However, if you are using WLM LPAR CPU Management, a definition like this may not work as you intend. What could potentially happen now is that an SCP on the development system could end up taking weight from a more important (but defined with equal or less Importance in WLM) SCP in the production LP. For this reason, you must make sure that the WLM Importances that you assign to SCPs actually make sense at the sysplex level. This is discussed further in 4.2, "WLM Policy definitions" on page 105.

If there is insufficient improvement or a suitable donor LP cannot be found, go to **12**. If there is sufficient improvement and a donor LP is found, go to **11**.

11 Change LP weights.

Having decided on the donor and receiver LPs, WLM now adjusts the current weights of those LPs. Go to **13**.

12 No further CPU action.

If WLM reaches this step, there is nothing else it can do to alleviate any CPU delays for this SCP in this Policy Adjustment cycle. However, WLM will now use the appropriate delay processing algorithms for either storage or I/O delay to try and help this SCP if the problem is caused by one of those delays.

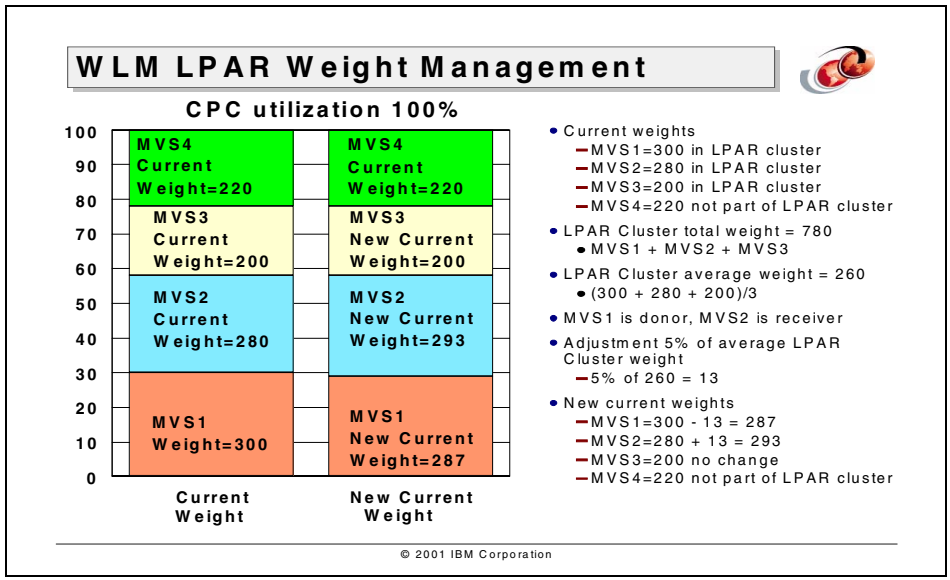
13 WLM uses a new LPAR interface to alter the current weights of the receiver and donor LPs. This interface does the following:

- It is used to change the current weights for the receiver and donor LPs.

- It ensures that the total weight of the LPAR cluster does not change.

This completes the processing performed during the Policy Adjustment routine for attempting to help an SCP that is delayed because it is waiting for CPU.

3.8.1 WLM LPAR Weight Management example



The example above shows how WLM LPAR Weight Management works in a three-system LPAR Cluster. There are four systems on the CPC: MVS1, MVS2 and MVS3 are part of the LPAR Cluster, while MVS4 is not (it is not in the same sysplex, or is not z/OS, for example). The three systems in the cluster have the following current weights:

1. MVS1 has a current weight of 300.
2. MVS2 has a current weight of 280.
3. MVS3 has a current weight of 200.

The total weight of the LPAR Cluster is 780, which is the total of the three LPs in the LPAR Cluster:

300 + 280 + 200 = 780

The LPAR Cluster total weight must remain the same after any weight adjustments.

An important SCP has a sysplex PI greater than 1 and the problem has been determined to be a CPU delay on MVS2. WLM tries to increase MVS2's current weight by 5% of the average LPAR Cluster weight, which in this case is 13 weight units. The weight units can come from MVS1 or MVS3. In this case, the weight units are taken from MVS1.

So we see the new current weights as:

1. MVS1 has a new current weight of 287.
2. MVS2 has a new current weight of 293.
3. MVS3 has the same current weight of 200 as before.

It is important to note that the total weight of the LPAR Cluster (780) has not changed. We see this by adding the new current weights of the three systems in the LPAR Cluster:

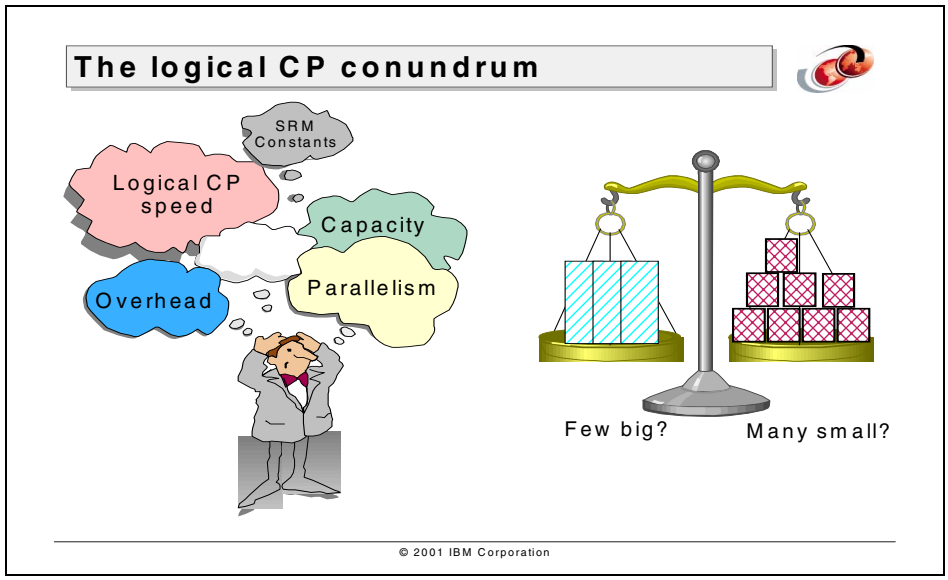
287 + 293 + 200 = 780

Although the example shows the processing at the LP level, it is important to remember that the trigger for the weight adjustment is always based on helping an SCP. As a consequence of this, the LP gets more CP resource. As a by-product of the change, other work running in the receiver LP may also get additional resource, also known as a “free ride”.

Another thing to consider is that WLM LPAR Weight Management may decide to increase the weight based on either the sysplex or the local PI of the SCP. For example, if an important SCP has a sysplex PI that is greater than 1, WLM LPAR Weight Management may decide to increase the weight of that LP even if the local PI is less than 1. The logic being that the overall sysplex PI for the SCP may decrease (that is, improve) if this LP can provide even better service to that SCP. Similarly, even if the sysplex PI of an important SCP is less than 1, WLM LPAR Weight Management may decide to increase the LP’s weight if the local PI of the SCP is greater than 1.

The current weight of the fourth system, MVS4, which is not in the LPAR Cluster, is not considered and remains constant. Of course, the CP resources not consumed by MVS4 may still be used by the LPs in the LPAR Cluster; this is independent of whether those systems are in an LPAR Cluster or not.

3.9 WLM LPAR Vary CPU Management



WLM LPAR Vary CPU Management addresses a question that Systems Programmers have pondered since the introduction of CPCs with large numbers of CPs: how many logical CPs should you define in each LP?

The best answer is one you might have heard before—it depends! The reason is that at one point in time, the correct answer is five logical CPs. But a few minutes later it may be three, and two logical CPs a few minutes after that. The following is a list of the variables that contribute to getting the “right” answer:

- LP weight assigned value. The number of logical CPs should not be less than that implied by the Target(LPx), that is, the number of physical CPs guaranteed to the LP based on the weight that you assigned. In other words, if the weight indicates that you wish the LP to have 50% of an 8-way CPC, don’t define the LP with just 2 logical CPs. Refer to 3.5, “LPAR weights” on page 59, for more information on Target(LPx).
- LPAR LIC overhead. The higher the ratio of logical CPs to physical CPs, the higher is this overhead. However this is not a concern at low CPU utilizations.
- LP capacity. If you want to limit the capacity of an LP, you can define a smaller number of logical CPs.
- The number of physical CPs actually used by the LP. The number of logical CPs should be close to the number of *used* physical CPs, in order to provide the highest effective logical CP speed. Again, this is more relevant at higher

CPU utilizations. If the LP is only using a small percent of its timeslice each time it is dispatched, the logical CP speed is not as much of an issue.

The term *used* in the previous sentence should be understood as the guaranteed amount (Target(LPx)), if this is more than the amount actually consumed, or the amount actually consumed, if this is larger than the guaranteed amount. In other words, it is the greater of the consumed capacity and the guaranteed capacity.

The following example shows how the logical CP speed is affected by changing the number of logical CPs in an LP: Let us assume that we have an LP that can consume the equivalent of four physical CPs (200 MIPS each) of service and is defined with eight logical CPs. In order for the eight logical CPs to execute on the four physical CPs, each logical CP only gets half of a physical CP, and therefore appears to execute at half the speed of the physical CP (100 MIPS). This occurs because each logical CP gets fewer time slices. If you defined the LP to have the same weight (and thus be guaranteed 800 MIPS), but only four logical CPs, each logical CP would now appear to be 200 MIPS.

It is important to note, however, that the captured CPU time per transaction does not reflect the slower logical CP speed. The reason for this is that z/OS uses the CPU Timer to account for the use of CPU by its dispatchable units, and when the logical CP is not dispatched on a physical CP, that clock logically stops. The same applies to the numeric value of the SRM constant (CPU services units per second), which does *not* change when WLM LPAR Vary CPU Management adds or removes a logical CP.

- The level of parallelism. Some workloads, such as non-MRO CICS or CPU-intensive batch jobs, do most of their work under a single TCB, and therefore operate better on a smaller number of faster CPs. Other workloads, such as IMS, tend to split their work across many TCBs, and therefore operate better in an environment with a larger number of CPs.

  Having said this, most z/OS systems support many workload types—CICS, IMS, batch, TSO, Business Intelligence, Web serving, and so forth—and therefore it is not as easy to say whether a particular system operates better with more or fewer CPs. For a medium to large system (500 MIPS+), it probably balances out over a 24-hour period, whichever configuration you choose.

3.9.1 WLM LPAR Vary CPU Management concepts

WLM LPAR Vary CPU Management tries to determine the appropriate number of logical CPs in an LP, aiming for high logical CP speed and less overhead, while also providing maximum flexibility. When we say an “appropriate” number of CPs, we mean:

# of online logical CPs in an LP = # of used physical CPs in LP + Z

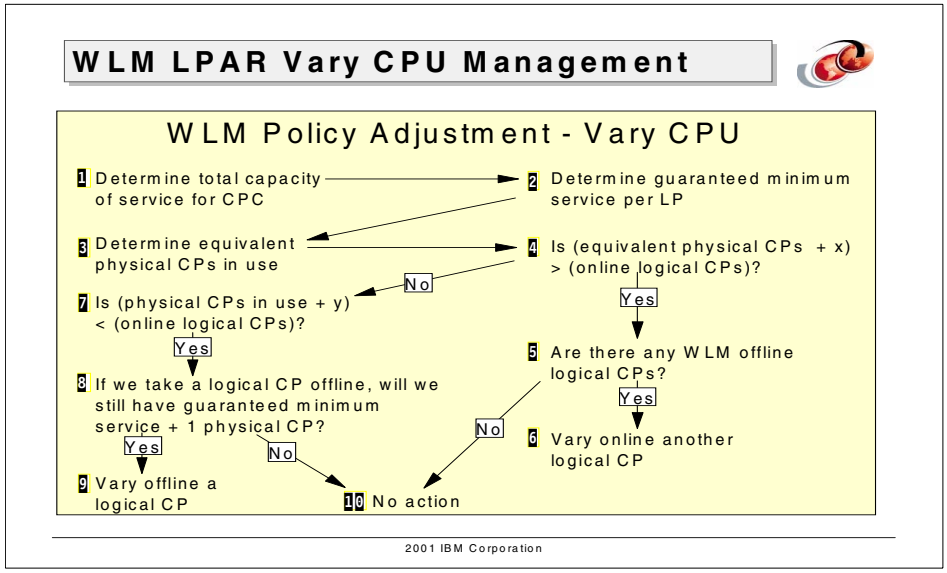
Where Z is a buffer, the size of which varies in accordance with the number of currently online logical CPs *(*refer to 3.9.2, “WLM Vary CPU Management logic” on page 83, for more details). The idea of providing more CPs than are currently needed (the Z value) is to provide a buffer in case the workload’s CP requirement increases rapidly. This value is large enough to provide a safe buffer without significantly decreasing the effective logical speed or increasing the overhead.

With the availability of WLM LPAR Vary CPU Management, the answer to “how many logical CPs to define” becomes simple—define as many as you are ever likely to need. For most production LPs, this generally means specifying an Initial number (in the HMC Image Profile) equal to the number of shared CPs currently installed on the CPC. For development or test LPs, you may wish to limit the consumption of those LPs by specifying a smaller number of CPs.

Note that the maximum number of logical CPs that can be online in an LP is the total of the Initial and Reserved values defined in the HMC (assuming that many shared physical CPs are available). When an LP is activated, it will come up with the number of logical CPs that are defined in the Initial field. If they are available, an Operator may configure online additional logical CPs, up to the value specified in the Reserved field. WLM might reduce the number of online logical CPs in reaction to the workload in the LP, but it can never *exceed* this number (that is, WLM will never vary on a logical CP from the Reserved pool unless that CP was first configured online manually).

A characteristic of WLM LPAR Vary CPU Management is to be more aggressive at bringing a logical CP *online* than when it is taking one *offline*. This is to balance the needs of responsiveness with efficiency. You should have a buffer to allow for growth, and should also be careful not to overreact to a temporary dip in system utilization. As a result, WLM LPAR Vary CPU Management uses a larger buffer when deciding if it should take a CP offline than when deciding to add one.

3.9.2 WLM Vary CPU Management logic



The figure above shows the logic used by WLM LPAR Vary CPU Management when deciding whether to add or remove a logical CP. These steps are as follows:

1 Determine the total capacity of service for the entire CPC. WLM converts the total capacity from physical CPs to CPU service units per second (service rate), so it can use this in determining how many physical CPs of service an LP is using in later calculations. The following formula is used:

```
Total capacity = Service_Rate_per_CP * Total_physical_CPs_on_CPC
```

Where Service Rate per CP is based on the SRM constant.

2 Determine the guaranteed minimum amount of service an LP should get, based on its current weight. To get this number, the Weight(LPx)% figure is used. In 3.5, “LPAR weights” on page 59, we describe how this value is calculated.

WLM totals the current weights of all active LPs, including LPs which are not in the LPAR cluster. This is because an LP that is not in the cluster is still using CP resource that will not be available to the cluster.

It is important to note that an LP must be active (not necessarily IPLed) for its current weight to be considered when determining the total weight of all LPs on the CPC. This is the minimum percentage of total capacity of the CPC that this LP must get based on its current weight, when all LPs are running at 100%. It is also effectively the maximum amount of service the LP will get if all LPs are running at 100%. The guaranteed minimum service is calculated by the formula:

```
Guaranteed       (LP current weight)
minimum service = ------------------------------  * Total capacity
                    (Total weight)
```

This calculation uses the current weight of an LP. This is because the LP current weight may have been changed by WLM LPAR Weight Management since it was IPLed.

3 Determine the equivalent number of physical CPs being used by the LP or being guaranteed to the LP. This calculation converts the amount of service actually being used into an equivalent number of physical CPs. If the number being used is less than the guaranteed minimum, then use the guaranteed minimum instead. The following calculation is used:

```
Equivalent       Max(CP service used,guaranteed minimum service)
physical CPs  =  ----------------------------------------------
                 (CP service per physical CP)
```

The number of equivalent physical CPs is the value used to determine whether an LP has too few or too many logical CPs online. CP service in the above formula means CP service rate.

4 Determine if there are too few logical CPs online. The following calculation is used:

```
Is (Equivalent physical CPs + x) > (online logical CPs)?
```

Where x is a number determined by WLM. This is an aggressive target. When an LP is using all but the buffer amount of its online logical CPs, WLM varies another one online. For example, if the buffer were 1 logical CP, this would ensure that the LP always has 2 logical CPs available as spare capacity.

If the answer is yes, go to step **5**. If the answer is no, go to step **7**.

5 There are too few logical CPs online. Are there currently any WLM offline logical CPs for this LP? A WLM offline logical CP is one which has been taken offline by WLM.

Note: If a logical CP has been taken offline by an operator using the CF CP(xx),OFFLINE command, it cannot be varied online by WLM. Similarly, the operator is not allowed to bring online any CP which is currently taken offline by WLM.

If the answer is yes, go to step 6. If the answer is no, go to step 10.

6 Vary another logical CP online.

7 Determine if there are too many logical CPs online. The following calculation is used:

```
Is (Equivalent physical CPs + y) < (online logical CPs)?
```

Where y is a number determined by WLM. This is not an aggressive target. WLM allows a system to have a significant amount of spare capacity before it takes a logical CP offline.

If the answer is yes, go to step 8. If the answer is no, go to step 10.

8 WLM always leaves enough logical CPs online so the LP can use its whole Target(LPx) + 1 logical CP. This step ensures that we abide by this rule before taking a logical CP offline. If the answer is yes, go to step 9. If the answer is no, go to step 10.

9 There are too many logical CPs online, so vary 1 logical CP offline.

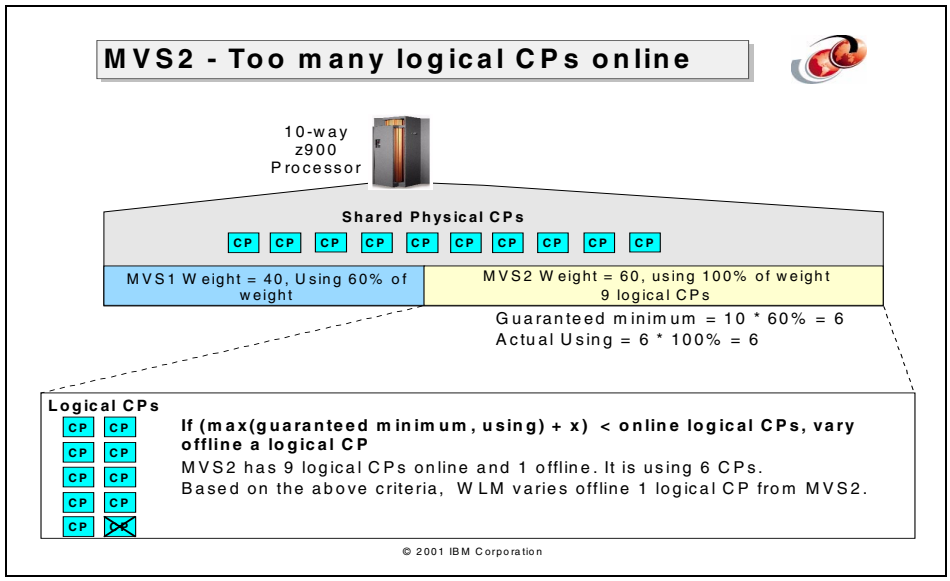
10 No action is taken.

As a final action, and if the following conditions are met, WLM will bring (or leave) an additional logical CP online:

- Overall CPU utilization is low
- The number of logical CPs is less than the maximum allowed for this LP

The reason for this action is to allow the LP to benefit from a higher degree of multiprocessing. Although this results in increased LPAR overhead, at lower CPU utilizations the overhead is an acceptable tradeoff for the benefits of increased multiprocessing. A result of this action is that you may see that WLM does not start decreasing the number of logical CPs immediately after an LP is IPLed. Because CPU utilization is low, it may leave additional logical CPs online. As CPU utilization increases, it will start configuring logical CPs offline until you get to the point that the number of logical CPs starts approaching the number of equivalent physical CPs guaranteed to the LP. If the utilization continues to increase, it will configure logical CPs back online again to provide the required capacity.

3.9.3 Example - Too many logical CPs online



We now provide three examples to show how this logic works in practice.

In the example above, we have a 10-way CPC with 2 LPs. MVS1 has a current weight of 40 (and 10 logical CPs defined in the HMC) and MVS2 has a current weight of 60 (and also has 10 logical CPs defined in the HMC). The following is a summary of the current status of the MVS2 LP:

- It is defined with 10 logical CPs. It currently has 9 online and 1 offline by WLM.
- It is using 600 CPU service units per second, and the service rate per CPU of this CPC is 100 per second.
- Therefore it is consuming 100% (600/600) of its current Target(LPx) - that is, 6 physical CPs.
- The guaranteed minimum service for MVS2 is 600 (using the service rate per CPU of 100), which is also equivalent to 6 physical CPs.

  This is the amount of service equivalent to 100% of the LPs current weight. This is arrived at by using the calculations used in 1, 2 and 3 of 3.9, “WLM LPAR Vary CPU Management” on page 80.

```
1 Total capacity = Service rate per CPU * Total physical CPs on CPC
                 = 100 * 10
                 = 1000 service units/sec
```

```
2 Guaranteed        (LP's current weight)
  minimum service = ------------------------------  * Total capacity
                     (Total weight)

                  = 60 / 100 * 1000
                = 600 service units/sec

3 Equivalent       Max(CPU service used,guaranteed minimum service)
  physical CPs  =  ------------------------------------------------
                  (CPU service rate per physical CP)

                = Max(600,600) / 100
                = 6 physical CPs
```

- The higher of used capacity and guaranteed capacity (both the same, in this case) is the equivalent of 6 physical CPs of service (based on 100% of its current weight).
- For the sake of this example, assume the buffer value used by WLM when deciding if it should add a logical CP is 1, and the buffer used when deciding if it should remove a logical CP is 2. Using this value then, WLM calculates that this LP should have at least 7, but not more than 8 logical CPs online.

Based on this summary of the current environment, WLM LPAR Vary CPU Management uses the following calculations to decide if it should add a CP, remove a CP, or do nothing:

1. Is (MVS2 equivalent physical in use CPs + 1) > (online logical CPs)?

   Is (6 + 1) > (9)? The answer is no.

2. Is (MVS2 equivalent physical CPs in use + 2) < (online logical CPs)?

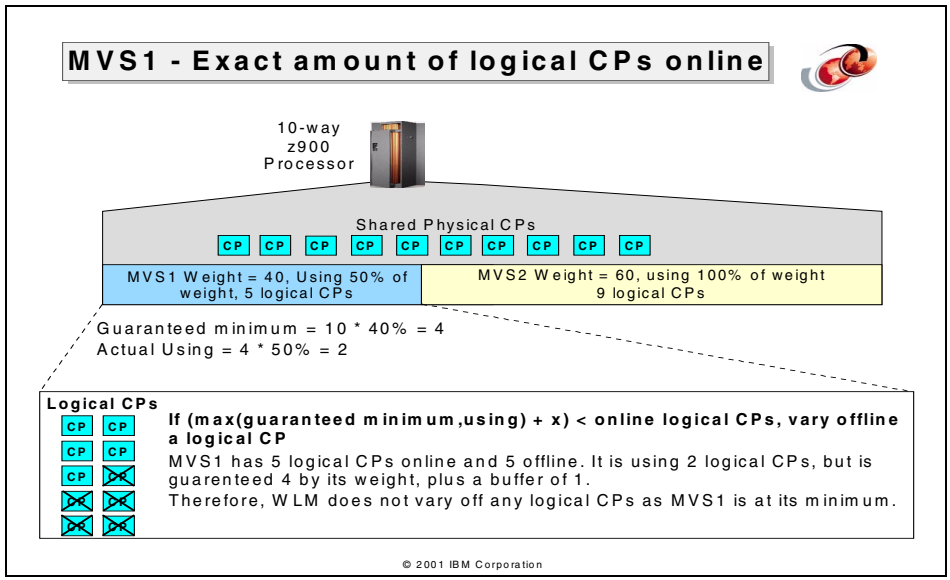
   Is (6 + 2) < (9)? The answer is yes.

3. If we take 1 logical CP offline, do we still have guaranteed minimum service (or minimum number of logical CPs online + 1 logical CP)?

   Is (9 - 1) > 7? The answer is yes.

4. Therefore WLM varies offline 1 logical CP from MVS2.

3.9.4 Example - Exact amount of logical CPs online



For the next example, we use the same configuration and the same example buffer values, but look at LP MVS1 this time. Once again, we have a 10-way CPC with 2 LPs. MVS1 has a current weight of 40 and MVS2 has a current weight of 60. The following is a summary of the current status of the MVS1 LP:

- It is defined with 10 logical CPs. It currently has 5 online and 5 offline.
- It is using 200 CPU service units per second, and the service rate per CPU of this CPC is 100 per second.
- Therefore, it is using 50% of its current Target(LPx) - that is, 2 physical CPs.
- The guaranteed minimum service for MVS2 is 400 CPU service units per second (using the service rate per CPU of 100), which is equivalent to 4 physical CPs.

  This is the amount of service equivalent to 100% of the LPARs current weight. This is arrived at by using the calculations used in **1**, **2** and **3**of 3.9, “WLM LPAR Vary CPU Management” on page 80.

```
1 Total capacity = Service per CPU * Total physical CPs on CPC
                 = 100 * 10
                 = 1000


2 Guaranteed          (LP's current weight)
  minimum service = ------------------------------  * Total capacity
```

```
                    (Total weight)

                = 40 / 100 * 1000
                = 400

3 Equivalent      Max(CPU service used,guaranteed minimum service)
  physical CPs =  ------------------------------------------------
                  (CPU service per physical CP)

              = Max(200,400) / 100
              = 4
```

- It is actually using the equivalent of 2 physical CPs of service (based on 50% of its current weight).
- It must have 5 logical CPs online as the minimum number of logical CPs online.

  The minimum number of logical CPs that WLM leaves online to an LPAR is the guaranteed minimum service (expressed as the number of equivalent physical CPs) plus 1 CP.

Based on this summary of the current environment, WLM LPAR Vary CPU Management uses the following calculations to decide if it should add a CP, remove a CP, or do nothing:

1. Is (MVS1 equivalent physical CPs in use + 1) > (online logical CPs)?

   Is (2 + 1) > (5)? The answer is no.

2. Is (MVS1 equivalent physical CPs in use + 2) < (online logical CPs)?

   Is (2 + 2) < (5)? The answer is yes.

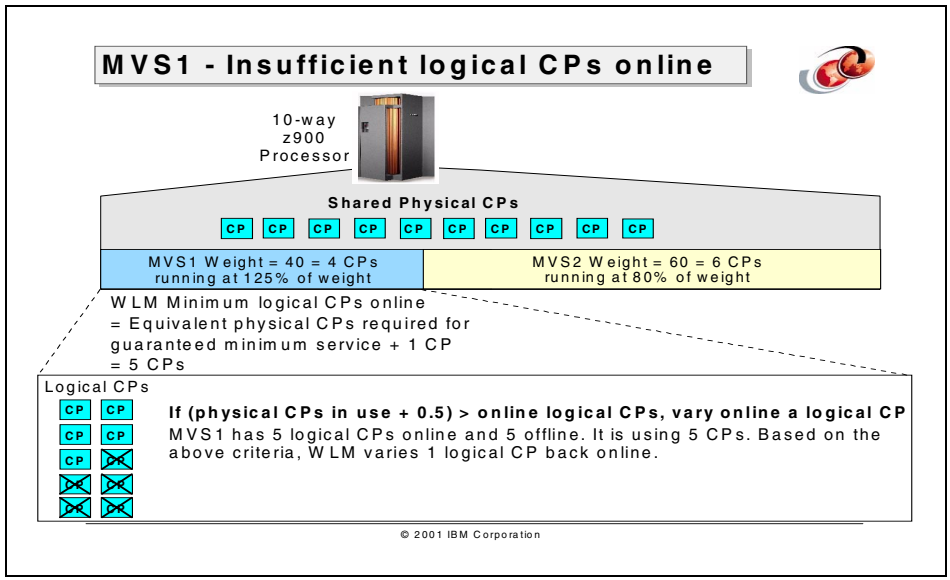
3. If we take 1 logical CP offline, do we still have guaranteed minimum service (or minimum number of logical CPs online + 1 CP)?

   Is (5 - 1) > 5? The answer is no.

4. Therefore, even though MVS1 has 3 more logical CPs online than it is currently using, WLM does not take a CP offline. The minimum number of logical CPs online + 1 CP, which is 5 CPs, must remain online.

Note that in this example, if MVS1 continued to use less than its weight, and MVS2 required more CP resources, the weight of MVS1 would eventually be decreased (it would be moved to MVS2 by WLM LPAR Weight Management), resulting in a smaller number of online logical CPs.

3.9.5 Example - Too few logical CPs online



For our final example, we once again look at LP MVS1, some time later in the same day. Once again, we have a 10-way CPC with 2 LPARs. MVS1 has a current weight of 40 and MVS2 has a current weight of 60. The following is a summary of the current status of the MVS1 LP:

- It is defined with 10 logical CPs. It currently has 5 online and 5 offline.
- It is using 500 CPU service units per second, and the service rate per CPU of this CPC is 100 per second.
- Then it is using 125% of its current Target(LPx) - that is, 5 physical CPs.
- The guaranteed minimum service for MVS2 is 400 service units per second (using the service rate per CPU of 100), which is equivalent to 4 physical CPs. This is the amount of service equivalent to 100% of the LP's current weight. This is arrived at by using the calculations used in 1, 2 and 3 of 3.9, "WLM LPAR Vary CPU Management" on page 80.

```
1 Total capacity = Service per CPU * Total physical CPs on CPC
                 = 100 * 10
                 = 1000

2 Guaranteed          (This partition current weight)
  minimum service = ------------------------------  * Total capacity
                          (Total weight)

                  = 40 / 100 * 1000
```

```
                = 400

3 Equivalent       Max(CPU service used,guaranteed minimum service)
  physical CPs  =  ------------------------------------------------
                          (CPU service per physical CP)

                = Max(500,400) / 100
                = 5
```

- It is using the equivalent of 5 physical CPs of service (based on 125% of its current weight).
- Therefore, it should have 6 logical CPs online as the minimum number of logical CPs online.

Based on this summary of the current environment, WLM LPAR Vary CPU Management uses the following calculations to decide if it should add a CP, remove a CP, or do nothing:

1. Is (MVS1 equivalent physical CPs + 1) > (online logical CPs)?

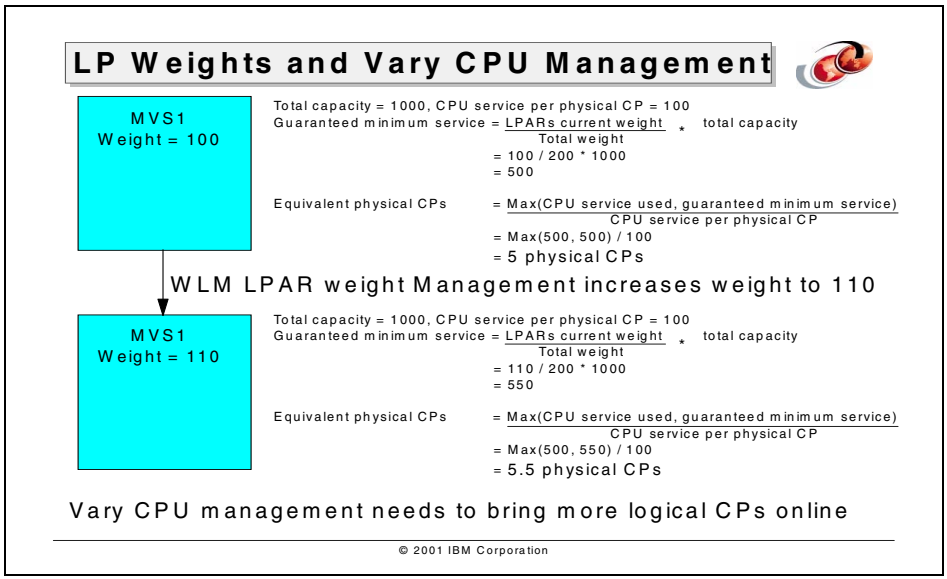
   Is (5 + 1) > (5)? The answer is yes.

2. Are there any offline logical CPs for this LPAR?

   The answer is yes.

3. Therefore WLM varies online 1 of the logical CPs that it previously varied offline.

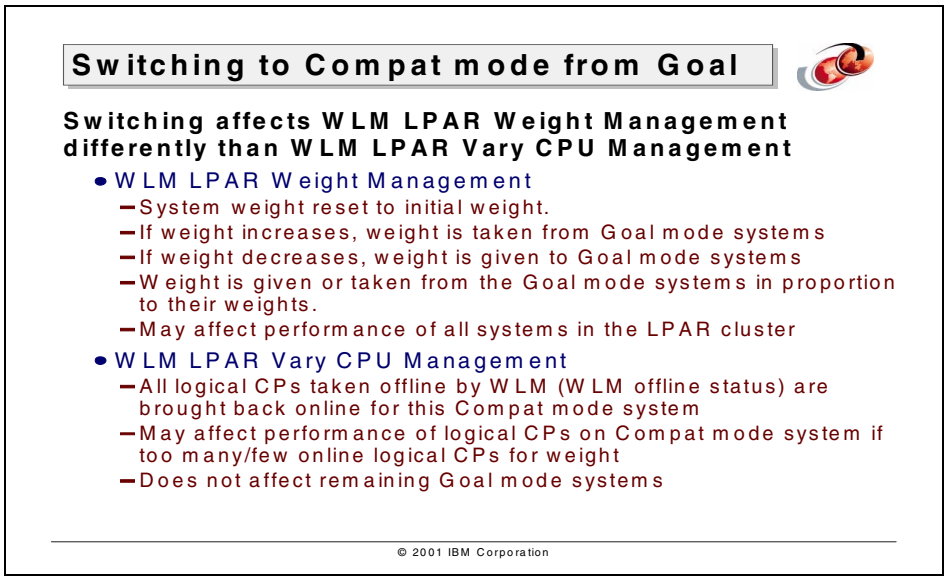
3.10 Effect of WLM Weight Management on WLM Vary CPU Management



The above example shows WLM LPAR Weight Management changing the current weight of an LP from 100 to 110. As a result, the guaranteed minimum number of CPs will also increase—from 5 to 5.5.

This figure is used by WLM LPAR Vary CPU Management to determine whether more logical CPs should be brought online. It shows how an increase in the current weight of an LP may cause an additional logical CP to be varied online by WLM LPAR Vary CPU Management. This is precisely what you wish to happen - in order for the LP to use the additional CP resources, it needs the appropriate number of logical CPs online. There would be no point in increasing the LP weight to give it the equivalent of 6 CPs of service if it only had 5 logical CPs online. You cannot get more CP service than you have logical CPs online. Therefore in this case, a WLM offline logical CP is brought back online.

3.11 Switching to WLM Compatibility mode



Before we discuss the impact of switching from WLM Goal mode to WLM Compatibility mode, we just want to remind you that the base level for WLM LPAR CPU Management is z/OS 1.1, and that IBM has announced that z/OS 1.2 will be the last release to support WLM Compatibility mode. So the ability to be running WLM LPAR CPU Management and revert back to Compatibility mode will only exist for two releases of z/OS.

Switching a system in an LPAR cluster from WLM Goal mode to WLM Compatibility mode affects not only the system whose mode is being changed, but also has an affect on other systems in the LPAR cluster. We first look at the impact on WLM LPAR Weight Management as this affects not only the system whose mode is being changed, but potentially the remaining WLM Goal mode systems in the LPAR cluster.

When a system is switched into WLM Compatibility mode, WLM resets its current weight back to its initial processing weight as specified in the Image Profile on the HMC. If this involves increasing or decreasing its current weight, WLM needs to alter the current weights of the other LPs in the LPAR cluster accordingly. This is because the total weight in the LPAR cluster must remain constant. In a two-system LPAR cluster, this is simple. If WLM increases one LP current weight

by 10 weight units, then WLM must decrease the other LP current weight by 10 units. In a three or more system LPAR cluster, the weight units are divided proportionally between the other LPs. The following table provides some examples. It shows three separate clusters:

- Cluster 1 has three systems - MVS1, MVS2, MVS3. MVS1 is placed into Compatibility mode.
- Cluster 2 has three systems - MVS4, MVS5, MVS6. MVS5 is placed into Compatibility mode.
- Cluster 3 has two systems - MVS7, MVS8. MVS7 is placed into Compatibility mode.

The indicated weight changes in the Goal mode LPs are only approximate. In each case, the system marked with an asterisk (*****) is changed from WLM Goal mode to Compatibility mode:

Cluster/ System Name	Initial Processing Weight	Current Weight	Current Weight after switch to compat mode	Weight change (+/-)
LPAR CLUSTER 1				
MVS1 *	100	80	100 **Compat mode**	+20
MVS2	50	80	66 Goal mode	-14
MVS3	50	40	34 Goal mode	-6
LPAR CLUSTER 2				
MVS4	70	70	81 Goal mode	+11
MVS5 *	50	70	50 **Compat mode**	-20
MVS6	80	60	69 Goal mode	+9
LPAR CLUSTER 3				
MVS7 *	60	30	60 **Compat mode**	+30
MVS8	40	70	40 Goal mode	-30

In LPAR cluster 1, MVS1 is switched to Compatibility mode. Its current weight before the switch was 80. Its initial processing weight was 100, therefore its current weight must be increased by 20. WLM takes the 20 weight units from systems MVS2 and MVS3. MVS2 has a higher current weight than MVS3, therefore MVS2 loses more weight than MVS3. However, proportionally, they both lose approximately the same percentage of their current weight. In this case it is in the range 15% to 17%.

In LPAR cluster 2, MVS5 is switched to Compatibility mode. Its current weight before the switch was 70. Its initial processing weight was 50, therefore its current weight must be decreased by 20. WLM gives the 20 weight units to systems MVS4 and MVS6. MVS4 has a higher current weight than MVS6, therefore MVS4 gains more weight than MVS6. However, proportionally, they both gain approximately the same percentage of their current weight. In this case it is about 15%.

In LPAR cluster 3, MVS7 is switched to Compatibility mode. Its current weight before the switch was 30. Its initial processing weight was 60, therefore its current weight must be increased by 30. As there only 2 LPs in this cluster, WLM takes the 30 weight units from MVS8. This is the simplest case.

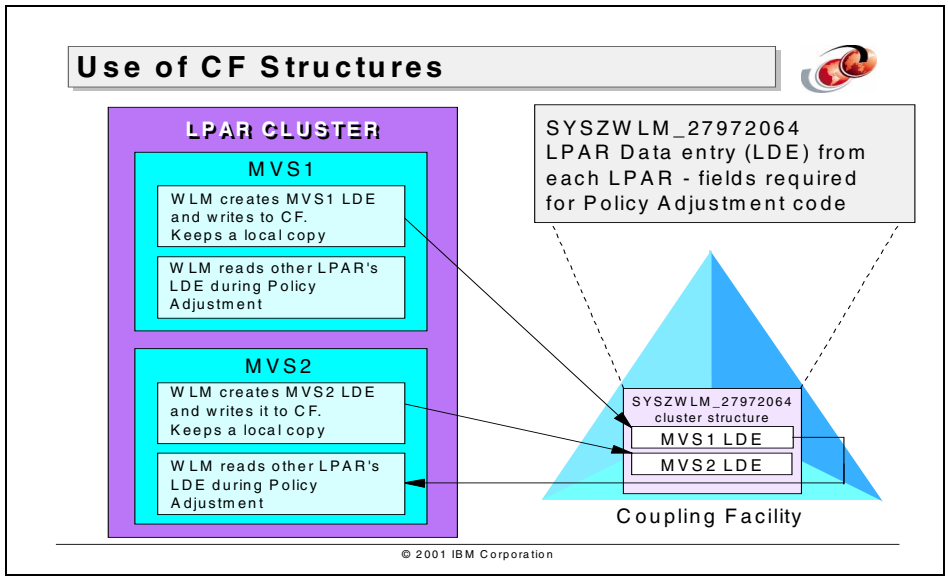
The impact to the other systems in the LPAR cluster depends on whether we need to take or give weight units to the remaining Goal mode systems. If WLM is giving weight units to the Goal mode systems, then these systems have extra capacity. This is not really a concern. However, if WLM needs to take weight units from the other systems, this could cause an impact as suddenly these systems have fewer CP resources.

Whether these systems are impacted or not depends on how busy they are and how many weight units they lose. If they are not running at 100%, there may be no impact. If they are running at 100% and only lose a small number of weight units, perhaps only low priority work suffers. If they are running at 100% and they lose a large number of weight units, they may be more severely impacted.

Next, we look at what occurs for WLM LPAR Vary CPU Management. When a system is switched into Compatibility mode, all logical CPs that were taken offline by WLM (referred to as WLM offline CPs) are brought back online. If the LP has sufficient current weight to use all the online logical CPs, then there is no impact. If, however, the LP does not have sufficient current weight, each logical CP will perform as a slower CP. For example, if your current weight entitles you to four physical CPs of service and it is now divided between eight online logical CPs, then each logical CP can only get CP service equivalent to half a CP. This is explained in 3.5, “LPAR weights” on page 59.

There is no impact to the number of online logical CPs of the systems in Goal mode by the system that switches to Compatibility mode.

3.12 Use of CF structures



Ever since WLM was introduced in MVS/ESA 5.1, all the WLMs in the sysplex exchange global data (such as response time distributions, using and delay figures, and so on) with each other using XCF. This process continues unaffected, regardless of whether Intelligent Resource Director is being used or not. This process did not use a Coupling Facility (CF) structure.

However, if you are using WLM LPAR CPU Management or Dynamic Channel-path Management, the amount of data that is shared between the systems increases significantly. As a result, WLM now uses a CF structure to share detailed information about the SCPs, the local PIs, and the resource usage of each of the SCPs in each system. In a similar manner, WLM also uses a CF structure to maintain information about resource usage by multi-system enclaves.

Each WLM instance in an LPAR cluster places this information in a control block called an LPAR Data Entry (LDE) during Policy Adjustment processing and places the LDE in a new WLM CF structure. This gives WLM in each system access to LPAR cluster-wide data required to project the impact of changing the weight of an LP.

Using this information, for example, WLM in LP MVS1 knows that the significant user of CPU in LP MVS2 is an SCP that has a higher importance than the SCP that WLM in MVS1 is trying to help—as a result, WLM in MVS1 will try to find a different donor.

If one WLM instance loses connectivity to the CF, then the WLM LPAR CPU Management functions will not be available on that system until connectivity is re-established. In this case, WLM performs as it did prior to WLM LPAR CPU Management. This means WLM only redistributes CPU resources within that system. In this case, you should regain connectivity for that system or manually rebuild the structure to a CF that has connectivity to all systems in the LPAR cluster. As soon as the system regains connectivity to the structure, either because you fixed the broken link or because you rebuilt the structure to another CF, the WLM LPAR CPU Management functions become available again automatically.

However, the other instances of WLM in the LPAR Cluster, which do not lose connectivity to the structure, go on doing weight management, but only among themselves—because those systems do not have detailed information about the SCPs in the system that lost connectivity, they cannot adjust the weight of that LP.

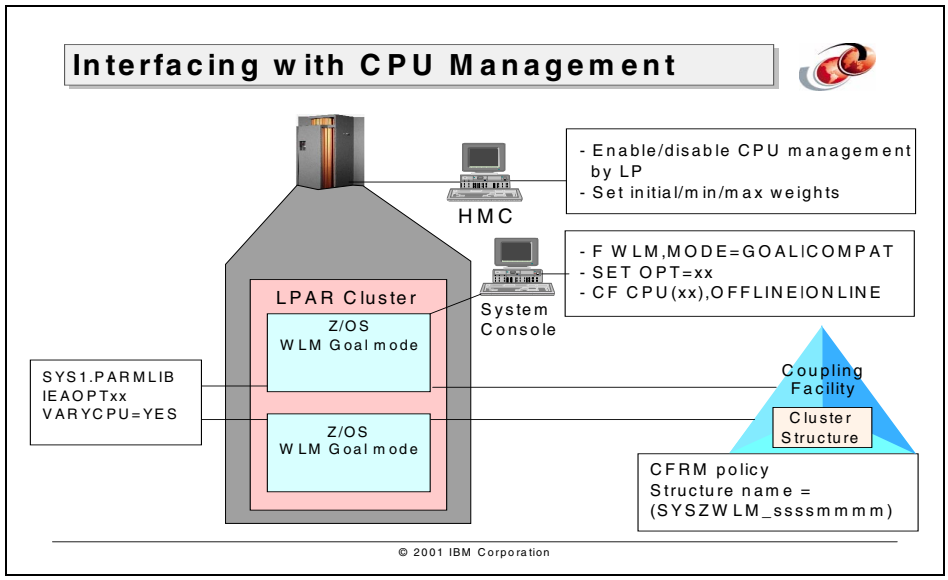
If WLM is placed into Compatibility mode, it maintains the connection to the structure. However, it stops writing the LDE to the CF. As part of the change in mode from Goal mode to Compatibility mode, WLM cleans up its entry in the structure.

The cluster structure has the following attributes:

- It supports System Managed Rebuild. System Managed Rebuild was introduced in OS/390 V2R8 and provides the ability to rebuild structures in advance of a planned configuration change. For WLM, you can also rebuild the structure after a connectivity failure, but this rebuild *must* be initiated manually - the system will not automatically rebuild the structure following the failure.
- It supports structure alter. The size of the structure can be changed using the SETXCF ALTER command. In addition, we recommend enabling Auto Alter support for the WLM cluster structure.
- The structure disposition is delete. This means that the structure will be deallocated after the last connector disconnects.
- The connection disposition is delete. This means that the structure stays in an undefined state in case of abnormal termination.

- The format of the structure name is shown in the foil, that is, SYSZWLM_ssssttt, where ssss is the last 4 characters of the CPU Serial Number, and tttt is the processor model - for example, 2064.

3.13 How to interface to WLM LPAR CPU Management



There are a number of external interfaces for controlling WLM LPAR CPU Management. They are discussed in detail in the implementation procedure covered in Chapter 5, “Implementing WLM LPAR CPU Management” on page 125. In this section, we just look at each interface and explain how it is used.

There are basically two interfaces for controlling WLM LPAR CPU Management:

1. At a hardware level, from the HMC
2. Within the software

The HMC contains the LPAR definitions for each LP in the Image Profiles. Refer to 5.5, “HMC definitions” on page 132, to see the relevant HMC panel. The LP characteristics that affect WLM LPAR CPU Management are:

- WLM Managed

  When this option is selected, the corresponding LP is enabled to be managed by WLM LPAR CPU Management. This turns on both WLM LPAR Weight Management and WLM LPAR Vary CPU Management at the hardware level only. There are still software actions that need to be completed.

- Initial processing weight

  This weight becomes the LP’s current weight after it is IPLed. When a system enters WLM Compatibility mode, its current weight is reset to this value.

- Minimum processing weight

  This is the minimum weight that WLM LPAR Weight Management can assign to this LP. We recommend that you let this value default (that is, don't specify a value for this field). This allows WLM to ensure that each LP has enough weight to ensure that critical system work gets sufficient CPU resource.

- Maximum processing weight

  This is the maximum weight that WLM LPAR Weight Management can assign to this LP.

  You can effectively turn off WLM LPAR Weight Management by specifying the same value for the Initial, Minimum, and Maximum weights.

- The number of CPs and whether they are shared or not

  This is the initial number of logical CPs that are brought online after an IPL. They must be shared for this LP to be included in the LPAR Cluster. If the LP is using dedicated CPs, the check box for WLM Managed cannot be selected. We recommend that you define the maximum number of logical CPs for each LP—this gives WLM the maximum flexibility to select the optimum number of logical CPs based on the LP's current weight and workload.

- The Initial Capping check box must not be selected. If this option is selected, the check box for WLM Managed cannot be selected.

In relation to the interfaces in the operating system, the following controls are available:

- You can have WLM LPAR Vary CPU Management inactive while leaving WLM LPAR Weight Management active.

  You do this by specifying VARYCPU=NO in IEAOPTxx.

- The z/OS system in the LP must be in WLM Goal mode in order to use either WLM LPAR Weight Management or WLM LPAR Vary CPU Management.

- The CFRM policy must contain a definition for a structure with the correct name. If the structure does not exist, or that LP does not have connectivity to it, WLM LPAR CPU Management cannot be used.

These interfaces are discussed in more detail in Chapter 6, "Operating WLM LPAR CPU Management" on page 139.

4

Planning for WLM LPAR CPU Management

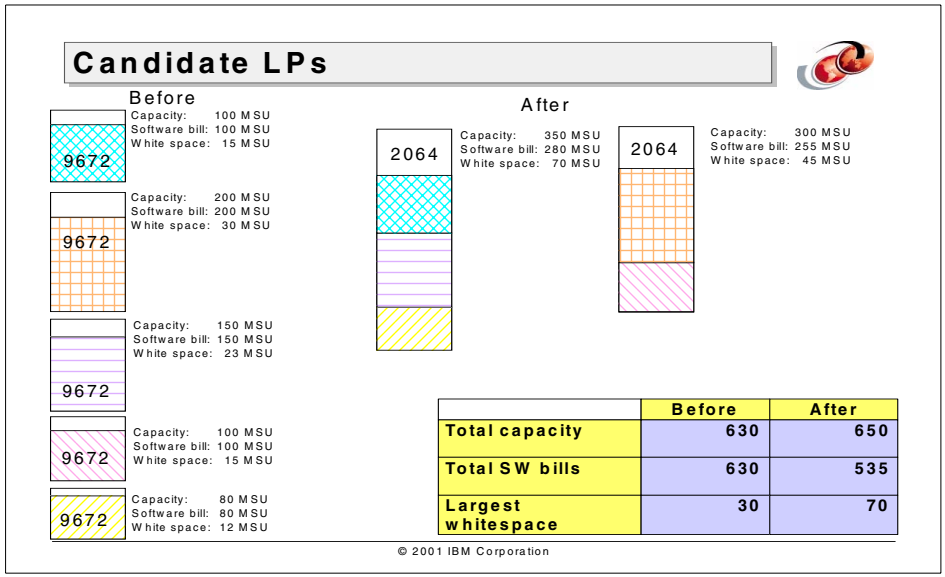
Actually implementing WLM LPAR CPU Management is not difficult, or even very time consuming. However, to ensure that the implementation goes smoothly and provides the desired results, it is important that you do careful planning in advance of the implementation.

This chapter discusses the planning required for implementing CPU management. We look at:

- Identifying candidate environments
- WLM policy definitions
- Hardware prerequisites
- Software prerequisites
- Mixed software releases considerations
- WLM mode considerations
- Coupling Facility requirements
- Recovery considerations
- Relationship to License Manager

Planning should not just include getting the correct software and hardware levels, but also building documentation for operations personnel. They may be required to turn WLM LPAR CPU Management off or on and therefore need an understanding of the external controls. They also need to know how to check the current status of the system as altered by WLM LPAR CPU Management. For more detailed information on operational considerations, see Chapter 5, “Implementing WLM LPAR CPU Management” on page 125.

4.1 Identifying candidate environments



In order to identify the candidate LPs for use with WLM LPAR CPU Management, you may need to look beyond your current configuration.

First of all, you should map out how your ideal environment would look, if you didn't have to be concerned about software licensing implications. For example, you would probably have at least two, and maybe three CPCs, for maximum availability. For those applications that support data sharing, you will need to run at least one LP containing that product on each CPC, to provide application availability across a hardware outage.

You would also want to provide a reasonable amount of "white space", or unused capacity, on each CPC, to allow for workload spikes in the LPs on that CPC.

The figure above shows a typical current environment, with a number of smaller CPCs, each of which has some products that are only licensed on that CPC. Prior to the ability to do subcapacity software licensing, it did not make sense to consolidate those systems onto a smaller number of more powerful CPCs because of the software cost implications.

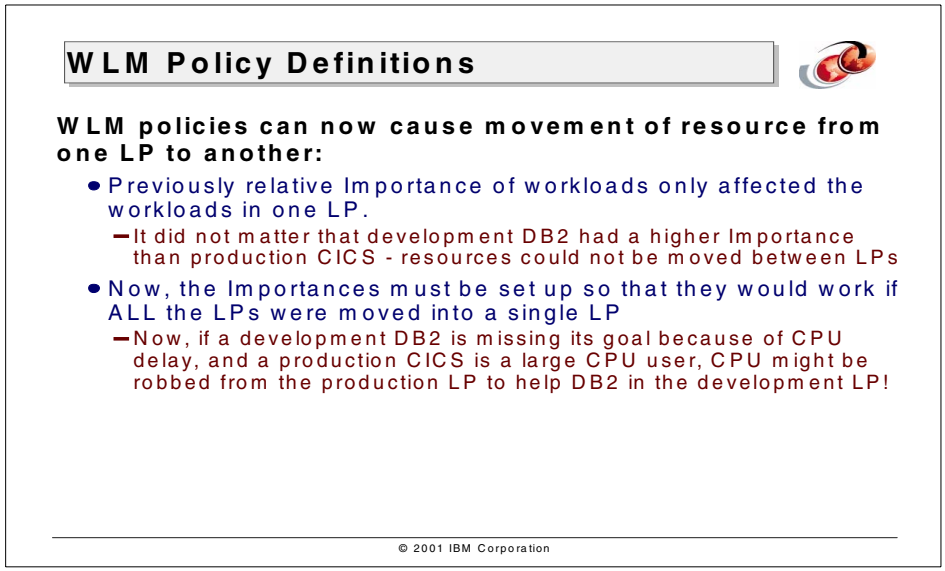
However, with the availability of subcapacity software licensing, it may be feasible to move these systems onto a larger CPC without increasing your software costs. You should take an inventory of the products used in each of your current systems and identify:

- Which ones avail of Variable Workload Charging. For these products, you will pay the same license costs regardless of the CPC they run on, assuming the size of the LP in which they run is the same. For example, whether you run one of these products on a 200 MSU CPC, or in an LP with a defined capacity of 200 MSUs on a 400 MSU CPC, you will pay the same software charges.
- Which products are charged on a Flat Workload Charging basis. For these products, you pay a license per CPC that they run on, regardless of the capacity of that CPC. So, for example, whether you run PLI V1 on a 2064-101 or a 2064-116, the software charge is the same. Obviously it makes sense to consolidate the LPs that use these products onto a small number of larger IBM zSeries 900 CPCs.
- Whether you have products that are charged on a Flat Workload Charging basis that have a newer version that is charged on a Variable Workload Charging basis—an old COBOL compiler, for instance. If you have such products, and they are only used by a subset of the systems on the CPC, it makes sense to upgrade them to the newer version and avail of Variable Workload Charging.
- Which LPs are in the same sysplex, and how many sysplexes you have. Obviously, to benefit from WLM LPAR Weight Management, you have to have more than one system on a CPC in the same LPAR Cluster. From a performance point of view, it also makes sense to have at least two LPs from each sysplex on a given CPC: if one LP is unavailable for some reason, having a second LP from the same sysplex on the same CPC ensures that the processing capacity of that CPC is not lost to the sysplex.
- Of the systems in each sysplex, are they all busy at the same time, or do their capacity requirements and workload mix change over the day? Systems that require capacity at different times of day, and those with shifting workload mixes are good candidates for residing in the same LPAR Cluster.

  However, if you are using the Workload Charging capability of the IBM zSeries 900 and z/OS, there are considerations relating to the use of defined capacities and prolonged shifting of weights from one LP to another. This is discussed further in 4.10, “IBM License Manager considerations” on page 121.

When you have completed this inventory of your systems, it should be clearer which ones are good candidates for residing within the same LPAR Cluster. If you currently have multiple systems from the same sysplex on the same CPC, you are off to a good start. However, even if your environment is not currently configured in this way, you should still consider this type of environment when planning your future configuration.

4.2 WLM Policy definitions



Prior to Intelligent Resource Director, the WLM decisions and actions on one system had a limited impact on other systems in the sysplex. However, once you start using Intelligent Resource Director, the WLM goals that you define take on a new, sysplex-wide effect. This is primarily a concern if you have development and production systems in the same LPAR Cluster.

For example, in the past it might have been acceptable to define your workloads with the following importances:

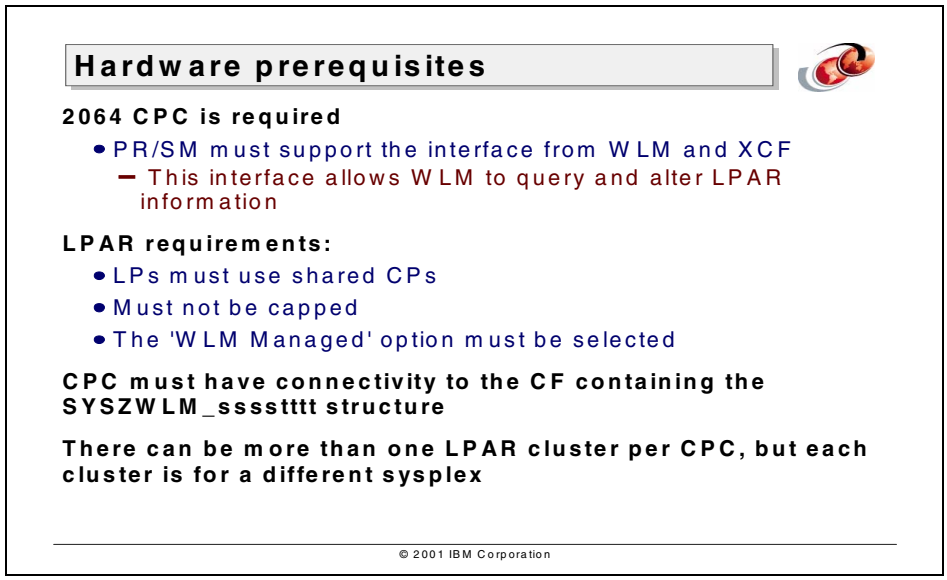
1. CICS
2. DB2
3. TSO
4. Short batch
5. Long batch

Because the scope of WLM control was effectively a single OS/390 image, and presuming that you only ran either production or development in the one image, the fact that the development DB2 was CPU-constrained had no impact on any of the production SCPs.

However, in an LPAR Cluster, WLM will now move CPU resource from one LP to another, based on the WLM importances of the work in those LPs. So you could now find that an Importance 3 production SCP is missing its goal because CPU was stolen to help an Importance 2 development SCP that was missing its goal. This is probably *not* what you want.

If you intend to place production and development systems in the same LPAR Cluster, you should review your WLM policies. The acid test is “What would happen if I were to merge all the systems in the LPAR Cluster into a single z/OS image?”. If the answer to this question is that the WLM Importance assigned to every workload makes sense in this merged environment, then you should have no problems. However, if the answer is that some development workloads will now have a higher WLM Importance than more critical production workloads, then you should modify your policy to change the relative Importances.

4.3 Hardware prerequisites



The IBM zSeries 900 server provides the new interfaces required for WLM LPAR CPU Management. These interfaces provide the ability for:

- WLM to query and update LP weights
- XCF to inform LPAR about the name of the system, and the name of the sysplex that this LP is a member of

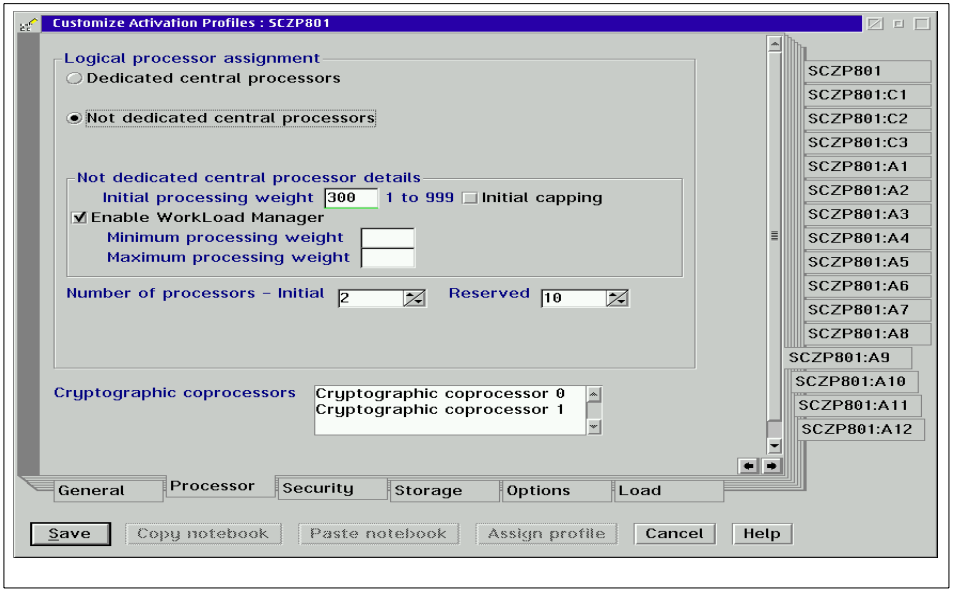
Each LP must be defined as follows:

- The `Enable Workload Manager` field in the HMC Image profile (as shown in the figure on the following page) must be selected.

  This option enables or disables both WLM LPAR Weight Management and WLM LPAR Vary CPU Management simultaneously at the hardware level. However, before WLM LPAR CPU Management can be used, there are software actions which need to be performed.

- The LP must use shared CPs.

  WLM LPAR CPU Management can only be used with LPs that use shared CPs. This is because one of the main objectives of WLM LPAR CPU Management is to move CPU resource from one LP to another, and dedicated CPs can only be used by a single LP.



We recommend defining all your physical CPs as shared. The only reason to use dedicated CPs is to minimize LPAR overhead, but the cost is that capacity that is not used by the associated LP cannot be used by other LPs, even if they are CPU-constrained. Given that WLM LPAR Vary CPU Management will work to minimize LPAR overhead, you should revisit any LPs that currently use dedicated CPs.

We also recommend defining your production LPs that will be using WLM LPAR CPU Management to have the Initial number of logical CPs equal to the number of shared physical CPs on the CPC. This gives WLM the most flexibility to make available the number of logical CPs that is most appropriate for the current capacity requirements.

- Do not use LPAR capping.

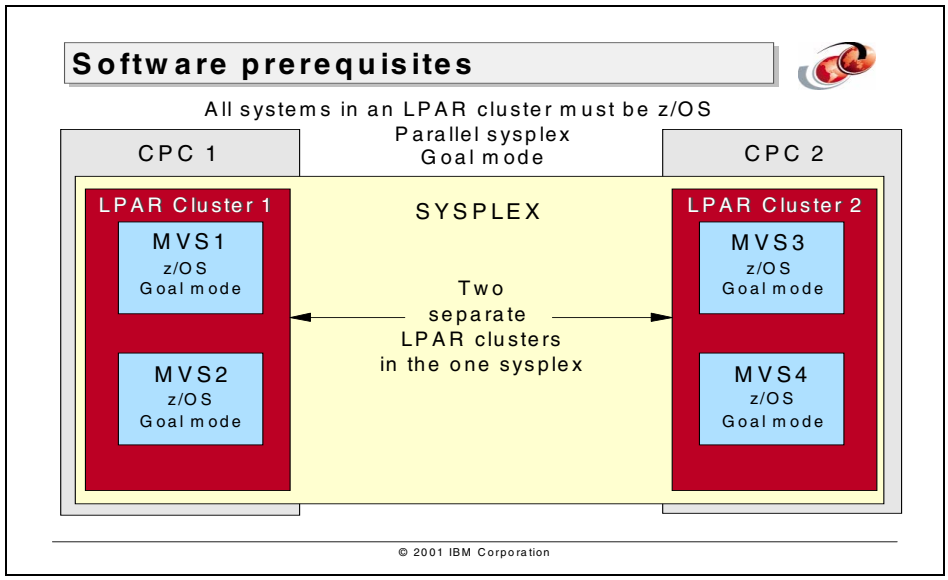
  The use of traditional LPAR capping is mutually exclusive with WLM management. As we discuss in 3.6, “LPAR capping” on page 64, there are alternatives to LPAR capping that should be considered in preference to the use of capping.

- Ensure the LP has connectivity to the CF containing the LPAR Cluster structure (SYSZWLM_ssssttt). The LPAR Cluster structure is discussed further in 4.7, “Coupling Facility prerequisites” on page 115.

Each LPAR should have *at least* a 2-digit weight, and a 3-digit weight is recommended if possible. Although this is not mandatory, a single digit weight is not meaningful to WLM. When calculating how much weight to move from one LP to another, any value less than 1 is not a complete weight unit, and therefore the weight is not adjusted.

There can be more than one LPAR Cluster *per sysplex* if the sysplex extends across multiple CPCs. There can also be more than one LPAR Cluster *per CPC* - but in this case each LPAR Cluster will consist of systems that are in different sysplexes. This is explained further in 4.8, “Multiple LPAR Cluster/sysplex configurations” on page 117.

4.4 Software prerequisites



The software requirements for WLM LPAR CPU Management are:

- The LP must be running z/OS 1.1 or later in z/Architecture mode.
- The system must be in WLM Goal mode.
- The system must be in a Parallel Sysplex.
- To get the new WLM LPAR CPU Management RMF reports, you must have RMF APAR OW46477 and OW48190.
- *All* the PTFs identified in the IRD PSP bucket should be applied. The UPGRADE is 2064DEVICE, and the SUBSET is IRD. Also, refer to the WLM Web site for the latest information at the following URL:

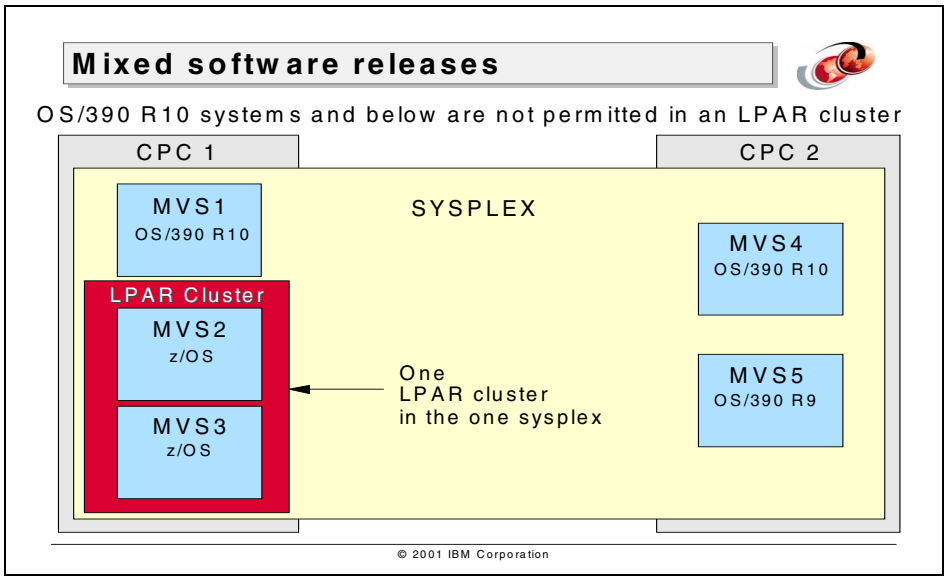
  `http://www.ibm.com/servers/eserver/zseries/zos/wlm/documents/ird/ird.html`

The example in the figure above shows two separate LPAR Clusters in one sysplex. As the scope of an LPAR Cluster is a single CPC, the above configuration where the sysplex includes two IBM zSeries 900 or later CPCs will therefore contain two LPAR Clusters.

This also shows that all the LPs on a CPC that are running z/OS and are in the same sysplex will automatically be in the same LPAR Cluster. From a WLM LPAR CPU Management point of view, there is nowhere that you actually define which systems are in the LPAR Cluster - the scope of the LPAR Cluster is automatically

defined based on the sysplex that each LP is a member of. WLM LPAR Weight Management can redistribute LPAR weights between the systems within a single LPAR Cluster—it cannot redistribute weights among systems that are not in the same sysplex (and therefore not in the same LPAR Cluster).

4.5 Mixed software releases

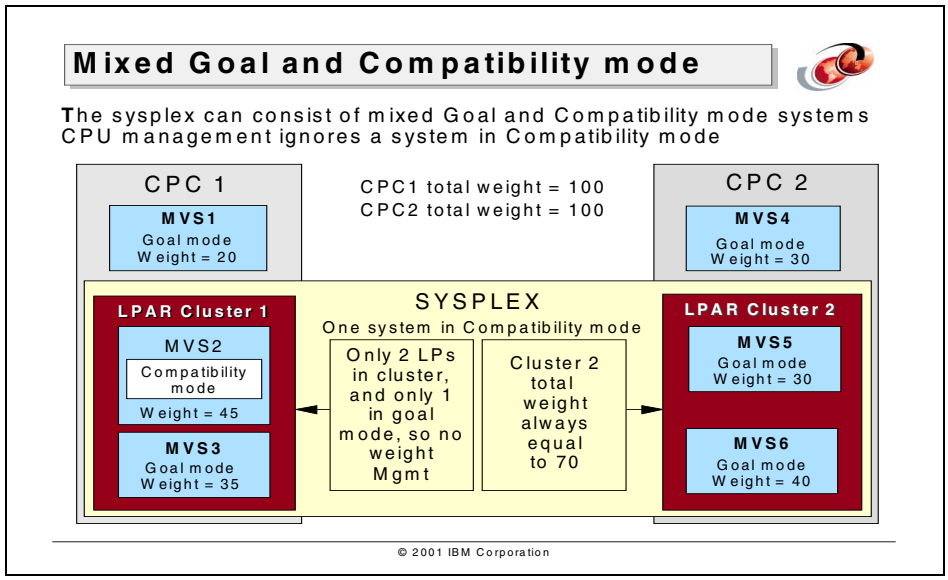


Only systems that are running z/OS 1.1 or later (and running in z/Architecture mode) can take part in an LPAR Cluster. Obviously, earlier level systems can be in the same sysplex, but they cannot be a member of the LPAR Cluster. This is in keeping with the sysplex coexistence policy of n, n + 3 releases. There is a special one-time extension to this policy for z/OS 1.1, whereby it can coexist with systems as far back as OS/390 2.6.

In the example shown above, there are two CPCs, each with LPs in a single sysplex. The systems on CPC2 are all OS/390 2.10 or earlier, and therefore do not support Intelligent Resource Director. The systems on CPC1 consist of one OS/390 2.10 system and two z/OS systems (MVS2 and MVS3). MVS2 and MVS3 are part of an LPAR Cluster; however, the OS/390 2.10 system which is in the same sysplex and on the same CPC cannot take part in the LPAR Cluster.

When WLM on MVS2 or MVS3 tries to help a service class period, it can alter the weights of either of these LPs to provide more CP resource. If WLM on MVS1, MVS4 or MVS5 tries to help a service class period, that system can only alter the dispatching priorities of work running within that system.

4.6 WLM mode considerations



A system must be in WLM Goal mode to use WLM LPAR CPU Management.

If a system in WLM Goal mode that is part of an LPAR Cluster is switched to Compatibility (COMPAT) mode, that LP's weight is reset to the Initial weight as defined on the HMC for that LP. However, the total weight of the LPAR Cluster cannot increase or decrease when weights are changed. Therefore, this switch to WLM Compatibility mode causes one of three results:

1. The COMPAT system's weight increases.

   This causes the weights of the other systems in WLM Goal mode in the LPAR Cluster to decrease. The decrease is proportionally distributed between the remaining Goal mode systems. That is, each system loses the same percentage of their weight.

2. The COMPAT systems weight decreases.

   This causes the weights of the other systems in WLM Goal mode in the LPAR Cluster to increase. The increase is proportionally distributed between the Goal mode systems. That is, each system gains the same percentage of their weight.

3. The COMPAT systems weight remains unchanged.

   The weights of the Goal mode systems in the LPAR Cluster are unaffected by this change.

It is important to understand this during the planning phase. Changing one system from WLM Goal model to Compatibility mode has the potential of impacting the amount of CP resource that is available to other systems in the LPAR Cluster.

An LPAR Cluster may contain systems that are in WLM Compatibility mode. However, WLM LPAR CPU Management ignores systems in Compatibility mode when performing management actions. They are still considered part of the LPAR Cluster even though they are not managed. One reason a system in Compatibility mode is still considered part of the LPAR Cluster is that, even though its weight is not be changed by WLM LPAR Weight Management, its weight is included as part of the total LPAR Cluster weight.

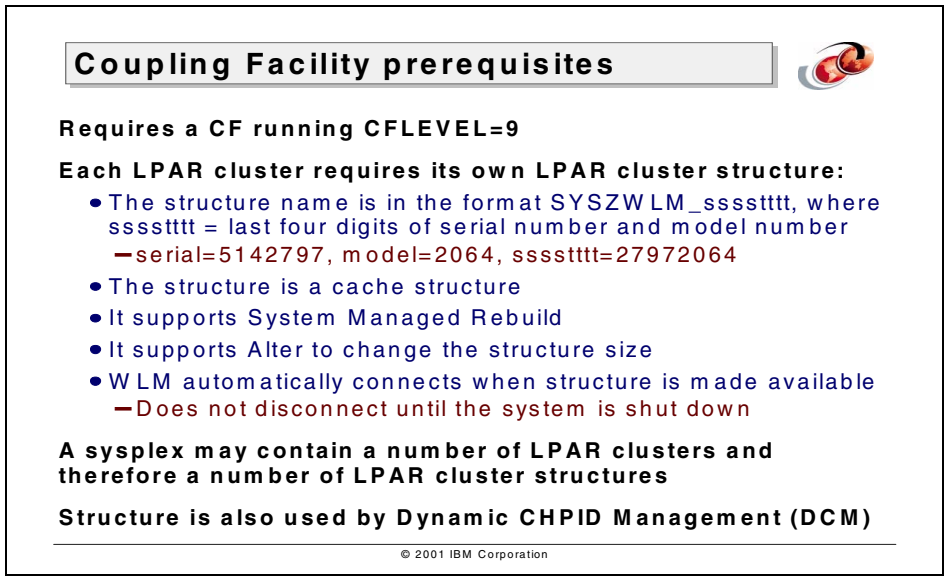
In the example shown on the previous page, we have two LPAR Clusters. LPAR Cluster 1 on CPC1 has two systems, MVS2 and MVS3. MVS2 is in WLM Compatibility mode, while MVS3 is in Goal mode. Because there are only two systems in this LPAR Cluster and one is in Compatibility mode, neither system's weight is changed by WLM LPAR Weight Management. This is because MVS2 can neither receive or donate any weight as long as it is in Compatibility mode.

You may ask what is the point of having a two-system LPAR Cluster where one of the LPs is in Compatibility mode? If you have a z/OS system on the same IBM zSeries 900 as another z/OS that is in the same sysplex, and you have defined the WLM LPAR Cluster structure, then the two systems will automatically be in the same LPAR Cluster—you have no control over this.

However, given that z/OS 1.2 will be the last release to support Compatibility mode, we expect that the normal mode of operation for all members of an LPAR Cluster will be Goal mode, and any system in a cluster that is in Compatibility mode will be an exception.

You should also understand that when a system is in WLM Compatibility mode, WLM LPAR Vary CPU Management does not take place. Once the system is switched to Compatibility mode, any CPs that were varied offline by WLM will be varied back online, and will remain online until either the system goes back into Goal mode, or an operator varies some of the CPs offline manually.

4.7 Coupling Facility prerequisites



WLM LPAR CPU Management requires a Coupling Facility (CF) running Coupling Facility Control Code (CFCC) Level 9 or higher. CFLEVEL 9 is only supported on IBM 9672-Rn6, Yn6, Xn7, and Zn7 (that is, 9672 G5 and G6), or any model of IBM zSeries 900.

Each LPAR Cluster requires its own LPAR Cluster structure, with a name in the format of SYSZWLM_ssssttt. The ssssttt is made up of the last four digits in the machine serial and the type of the CPC. For example:

Machine serial = 5142797

Model number = 2064

Therefore ssssttt = 27972064

The structure has the following characteristics:

- It is a cache structure.
- It supports System Managed Rebuild for rebuild from a planned configuration change.
- It supports Alter to change the structure size.
- It is recommended to use Auto Alter with this structure, so that if it starts to fill up, the system will automatically give it additional space.

- The recommended starting INITSIZE is 6144 KB - this should be sufficient for both the Dynamic Channel-path Management and the WLM LPAR CPU Management structures. The standard recommendation is that SIZE should be twice the value specified for INITSIZE, so in this case, the SIZE would be 12288 KB.
- Because the structure uses System Managed Rebuild, it does not automatically rebuild in case of an unplanned configuration change (for example, a broken CF link). The system that has lost connectivity will no longer be managed until it regains connectivity to the CF.

  If you wish, you can manually initiate a rebuild to an alternate CF; as soon as the rebuild completes, WLM on the system that lost connectivity will automatically reconnect to the new structure, and that system will once again be managed.

  If you wish to automate this process, the message that is issued when WLM loses connectivity to the structure is:

  ```
  IWM050I STRUCTURE(SYSZWLM_ssssttt),DISCONNECTED
  ```

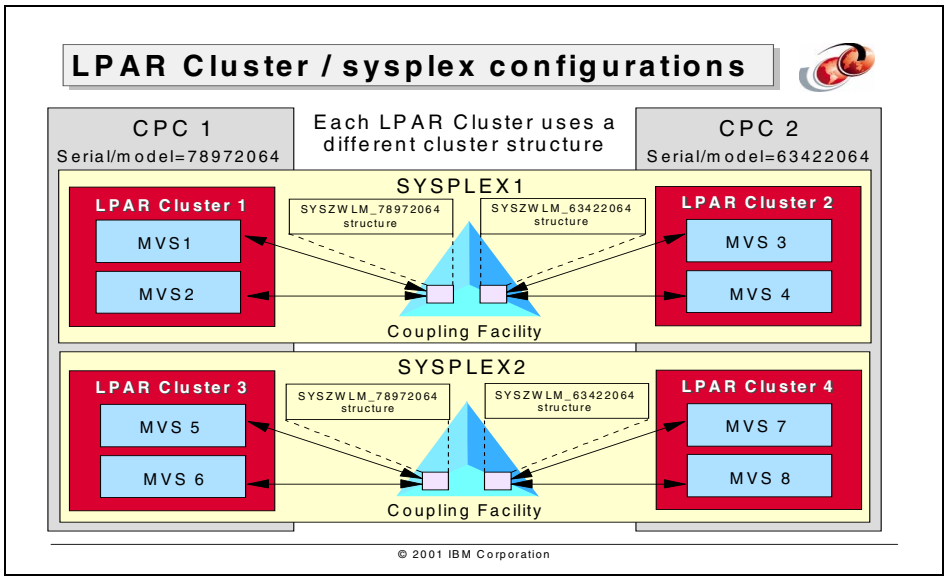
When the structure is made available in the CF, or a new CFRM policy is started that contains a definition for that structure, WLM is notified by XES. WLM then connects to the structure automatically and does not disconnect until an IPL.

If a system is placed in WLM Compatibility mode, WLM on that system remains connected to the cluster structure. However, it is no longer managed and it does not update the structure. When the system is placed back into WLM Goal mode, it once again starts to update the structure.

There may be a number of LPAR Clusters in a sysplex, although there is only one LPAR Cluster per sysplex per CPC. Therefore, there can be more than one cluster structure per sysplex.

This structure is also used by Dynamic Channel-path Management (DCM), another of the functions that forms Intelligent Resource Director. For more information on DCM see Chapter 9, “How Dynamic Channel-path Management works” on page 189.

4.8 Multiple LPAR Cluster/sysplex configurations



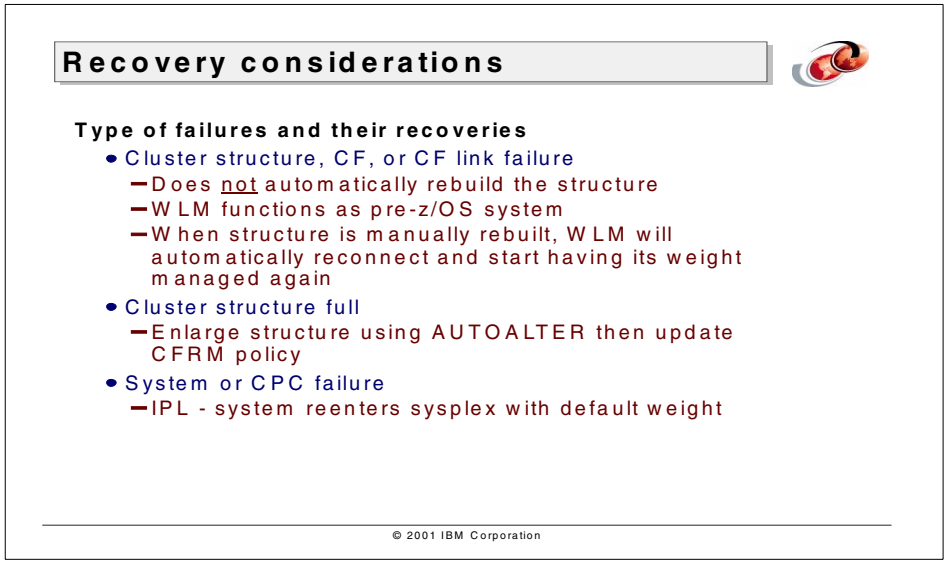
In the example above, we have two sysplexes running on two CPCs, resulting in four LPAR Clusters:

1. LPAR Cluster 1
   - Has two z/OS systems, MVS1 and MVS2
   - Is part of SYSPLEX1
   - Is on CPC1
   - Is connected to cluster structure SYSZWLM_78972064
2. LPAR Cluster 2
   - Has two z/OS systems, MVS3 and MVS4
   - Is part of SYSPLEX1
   - Is on CPC2
   - Is connected to cluster structure SYSZWLM_63422064
3. LPAR Cluster 3
   - Has two z/OS systems, MVS5 and MVS6
   - Is part of SYSPLEX2
   - Is on CPC1
   - Is connected to cluster structure SYSZWLM_78972064
4. LPAR Cluster 4
   - Has two z/OS systems, MVS7 and MVS8

- Is part of SYSPLEX2
- Is on CPC2
- Is connected to cluster structure SYSZWLM_63422064

LPAR Clusters 1 and 3 are both using the same cluster structure name; however, because the structures are in two different sysplexes, this does not cause a problem. Neither sysplex has access to the other's CF and therefore no access to each other's structures. The same is true for clusters 2 and 4. As each cluster on the same CPC is part of a different sysplex, they can use the same cluster structure name.

4.9 Recovery considerations



WLM performs most of its own recovery without operator intervention. In these cases, you simply need to understand what is happening.

However, when operator intervention *is* required, you must consider the following in relation to the LPAR Cluster and the sysplex:

- Cluster structure connection failure

  If one system loses connectivity to the CF containing the WLM structure, that system will stop taking part in WLM LPAR CPU Management until connectivity to the structure is reestablished. Because this is an unplanned configuration change, System Managed Rebuild will not automatically initiate a rebuild of the structure to an alternate CF.

  However, a rebuild can be done manually, using the SETXCF START,REBUILD command. As soon as the system regains connectivity to the structure, WLM LPAR CPU Management will recommence for that LP.

- CF failure

  If the CF containing the WLM structure fails, WLM will automatically allocate a new, empty, structure in the alternate CF, repopulate it with information about each system, and recommence using WLM LPAR CPU Management.

- Cluster structure full

  WLM issues a message (IWM053I) on the z/OS operator console. We recommend that the structure is defined with an INITSIZE of 6144 KB and a SIZE of 12288 KB—this gives you room for significant growth without having to update the CFRM policy. You can then issue the SETXCF START,ALTER command to increase the structure size.

  Further, we recommend using the AUTO ALTER function added by OS/390 V2R10. This is enabled for the structure by specifying ALLOWAUTOALT(YES) in the structure definition in the CFRM policy. The use of this feature allows z/OS to automatically increase the structure size if it exceeds a user-specified usage threshold (FULLTHRESHOLD, also on the structure definition).

- System failure or CPC failure

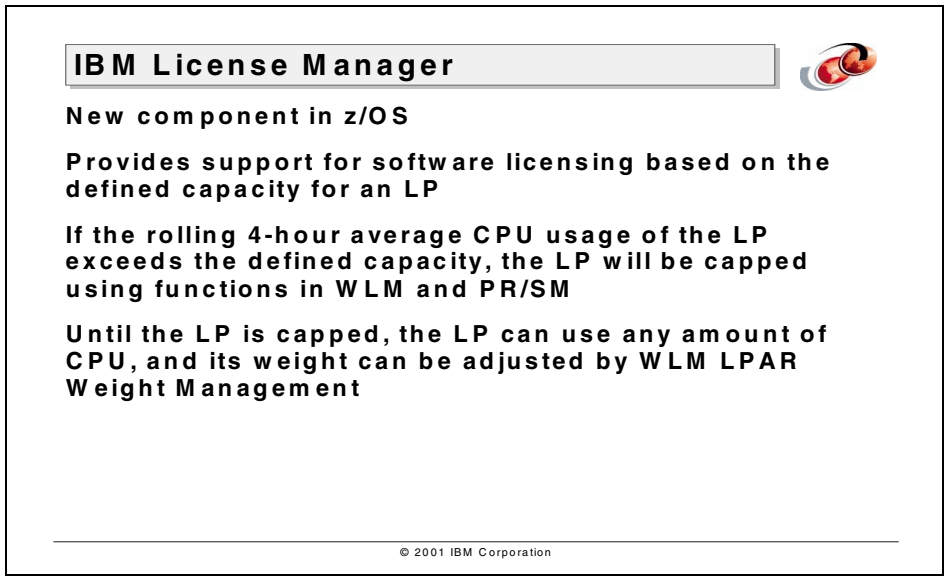
  In the case of a system failure, any remaining systems in the LPAR Cluster continue to operate and swap weights. The weight of the system that has failed will remain unchanged until either a System Reset is done on the LP (this would normally happen as part of the process of removing a system from the sysplex), or the LP is deactivated, or the operating system is re-IPLed.

  If the LP is deactivated, its weight is redistributed among *all* the active LPs on the CPC and the total weight of the LPAR Cluster is recalculated.

  If a System Reset is done for the LP, or it is IPLed, its weight is reset to its Initial weight and all logical CPs are brought online. This could impact other systems, especially if this system needs to take weight from the other systems to reach its initial weight. See 4.6, “WLM mode considerations” on page 113 for more detail. Once back in the LPAR Cluster, the system or systems will connect to the structure. The main point to note here is that whenever a system is IPLed, its weight is reset to its initial weight and all its logical CPs are brought online.

  In the case of a complete CPC failure, all the systems in the LPAR Cluster are dead. When the CPC is recovered, all LPs will start off with their Initial weight and all their logical CPs online.

4.10 IBM License Manager considerations



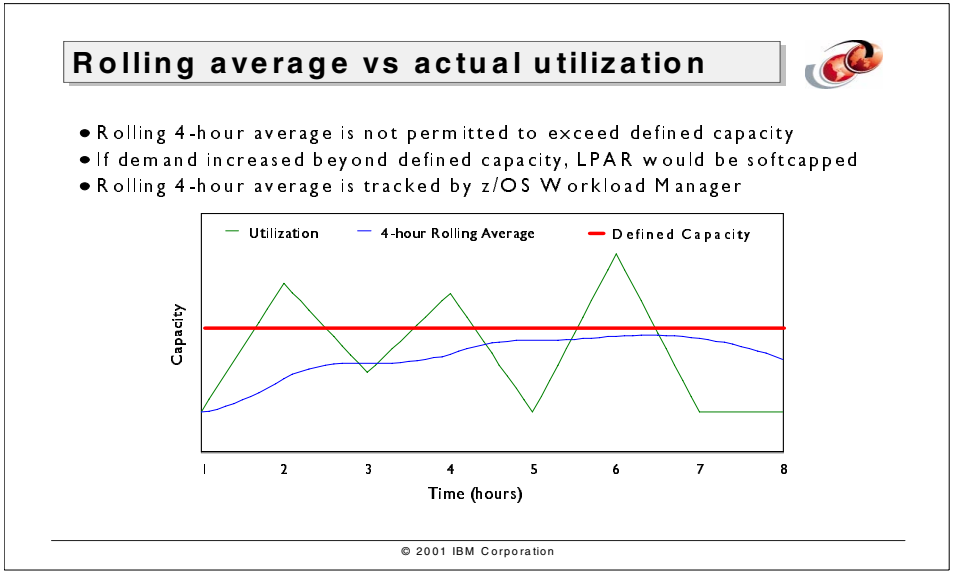
IBM License Manager is a new standard component of z/OS that will be used to control the licences for the products you run on your z/OS system. We recommend reviewing the available License Manager documentation before you read this section. You should also be familiar with the general concept of Workload Charging and how this relates to License Manager.

In this section, we discuss the License Manager considerations as they relate to implementing WLM LPAR CPU Management.

When you define an LP, in the HMC Image profile you can specify a “defined capacity” for the LP. This is used as an upper bound beyond which the rolling 4-hour average CPU consumption of a z/OS LP cannot proceed.

The important thing to note here is that it is the 4-hour average that determines whether the LP will be capped or not. This is different from traditional LPAR capping, where the LP cannot ever exceed the defined cap (which is determined by the weight specified for the LP).

An LP which has a defined capacity specified can (and probably will) exceed that capacity at some point. For example, if the LP has a defined capacity of 20 MSUs, but has been using only 5 MSUs for the last few hours, the LP



could expand to use the whole capacity of the CPC (which could be hundreds of MSUs) for a short time, assuming that no other LP requires that capacity.

In terms of the relationship between the defined capacity for an LP and the weight of the LP—the two values are independent. The defined capacity is specified by you, based on the amount of capacity that you have specified in the certificates for the products running in that LP. The figure above shows the relationship between the actual utilization and the rolling 4-hour average. The actual utilization is impacted by the weights defined for the LP. If the LPs on the CPC are all running at 100% of their guaranteed capacity, the actual consumption of the LP will be dictated by the current weight of the LP.

If you expect the LP to intermittently require capacity above that indicated by the defined capacity, you should specify the weights accordingly. Remember that WLM LPAR Weight Management will adjust the weight of the LP if the workload running in the LP is of sufficient importance. However, if the LP runs above its defined capacity for a prolonged period, the rolling 4-hour average will eventually reach the defined capacity, at which point the LP will be capped to the defined capacity (a process known as *soft capping*).

As WLM is responsible for both WLM LPAR Weight Management and soft-capping, it will be aware when an LP is being soft-capped, and will not make any further adjustments to the weight of the LP until the soft-cap is removed.

While it might be nice to have WLM adjust the defined capacity as it adjusts the weights for the LP, you have to remember that each LP could potentially be running different products. For example, assume LP1 is running CICS, and LP2 is running IMS. You would probably set your defined capacities to match the certificates for IMS and CICS. If WLM were to increase the defined capacity of LP2, you would then generate an exception if you tried to start IMS while the defined capacity of LP2 is greater than the certificate size for IMS.

Therefore, at the time of writing, there is no automatic way to adjust the defined capacity for an LP in line with changes in the LP weights. If you have this type of configuration, where the weights move significantly from one LP to another for whole shifts, then you must license the products in each LP at the highest rolling 4-hour average capacity that will be required by that LP.

For more information on Workload Charging and IBM License Manager, refer to the Web site:

http://www.ibm.com/servers/eserver/zseries/wlc_lm/

5

Implementing WLM LPAR CPU Management

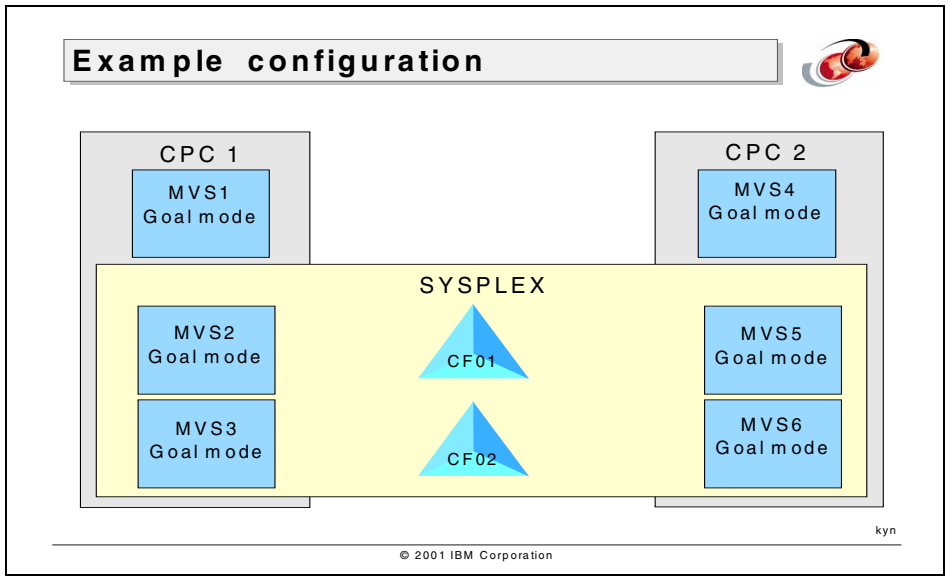
This chapter outlines, step by step, what needs to be done to both the hardware and software to implement WLM LPAR CPU Management. It assumes that you have already completed the planning tasks identified in Chapter 4, “Planning for WLM LPAR CPU Management” on page 101.

Specifically, the task of deciding which operating system images should reside in each LPAR Cluster is *not* covered—we expect that you already have this information before you start activating WLM LPAR CPU Management. We also do not discuss the task of migrating a system to WLM Goal mode in detail—this is a project in its own right, and should be completed before you start implementing WLM LPAR CPU Management.

In general, it does not matter in which sequence the steps are performed. However, if you are adding one LP at a time to the LPAR Cluster, then defining the WLM cluster structure should be the first step, and if you are placing all systems into the LPAR Cluster simultaneously, then the final step should be the definition and activation of the WLM cluster structure.

In this chapter, we follow the phased approach of building a 2-LP LPAR Cluster in a 4-system Parallel Sysplex. This, of course, implies that we are running two separate CPCs. We first implement one LPAR Cluster on one CPC, and then repeat the process on the other CPC.

5.1 Example configuration



The example foil above shows the starting configuration. It consists of a sysplex containing four systems, all of which are in WLM Goal mode:

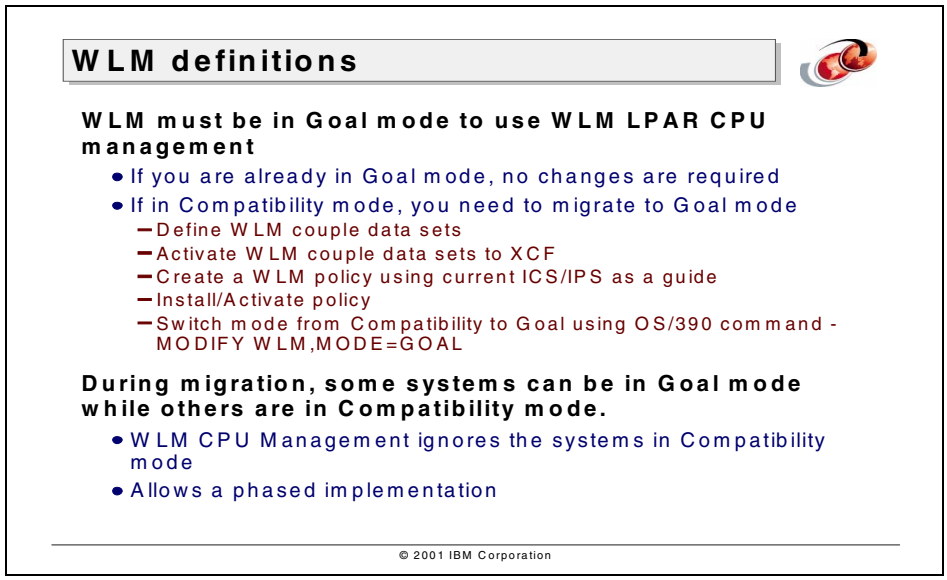
- MVS2
- MVS3
- MVS5
- MVS6

MVS2 and MVS3 are on CPC1, which is an IBM zSeries 900 CPC with 10 CPs. Its serial number is 000000042797 and its model number is 2064. It also has another LP, MVS1, which is not part of the sysplex and therefore is not taking part in the LPAR Cluster. This CPC will contain LPAR Cluster 1.

MVS5 and MVS6 are on CPC2, which is also an IBM zSeries 900 CPC with 10 CPs. Its serial number is 000000046342 and its model number is 2064. It also has another LP, MVS3, which is not part of the sysplex and therefore is not taking part in the LPAR Cluster. This CPC will contain LPAR Cluster 2.

There are two Coupling Facilities (CFs), CF01 and CF02.

5.2 WLM definitions



As mentioned in the introduction of this chapter, the actual implementation of WLM is not covered in any detail here. This section is included for completeness and gives you a brief overview of the steps involved.

If you are already in WLM Goal mode, there are no further definitions required in WLM for WLM LPAR CPU Management. You should, however, review the considerations discussed in 4.2, “WLM Policy definitions” on page 105 if you are placing production and development systems in the same LPAR Cluster.

If you are *not* in WLM Goal mode, the following steps need to be completed:

1. Define a WLM couple data set.

   This couple data set contains your WLM policy. It is another couple data set that is managed and owned by XCF, and can be allocated using the XCF format utility IXCL1DSU.

   Details of how to use this utility and an example of allocating a WLM couple data set are found in *OS/390 MVS Setting up a Sysplex*, GC28-1779. The WLM couple data set can also be allocated using the WLM ISPF dialog.

2. Define the WLM couple data sets to XCF.

   This makes the couple data sets accessible to XCF. It is done using operator commands, as follows:

   ```
   SETXCF COUPLE,PCOUPLE=WLM primary data set name,TYPE=WLM
   ```

```
SETXCF COUPLE,ACOUPLE=WLM alternate data set name,TYPE=WLM
```

3. Create a WLM policy.

   The WLM policy is how you define your workload goals and classification criteria to WLM. Some installations set up their WLM policies based on their existing IEAIPS and IEAICS members. Others use one of the sample starter policies that can be obtained on the WLM Web site at:

   ```
   http://www.ibm.com/servers/eserver/zseries/software/wlm/
   ```

   Customers that have a more complex environment may wish to use the WLM Migration Aid tool, also available on the WLM Web site. This tool takes a bit of work to set up, but the resulting WLM policy is closer to a production policy than either of the sample policies will be.

   Once you have created this policy, it is installed into your WLM couple data sets using the WLM ISPF dialog.

4. Install/activate the WLM policy.

   Installing the policy means to copy it into the WLM couple data sets. Activating the policy informs WLM which is the active policy (even in Compatibility mode, there would be an active WLM policy). Both of these actions can be performed from within the WLM ISPF dialog.

5. Switch from WLM Compatibility mode to WLM Goal mode.

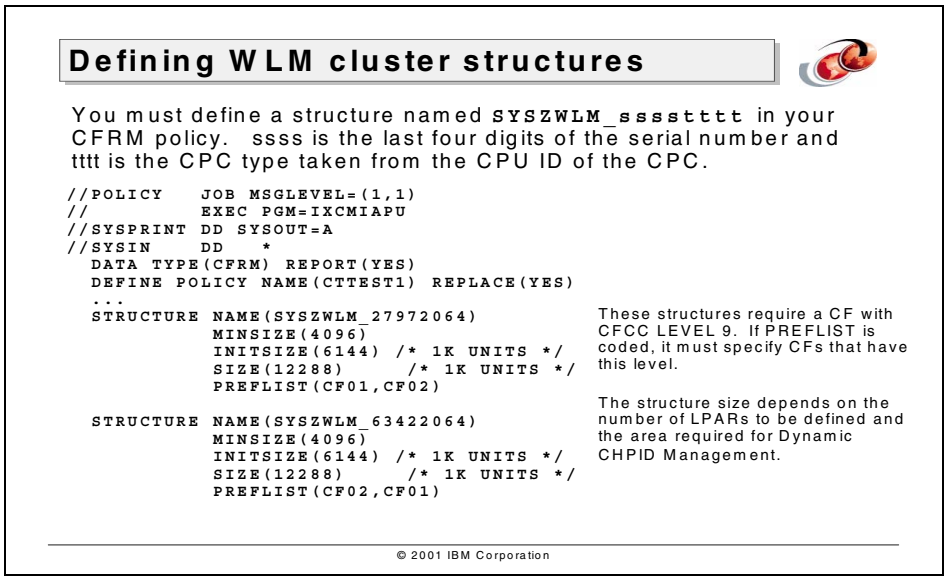
   When you switch WLM into Goal mode, WLM starts using the policy that you have just activated to manage the system according to the work classifications and goals in the policy. It no longer uses the ICS/IPS definitions. To switch to Goal mode, issue the following operator command:

   ```
   MODIFY WLM,MODE=GOAL
   ```

For information on implementing WLM Goal mode, see:

- *OS/390 MVS Planning: Workload Management*, GC28-1761
- *OS/390 Workload Manager Implementation and Exploitation*, SG24-5326

5.3 Defining WLM structures



Each LPAR Cluster requires a CF cluster structure for WLM LPAR CPU Management. This cluster structure is also used by Dynamic Channel-path Management. The DCM use of the structure is discussed in Chapter 9, “How Dynamic Channel-path Management works” on page 189 and is not covered here. Because we have two LPAR Clusters, one for each CPC, we define two CF structures in the CFRM policy.

Because WLM uses System Managed Rebuild to rebuild the structure in the event of planned configuration changes, the CFRM policy must contain support for System Managed Rebuild by specifying ITEM (NAME(SMREBLD) NUM(1)).

Also, due to the use of functions that only exist in CFLEVEL=9 or higher, the WLM structure must exist in a CF running at least that level of CFCC. In order to be able to rebuild, we recommend that you upgrade *all* your CFs to this level before implementing Intelligent Resource Director.

The CFRM policy is created and updated using the XCF administrative utility IXCMIAPU which is documented in *OS/390 MVS Setting up a Sysplex*, GC28-1779. The name of the structure must be SYSZWLM_ssssttttt, where ssss is the last 4 digits of the CPC and the tttt is the type of the CPC. The following are details about the CFRM policy that defines our cluster structures:

- Our CFRM policy name is CTTEST1.

- The structure name for cluster structure one which is on CPC1 is SYSZWLM_27972064, and the structure name for cluster structure two which is on CPC2 is SYSZWLM_63422064.
- The minimum size and initial sizes of each structure are 4096 KB and 6144 KB respectively. The maximum size for both structures is 12288 KB. The structure will initially be allocated with a size of 6144 KB. If we need a larger structure, we can use the SETXCF ALTER operator command to increase the size of the structure up to the value specified on the SIZE parameter which in this case is 12288 KB.
- In our structure definition, we have placed each cluster structure in a different CF. This is seen in the PREFLIST statement. The preferred CF for Cluster structure SYSZWLM_27972064 is CF01, but it can also reside in CF02, while the preferred CF for Cluster structure SYSZWLM_63422064 is CF02, but it can also reside in CF01.

  From an availability point of view, it really doesn't make any difference whether the structure is in the same CPC as the operating systems in the LPAR Cluster. However, from a performance point of view, it is probably better to balance the load of the two structures across the two CFs rather than placing both of them in the same CF. Either structure can be moved to the other CF using the SETXCF REBUILD operator command.

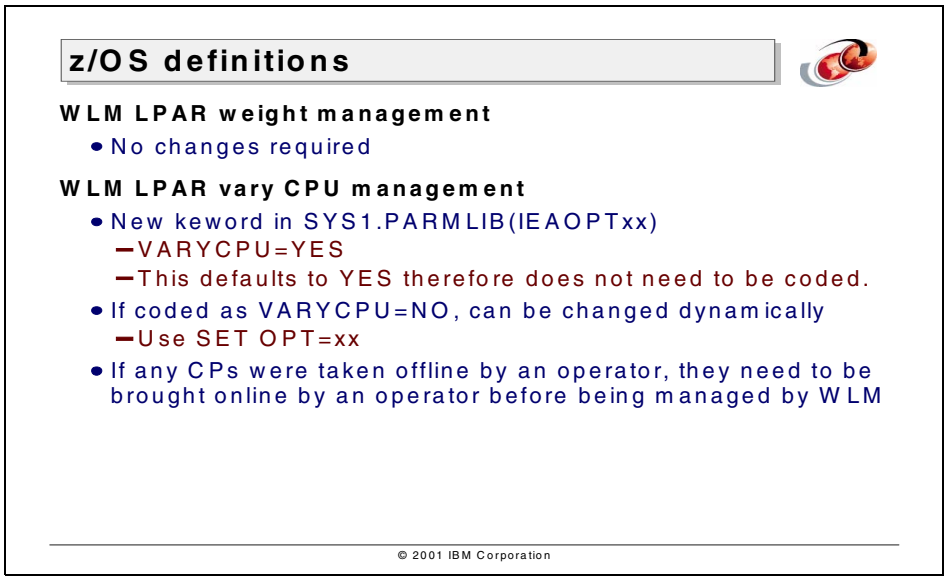
Once you have defined the new CFRM policy, it can be activated using the following operator command:

```
SETXCF START,POL,POLNAME=CTTEST1,TYPE=CFRM
```

This makes the structure available to WLM. When a new policy is activated, an ENF 35 signal is sent on all systems in the sysplex. WLM in z/OS listens for this ENF. WLM now knows there is a cluster structure available and each system connects to the appropriate cluster structure. The CF structure is actually allocated when the first WLM instance connects to it. WLM is ready to use the cluster structure for WLM LPAR CPU Management once all the remaining software and hardware actions are completed.

Don't forget to give the RACF userid associated with the WLM address space access to the LPAR Cluster structure. This is protected by the IXLSTR.structure_name profile in the SAF class CLASS(FACILITY). If you have not defined a userid for WLM, and you have a RACF profile that covers the LPAR Cluster structure, the RACF violations (ICH408I messages) will not tell you which address space is failing to obtain access.

5.4 z/OS definitions



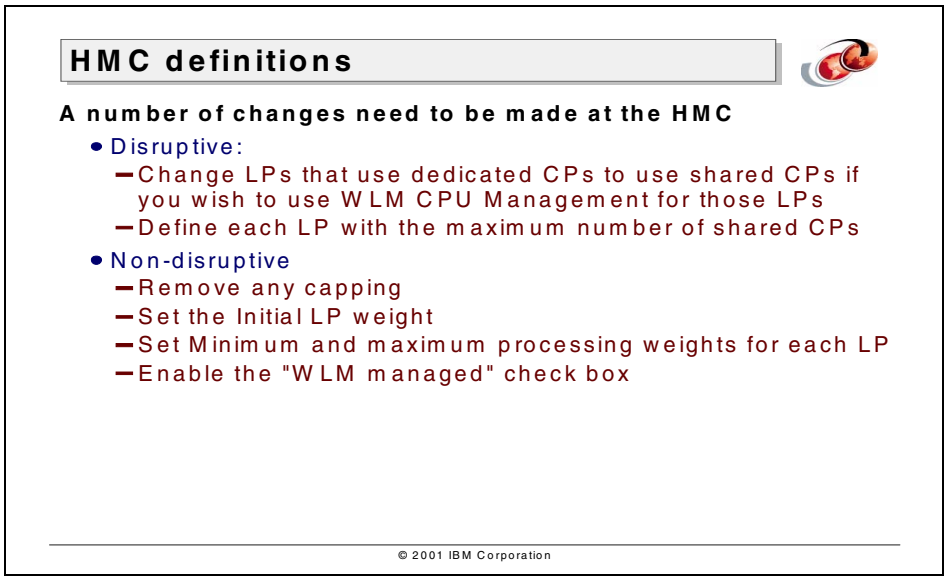
There is no need to make any changes to any of the z/OS definitions in order to use WLM LPAR CPU Management. Once the hardware is enabled, the structure is available and WLM is in Goal mode, it will automatically be active.

Having said that, if you wish to *stop* WLM LPAR Vary CPU Management, it is possible to do so by updating the IEAOPTxx member of SYS1.PARMLIB:

- You can add the keyword VARYCPU=NO to member IEAOPTxx, then issue the SET OPT=xx operator command to activate the change dynamically. The scope of this command is a single system. If there are multiple systems in the LPAR Cluster, the other systems will continue to use WLM LPAR Vary CPU Management.
- If any logical CPs were taken offline by an operator, they will need to be configured back online using the CF CP(xx),ONLINE system command before they can be managed by WLM LPAR Vary CPU Management. Once they are brought back online, WLM will vary them offline and online as it sees fit.

z/OS is now enabled for WLM LPAR CPU Management. This only leaves activation at the hardware level.

5.5 HMC definitions



The last changes are hardware changes. The reason these have been left until last is so we can use a phased approach to the implementation. We can enable WLM LPAR CPU Management one LP one at a time.

When changing the definition for an LP on the HMC, you have to remember to change both the definition for the LP in the Image Profile (which controls how the LP comes up after the next deactivation), as well the changing the current configuration of the LP in the Change LPAR Controls panel. This section covers the changes to the Image Profile—changes to the current LP configuration are covered in 5.5.1, “HMC Change LPAR Controls panel” on page 135.

If you wish to change an LP from using dedicated CPs to shared CPs, or to change the Initial number of logical CPs for the LP, those changes are disruptive. However, if your LP is already using shared CPs, and it has an acceptable number of logical CPs, that LP can be changed to use WLM LPAR CPU Management non-disruptively.

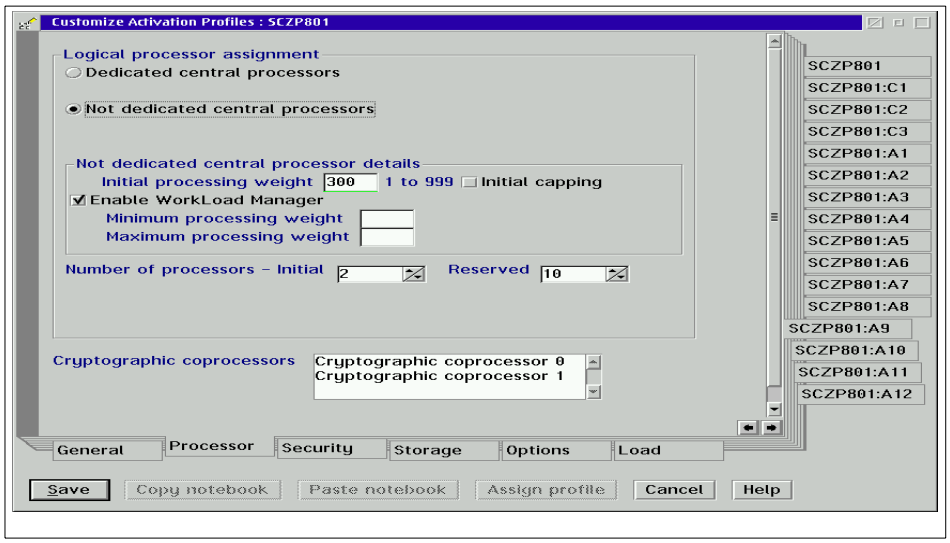
The disruptive changes consist of:

- For an LP that is currently using dedicated LPs, you need to update the Image Profile for that LP to indicate that you wish to use shared rather than dedicated CPs. This panel is shown on the next page.

- Define the maximum number of logical CPs you wish to have available to each LP.

  Once again, the HMC Image Profile is used to define the Initial number of logical CPs. The same panel is used to define the number of Reserved CPs. If you wish to change either of these values, you must deactivate and reactivate the LP. However, once a number of Reserved CPs has been defined, those CPs can be brought online without a further disruption. For this reason, we recommend that the total of the values in the Initial and Reserved fields should be equal to the maximum number of CPs that can be installed on the CPC. This allows you to upgrade the CPC with additional CPs and access those CPs without disrupting the LP.

  We recommend setting the Initial field to be equal to the total number of shared CPs currently installed on the CPC—especially for production LPs. For development, you may specify a smaller value if you wish to reduce the amount of CP resource that those LPs can consume. The Image Profile is shown in the following chart.



Those are all the disruptive changes that may be required. Depending on your current configuration, you may not even require these changes, and therefore can avoid any disruption.

The following attributes may also need to be modified in the Image Profile, to ensure that the LP comes up correctly after the next time the LP is activated. These attributes can also be changed non-disruptively as described in 5.5.1, “HMC Change LPAR Controls panel” on page 135:

- Remove any capping.

  LPs in an LPAR Cluster cannot be capped. Therefore, you should remove any capping by ensuring that the check box for Initial Capping is not checked.

- Set the Initial processing weight.

  This becomes the LP's current weight when it is first IPLed. It is also the weight that an LP in WLM Compatibility mode uses as its current weight.

- Set the Minimum and Maximum processing weights.

  The Minimum and Maximum weights set the lower and upper limits for weights which WLM LPAR Weight Management will assign to an LP.

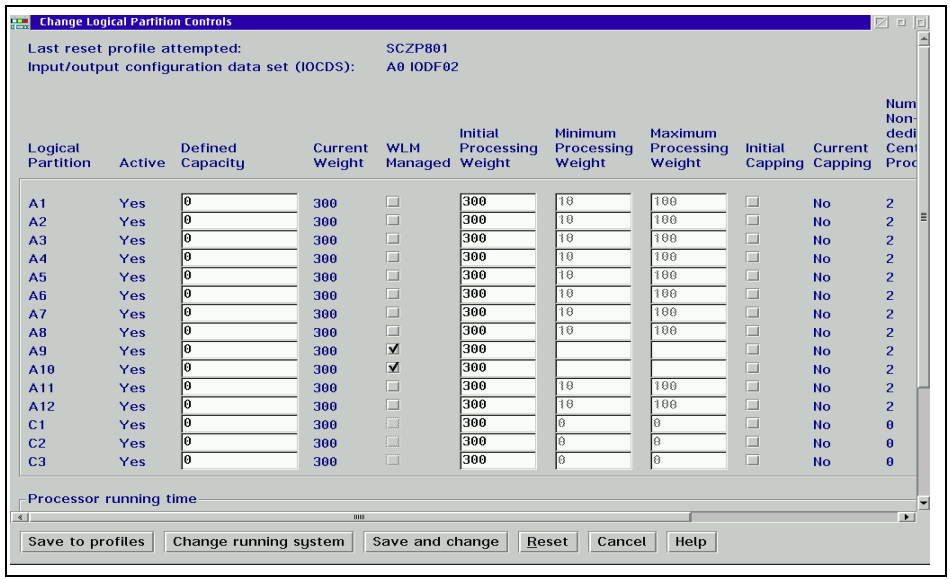
  These two weights can be used to disable WLM LPAR Weight Management while leaving WLM LPAR Vary CPU Management active. To do this, set the Minimum and Maximum Processing Weights equal to the Initial weight.

  To give WLM the maximum flexibility to adjust the weights of the LPs in the LPAR Cluster, these fields should be left blank. WLM will always ensure that every LP is left with enough weight to maintain its share of sysplex processing.

- Check the "Enable Workload Manager" check box.

  This is the final step in activating WLM LPAR CPU Management at the hardware level. Once this check box is selected, and providing all the other changes have been made as documented in the previous sections of this chapter, the LP will use WLM LPAR CPU Management starts to manage its weight and number of online CPs every time it is activated.

5.5.1 HMC Change LPAR Controls panel



The figure above shows the Change Logical Partition Controls panel on the HMC. This panel is used to dynamically change the following:

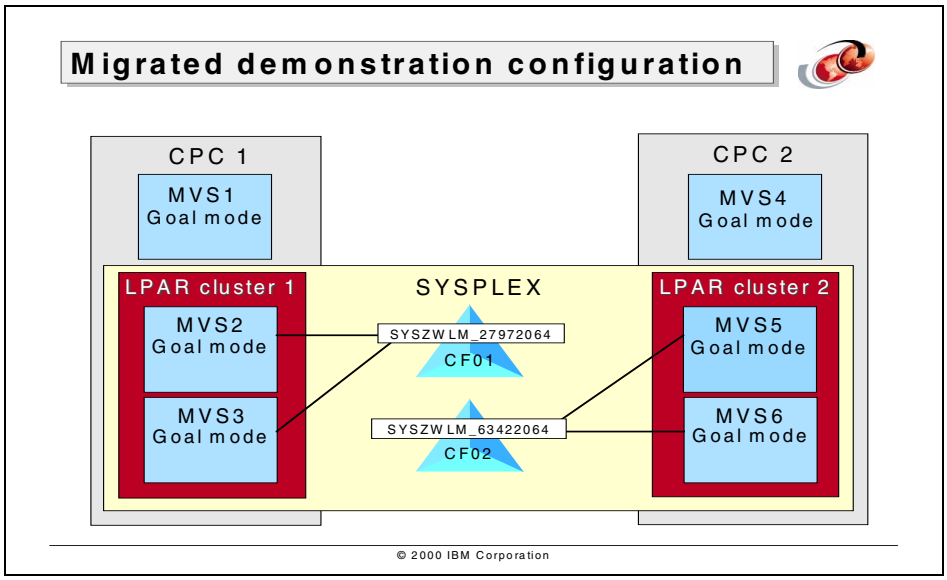
- WLM Managed status
- Initial Processing Weight
- Minimum Processing Weight
- Maximum Processing Weight
- Initial Capping

All these functions can be changed dynamically. However, don't forget that when you want to make the changes permanent you have to update the Image Profiles as well.

To enable WLM LPAR CPU Management functions dynamically, you must update the LP attributes (Initial, Minimum, and Maximum weights and possibly Initial Capping) then select the "WLM Managed" check box. You can select the WLM Managed check box one LP at a time if you want to implement WLM LPAR CPU Management in a phased manner.

In addition, the field entitled "Current Capping" indicates whether the LPAR is currently being soft-capped. At the time of writing, this is the *only* real-time way to find out if an LP is being soft-capped. It is planned to add this information to RMF Monitor III in z/OS 1.2.

5.6 Migrated demonstration configuration



The example above shows our demonstration configuration after it has been migrated into the two LPAR Clusters. MVS2 and MVS3 are in LPAR Cluster 1. Both systems have a connection to the WLM cluster structure in CF01 called SYSZWLM_27972064. MVS5 and MVS6 are in LPAR Cluster 2. Both these systems have a connection to the WLM cluster structure in CF02 called SYSZWLM_63422064. WLM LPAR CPU Management is now active and is managing LPAR weights and number of logical CPs in each LPAR Cluster.

5.7 Summary

As you can see, the actual implementation itself is very simple. If you don't have to make any disruptive changes to any of the LPs, you could easily implement WLM LPAR CPU Management for one or more LPs in a morning—including time for coffee breaks!

As we said at the start of the planning chapter, the most time-consuming part of implementing WLM LPAR CPU Management is deciding how you are going to set up your LPs. Once you have decided on that, activating it is very easy.

6

Operating WLM LPAR CPU Management

Operating a system that is using WLM LPAR CPU Management is not complex—in fact, the fact that it is being used should be transparent. Day-to-day operations of the LPAR Cluster should be minimal. Because it is an extension of WLM's functions, the day-to-day operational considerations are similar to what is performed today for WLM.

This chapter covers operations in two parts. The first part looks at what actions are performed from what interface, and what these actions allow an operator to do. The commands and controls are broken into 3 sections:

- Actions at the HMC
- Actions using operator commands and SYS1.PARMLIB members

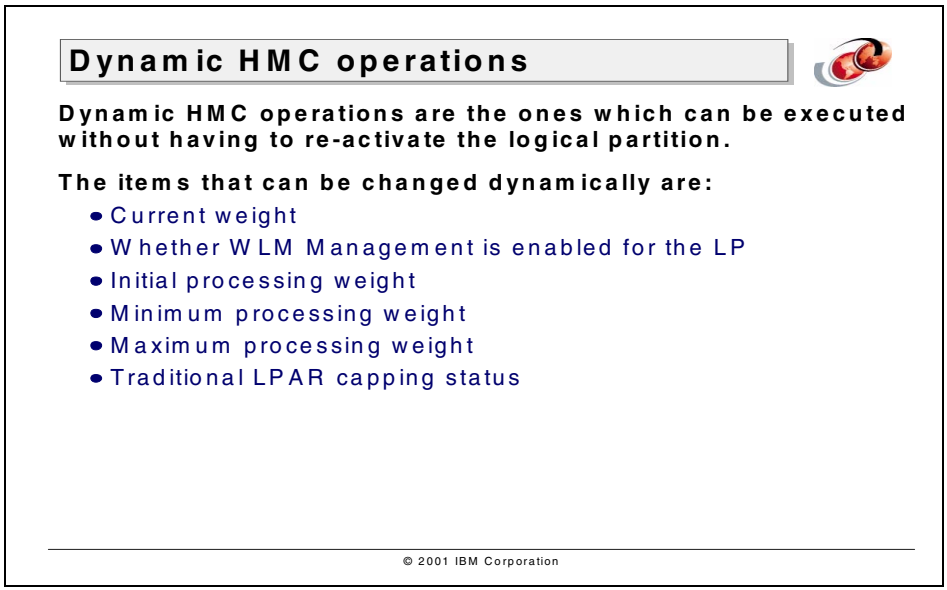
  These two are covered together as the change to SYS1.PARMLIB requires a z/OS command to make it active.
- Actions using SYS1.PARMLIB

Once we look at where the controls are and what we can do with them, we look at some of the scenarios in which you use these controls. In some scenarios you may have a few options which we outline in the pertinent sections. The scenarios are as follows:

- Getting the status of WLM LPAR CPU Management in the LPAR Cluster

- Finding out the current weight of each of the LPs in the LPAR Cluster
- Adding or removing an LP from an LPAR Cluster
- Enabling or disabling WLM LPAR CPU Management or either of the components which make up WLM LPAR CPU Management
- Operational considerations when making changes to WLM LPAR CPU Management
- Automation considerations
- Cluster structure operations
- Problem determination

6.1 Dynamic HMC operations



The following are the items that can be changed dynamically using the Change Logical Partition Controls panel on the HMC:

- Determine the current weight of an LP.

  This is shown in the Current Weight field.

- Switch off WLM LPAR CPU Management for an LP.

  Removing the check from the WLM Managed check box disables WLM LPAR CPU Management for the associated LP. The weight will be reset to the Initial value, and any CPs that were varied off by WLM will be varied back online.

  Other LPs in the LPAR Cluster continue to use WLM LPAR CPU Management. However, this does affect them: the remaining LPs can no longer change the number of weight units on this system.

- Change the Initial Processing Weight.

  The Initial weight becomes the LP's Current Weight when the LP is IPLed or is placed into WLM Compatibility mode.

Note that if you change the Initial Weight while the LP is running, the delta from the old Initial Weight is distributed across all the LPs currently in the LPAR Cluster in proportion to their current weights. This is either plus or minus. You will not set the current weight set to the initial weight. In effect you are changing the relative weight of the whole LPAR Cluster in proportion to whatever else is on the CPC; not the LP itself. Certainly the new value will be used as a future start-up value for the LP the next time it is activated.

- Change the Minimum or Maximum Processing Weights.

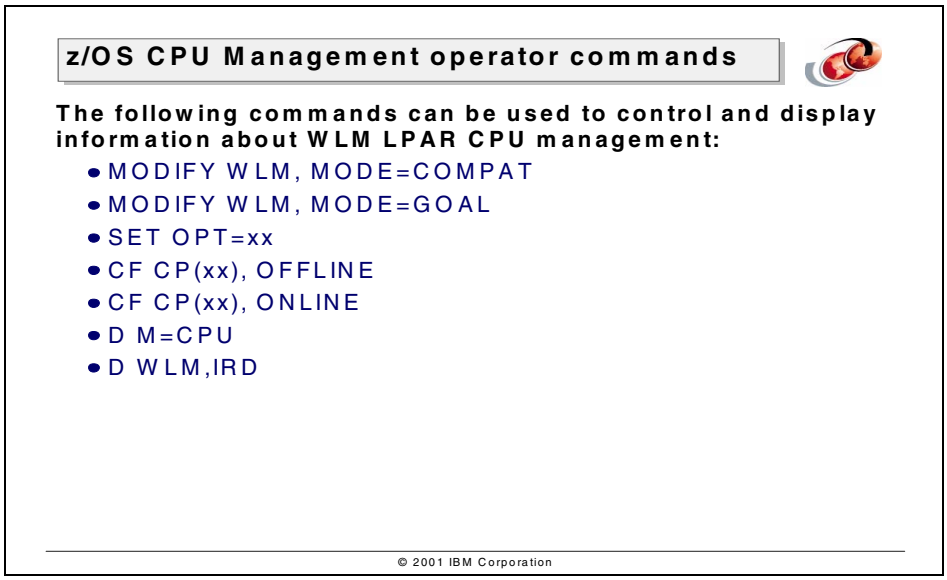
  These are used to limit the minimum or maximum weight that WLM LPAR Weight Management uses when changing an LP's current weight. These could be used to temporarily disable WLM LPAR Weight Management. By setting these values equal to the current weight, the LPAR's current weight can no longer be changed. Once again, any action that disables WLM LPAR Weight Management on one system in the cluster affects the other systems' ability to move CP resources between LPARs.

- Select traditional LPAR capping.

  This box cannot be selected while the WLM Managed check box is selected. If you wish to cap an LP, you must stop using WLM LPAR CPU Management for that LP first.

These are the only changes that an operator would be expected to make. Other changes, such as number and type (dedicated or shared) of CPs, are not covered here since these would be made before the LPAR is activated and are not considered operational actions.

6.2 z/OS operator commands



There are a number of operator commands that have an impact on the operation of an LPAR Cluster. Some are existing commands that have been updated for LPAR Cluster support. The following commands are available:

- MODIFY WLM,MODE=COMPAT

  This command places the system that it is entered on into WLM Compatibility mode. This command always has a scope of just the system that it was entered on. When it is issued, the LP's current weight is reset to its Initial weight and all WLM-offline CPs are brought online. This may affect other systems in the LPAR Cluster. If this LP needs to increase its current weight to achieve its Initial weight, it takes the weight units from the other LPs in the LPAR Cluster. This may leave them short of CP resources. For a description of all possible outcomes, see 4.6, "WLM mode considerations" on page 113. Alternatively, it may need to lose some weight units which may affect the performance on this system.

  Also consider that it is likely that the change will result in there being too many logical CPs online for the LP weight. This is because we recommend that all LPs in the LPAR Cluster be defined with a large number of logical CPs. Normally, some of these will be offline, but they will all come back online as soon as the system enters Compatibility mode, thus increasing the LPAR overhead. You may be required to vary some logical CPs offline to reduce the LPAR overhead.

- MODIFY WLM,MODE=GOAL

  This command changes a system from Compatibility mode to Goal mode. If all hardware actions and other software actions and CF tasks to enable WLM LPAR CPU Management are complete, this LP will now go back into WLM LPAR CPU Management mode.

- SET OPT=xx

  The VARYCPU parameter in the IEAOPTxx member controls the status of WLM LPAR Vary CPU Management provided that all other hardware and software actions have been completed. When this capability is active, WLM varies logical CPs online and offline based on the current capacity requirements of the LP. This capability can be turned off by updating the IEAOPTxx member and issuing a SET OPT=xx command.

  It is unlikely that an operator will update the IEAOPTxx member to switch this on or off. It is more likely that the Systems Programmers will provide the Operators with two members: one that will turn WLM LPAR Vary CPU Management off, and another that will turn it on. Similarly, it is unlikely that an operator will unilaterally decide to turn this off; in most cases, it will be at the request of a systems programmer.

  Remember that WLM will not manage any logical CPs that were taken offline by an operator command. If WLM LPAR Vary CPU Management is active, you should allow it to do its job by making sure that there are no logical CPs that were taken off by an operator. The D M=CPU command can be used to display the status of each logical CP.

  At the time of writing, there is no way to display whether WLM LPAR Vary CPU Management is enabled or not. However, if the result of the D M=CPU command shows some CPs with a status of -W, you know that WLM LPAR Vary CPU Management is currently active. If there are no CPs with this status, that does not necessarily mean that WLM LPAR Vary CPU Management is not active—it could just be that WLM has determined that the LP currently needs all its logical CPs to be online.

- CF CP(xx),ONLINE/OFFLINE

  An operator uses this command to take logical CPs online and offline from z/OS. Once an operator has taken a logical CP offline with this command, WLM cannot bring this CP online. Only an operator can bring this logical CP back online.

- D M=CPU

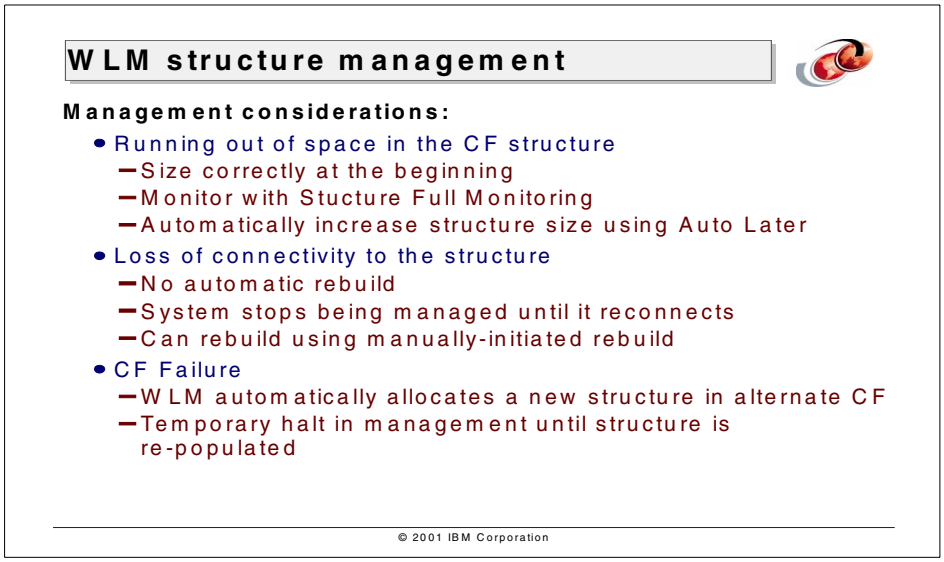
  This command shows the status of logical CPs. If they are offline, it shows whether they were taken offline by an operator command or by WLM LPAR Vary CPU Management. The following command output is from an LP with seven Initial CPs, two of which are currently offline because WLM took them offline:

```
D M=CPU
IEE174I 15.52.26 DISPLAY M 594
PROCESSOR STATUS
ID  CPU                  SERIAL
0    -W
1    -W
2    +                   2215342064
3    +                   3215342064
4    +                   4215342064
5    +                   5215342064
6    +                   6215342064
CPC ND = 002064.109.IBM.02.000000051534
CPC SI = 2064.109.IBM.02.0000000000051534
CPC ID = 00

+ ONLINE    - OFFLINE    . DOES NOT EXIST    W WLM-MANAGED
```

- D WLM,IRD

  This new command, delivered by APAR OW48601, provides system-specific information about both WLM LPAR Vary CPU Management and WLM LPAR Weight Management. More information about this command, and sample output, is available in 12.1.6, “D WLM,IRD command” on page 327.

6.3 Managing the WLM CF structure



The availability of the WLM CF structure is vital to the operation of WLM LPAR CPU Management. The structure is used to hold information about the workloads in each system in the LPAR Cluster. If this information is not available, WLM LPAR CPU Management cannot make any changes to the associated systems.

The possible problems related to the structure are:

- The structure might run out of storage.
- One system may lose connectivity to the structure.
- The CF containing the structure might fail.

If the structure fills up, you will get an IWM053I message from WLM:

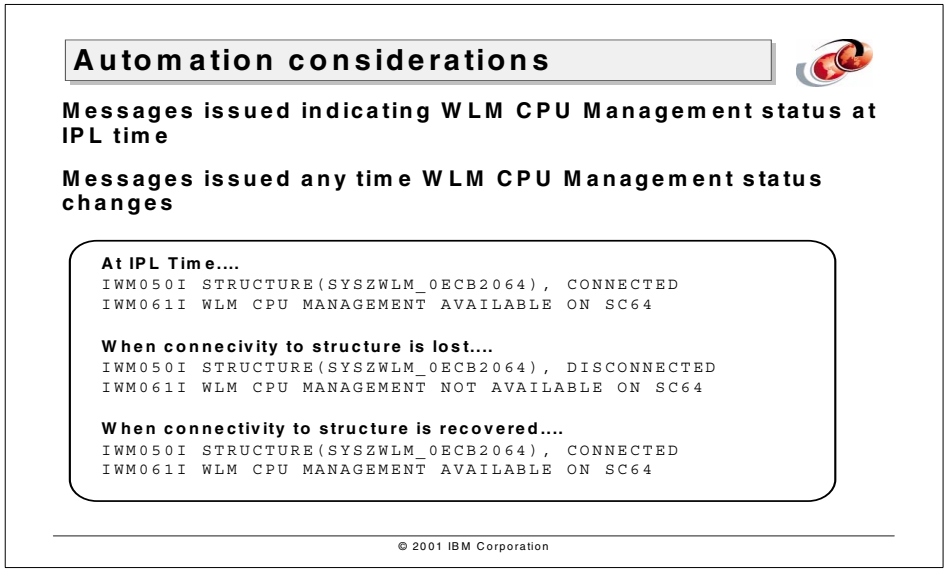
IWM053I STRUCTURE (SYSZWLM_12342064) FULL, ALLOCATE LARGER STRUCTURE VIA SETXCF ALTER OR REBUILD

In this case, you should issue the SETXCF START,ALTER command to increase the structure size. A preferable solution is to use AUTO ALTER for the structure by specifying (ALLOWAUTOALT(YES)) for the structure in the CFRM policy. In this case, XES will automatically increase the structure size when it exceeds the specified threshold.

If one system loses connectivity to the structure, WLM will not automatically rebuild. This is because System Managed Rebuild does not support rebuild in case of unplanned configuration changes. WLM LPAR CPU Management will stop making any changes to the system that lost connectivity. In this case, you must issue the SETXCF START,REBUILD,....,LOC=OTHER command for the WLM structure. This will cause the structure to be rebuilt to the alternate CF. As soon as the rebuild completes, all the systems will reconnect and WLM LPAR CPU Management will continue for all the systems in the LPAR Cluster.

Finally, if the CF containing the WLM structure fails, WLM will detect this and immediately stop doing any WLM LPAR CPU Management. It will also automatically allocate a new structure in one of the other CFs specified on the PREFLIST. After the new structure is allocated and all the systems have connected, WLM LPAR CPU Management will resume processing for all the systems in the LPAR Cluster.

6.4 Automation considerations



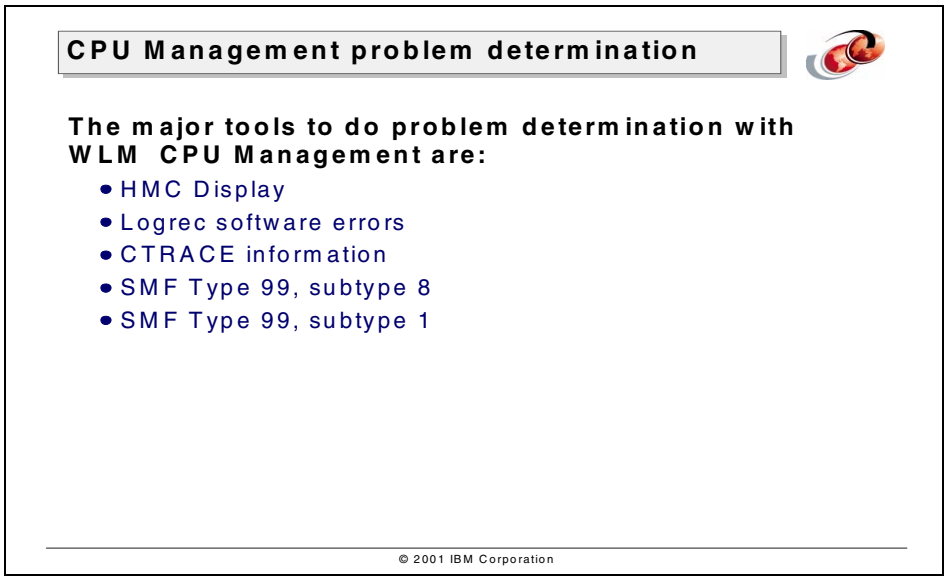
For monitoring purposes, you may wish to get your Automated Operations product to monitor for messages indicating that the status of WLM LPAR CPU Management has changed.

During IPL, WLM issues messages IWM050I and IWM061I, informing you that it has connected to the WLM LPAR Cluster structure. The same message numbers are used, with different text, for the WLM multisystem enclaves structure, so you will need to get your automation to check the text of the message.

Should you lose connectivity to the structure, WLM LPAR CPU Management will be stopped for the system that lost connectivity. In this case, you should see XCF and XES messages indicating that connectivity has been lost. You will also receive the same two messages from WLM, IWM050I and IWM061I, again with different text indicating that the structure has been disconnected and WLM LPAR CPU Management has stopped for that system.

When connectivity is reestablished, you will once again receive the XCF and XES messages, followed by the IWM050I and IWM061I messages indicating that WLM has connected to the structure again, and WLM LPAR CPU Management is once again active for that system.

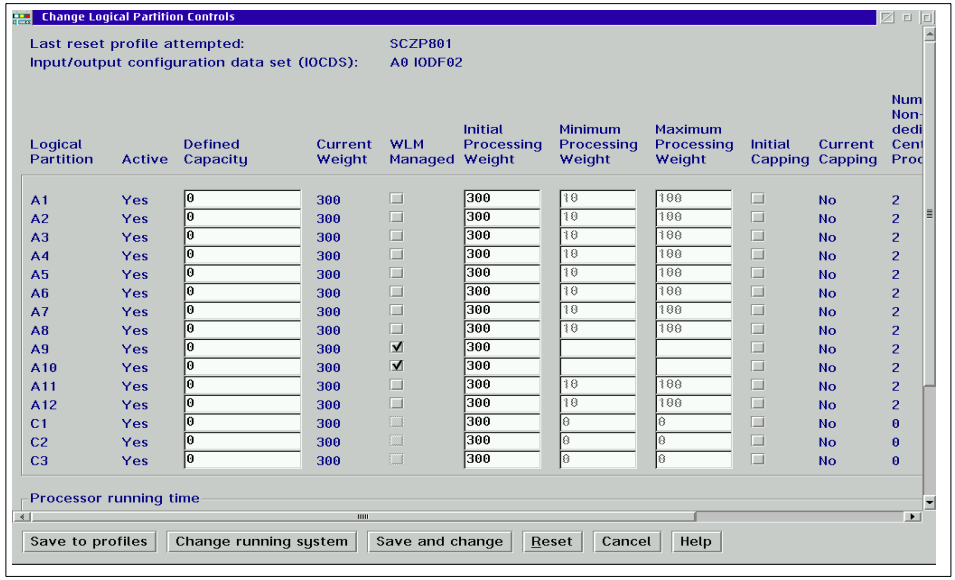
6.5 Problem determination



If there is a “problem” with WLM LPAR CPU Management, it is likely to be for one of the following reasons:

- WLM LPAR Weight Management has taken some weight from an LP to help another “more important” LP. As a result, lower importance workloads in the donor LP may experience a sudden increase in response times.
- An LP’s rolling 4-hour average utilization has reached its Defined Capacity, and as a result it has been soft-capped. Once again, some workloads in that LP may experience a sudden decrease in the amount of CPU available to them, with an attendant increase in response times.

In the first instance, WLM will happily take resource from a workload with lower importance if that resource is needed elsewhere. As a result, it is possible that the lower importance workloads in the donor LPs will see a degradation in response time when the weight is moved. This could lead to calls from users. If this happens, the operator’s checklist for such events should be expanded to include checking the current and initial weights of the LP containing the suffering workload. If it transpires that the current weight is much smaller than the initial weight, this information should be passed on to the Systems Programmers. If the suffering workload’s importance is lower than it should be, then it should be adjusted.



Another potential action is to increase the minimum weight of the impacted LP. Both of these actions, however, should only be carried out under the advice of a Systems Programmer.

The second instance will only arise if you are using sub-capacity charging—a feature of Workload Charging that lets you pay for software based on the size of an LP rather than the capacity of the whole CPC. In this case, if the amount of CPU the LP is currently using significantly exceeds its Defined Capacity and it is then soft-capped, the amount of CPU available to the LP will decrease sharply. This will cause elongated response and elapsed times for the lower importance workloads in the LP. Once again, this may lead to calls from the end users. Therefore, the list of investigative actions for the operators should be expanded to include checking the “Change Logical Partition Controls” panel in HMC to see if the LP in question is currently being soft-capped. The “Current Capping” column indicates if the LP is being capped, either because of soft-capping, or because of traditional LPAR capping. At the time of writing, this is the only way to find out in real time if an LP is being soft-capped.

Unfortunately, WLM LPAR CPU Management does not provide much easily-accessible information about its actions as they are taken. It is possible to use the RMF Postprocessor report to get after-the-fact information about LPAR weights, LP soft capping, and the number of online CPs in an LP. In real time, you can use the D M=CPU command to find out how many CPs are currently online,

and you can use the “Change Logical Partition Controls” panel on the HMC to find out the current weights for the LPs. What these commands do not tell you is what actions WLM LPAR CPU Management has implemented over the last few minutes.

If you find it necessary to do some debugging on WLM LPAR CPU Management, there are a number of tools to make it easier for IBM Service to debug WLM-related problems. Information about WLM actions is kept in the WLM address space, and also optionally written to the SMF data sets. In addition, you can use the Component Trace facility to gather additional information about WLM processing. You would normally only turn this on when requested to do so by IBM Service. To enable this, issue the TRACE CT,16M,COMP=SYSWLM command. This will cause Component Trace to hold 16 MB worth of trace data for WLM in fixed Common storage. If you do not have 16 MB available, you can specify a number smaller than 16M.

There is a new SMF record related to WLM LPAR CPU Management. The SMF Type 99, Subtype 8 records contain summary information about WLM LPAR CPU Management decisions. At the time of writing however, there is no IBM-supplied tool to process these records.

SMF Type 99 Subtype 1 contains WLM trace data that may be used to diagnose WLM actions. However, turning on the collection of these records will generate significant amounts of SMF data, so you should generally only enable these records if specifically requested to do so by IBM Service. Also, 15 minutes worth of this data is always kept in the WLM address space, so a dump of this address space will provide this data without having to suffer the overhead of filling up your SMF data sets with all those records. The trace codes in the Subtype 1 records are described in *OS/390 MVS Programming: Workload Management Services.*

There is one other thing to be aware of, which is not strictly related to problem determination, but is associated with actions taken if you encounter a problem. When a system is removed from the sysplex, and the LP is Reset, LPAR resets the weight of the LP back to the Initial value. As a result, the weight of other LPs in the LPAR Cluster may be affected.

7

Performance and tuning for WLM CPU Management

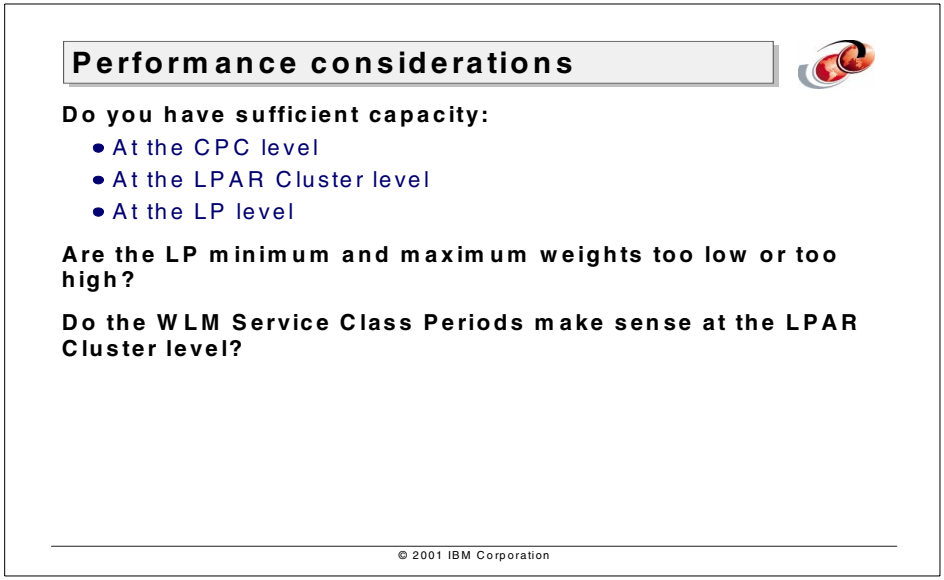
The whole intent of WLM LPAR CPU Management is to make the system self-tuning to the extent of moving CPU resource to where it is needed most. As a result, it should *reduce* the need for day-to-day performance monitoring and tuning, rather than *increasing* it.

Having said this, it does introduce some new considerations, and we discuss these in this chapter.

However, before we go further, we should stress that the effect of WLM LPAR Weight Management increases as the utilization of an LP approaches the capacity guaranteed by its Initial weight. At utilizations below this, the weight of an LP does not have as significant an impact on the amount of CPU resource that it receives.

In relation to WLM LPAR Vary CPU Management, assuming that you have specified a sufficiently large Initial number of logical CPs, there is really no tuning involved. Given the power of modern CPCs, and the way WLM LPAR Vary CPU Management always keeps a buffer of spare CP capacity, it is very unlikely that the load on a system could grow so quickly that you find yourself with an insufficient number of CPs.

7.1 WLM LPAR CPU Management considerations



Because WLM LPAR Weight Management manages the weight of an LP as part of a group (the LPAR Cluster), you need to be aware of the capacity consumed by the LPAR Cluster as a whole, as well as the capacity used by each LP. Remember that the weights are only redistributed between the LPs in the LPAR Cluster, so if all the LPs are having problems, it may be that the total weight of the LPAR Cluster needs to be increased.

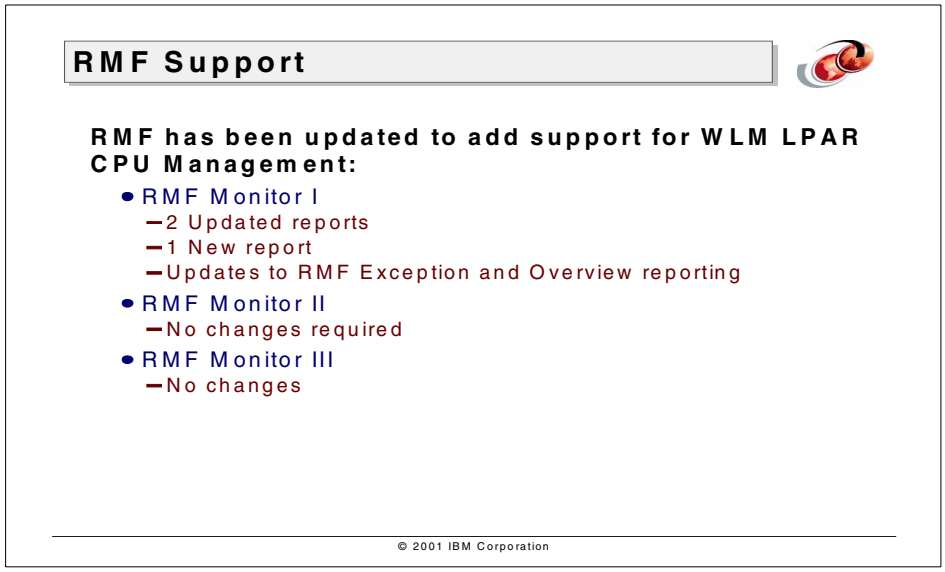
Another thing to consider is the minimum and maximum weights of each LP. The recommendation is to *not* specify a minimum or maximum weight, in order to give WLM the maximum flexibility to adjust the weights of the LPs within the LPAR Cluster. If no minimum is specified, WLM will reduce the weight as far as is required to move CP resource to another LP in need of that capacity; however, it will never reduce the weight of an LP to less than about 5% of the total capacity of the CPC. Note, however, that if you specify a minimum weight that is less than this value, WLM will abide by the limits you set, and possibly reduce the LP weight to that value. Using such a low weight can impact the ability of the LP to process sysplex requests in a timely manner, potentially leading to a sysplex disruption.

If the Service Class Periods (SCPs) within an LP consistently fail to meet their goal, and those SCPs are important to you, you have two choices:

- Review the WLM goals and importances of those SCPs. Maybe the goals are too stringent, or maybe the importance assigned to the SCP does not reflect the actual importance of the SCP to the business. If this is the case, you should adjust your WLM policy accordingly.
- Increase the minimum weight of the LP if CPU delay is the largest delay. This is the less attractive of the two options. The intent of WLM LPAR Weight Management is to help the SCPs that you have said are important. By narrowing the band of weights that WLM LPAR Weight Management can manage, you are tying its hands and circumventing the whole idea of WLM, which is that you define the importance and realistic goals for your SCPs and it will manage the resources of the system to achieve those goals.

The final thing to consider is the multisystem effect of your goals. As we discussed in 4.2, “WLM Policy definitions” on page 105, it is important that the importance of a given SCP makes sense at the whole sysplex level. It is not sufficient to check that the high importance SCPs are meeting their goal (although this is obviously important); you also have to make sure that SCPs that contain production SCPs are not suffering while SCPs that contain non-production work are exceeding theirs. The starting place for this work, in terms of RMF reports, is the RMF Workload Activity Report.

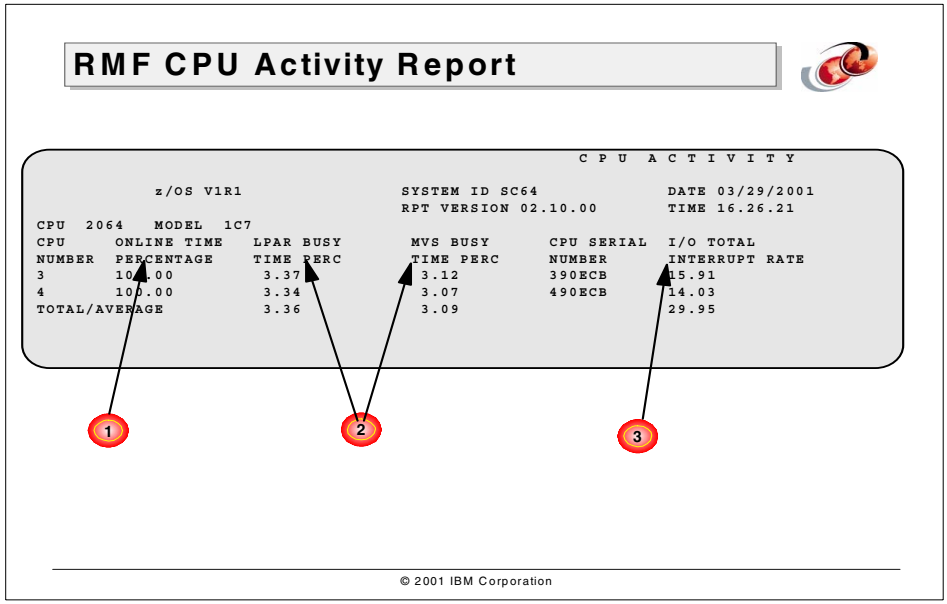
7.2 RMF reports



RMF has been updated to add specific support for WLM LPAR CPU Management. This support is based on RMF V2R10, with specific support added by APAR OW46477.

The only RMF reports affected by WLM LPAR CPU Management are the CPU-related Postprocessor reports. At the time of writing, there are no changes to Monitor II or Monitor III.

7.2.1 RMF Monitor I - CPU Activity Report



There are changes to the existing RMF Monitor I postprocessor CPU reports in support of WLM LPAR CPU Management. Two of the existing reports have been updated, and a new report, the LPAR Cluster report, has been added. These reports are produced when you run the RMF Postprocessor program (ERBRMFPP), specifying the REPORTS(CPU) option. To get all the information in the report, you need a merged SMF input file (or you can use the RMF SMF buffer if all the systems in the LPAR Clusters are in the same sysplex). To obtain global reports, the synchronization of RMF with SMF is a requirement.

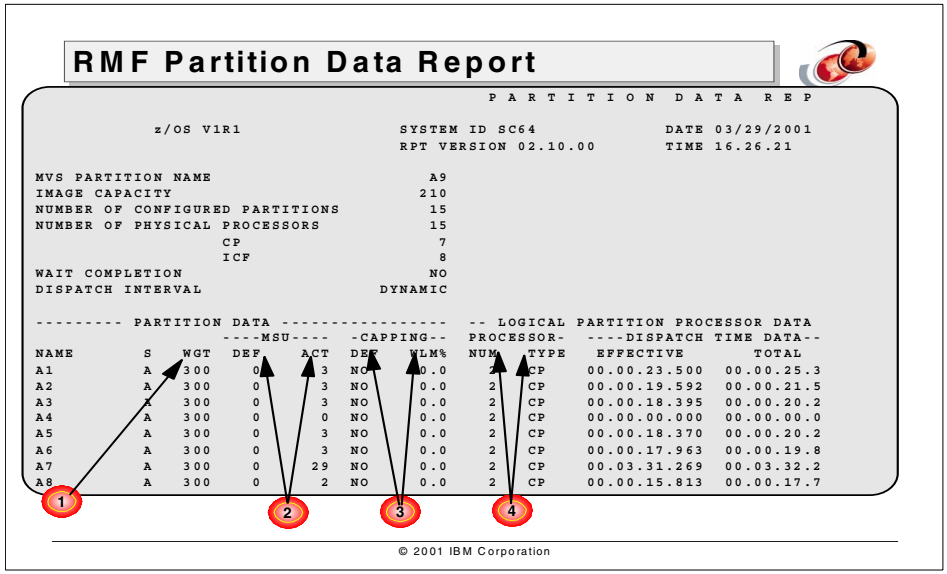
The first report that has changed is the RMF CPU Activity report. This report is shown in the figure above and contains the following changes:

1. The column entitled “Online Time Percentage” provides information about how long the related CP was online to that LP in the interval. If this field contains a value less than 100, it is a probable indicator that WLM LPAR Vary CPU Management took that CP offline for some of the interval.
2. The columns entitled “LPAR Busy Time Perc” and “MVS Busy Time Perc” are adjusted for the time that the CP was online during the interval.

   If the MVS Busy Time Perc value is significantly larger than the LPAR Busy Time Perc, this is an indicator that the LP is CPU-constrained and could benefit from a larger weight or more logical CPs.

3. The column entitled “I/O Total Interrupt Rate” is also adjusted to take into account if the CP was not online for the entire interval. In this case, the rate is the Interrupt rate during the time the CP was online rather than being averaged over the whole interval.

7.2.2 RMF Monitor I - LPAR Partition Report



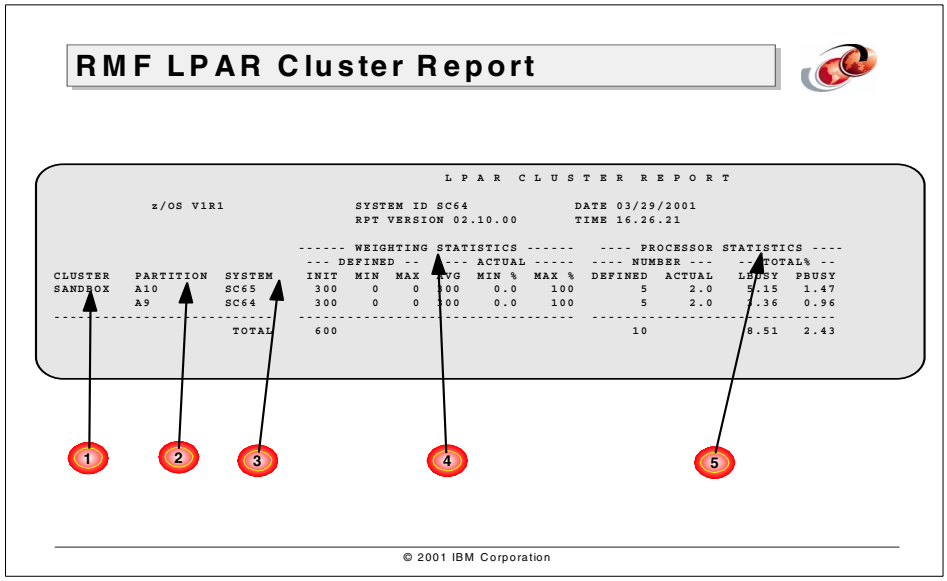
The next report to be updated in support of WLM LPAR CPU Management is the Partition Data Report. This report has been extensively updated, and as you can see in the figure above (which only contains a subset of the report!), there is not much room left to add any additional information. The changes specifically in support of WLM LPAR CPU Management are:

1. The column entitled “WGT” contains the average weight of the LP over the RMF interval.
2. There is a new pair of columns, entitled “MSU”. These are used in conjunction with Workload Charging. The column entitled “DEF” is the defined capacity of the LP as specified on the HMC. The column entitled “ACT” is the actual average capacity used over the interval. This is an indicator of the *average* rolling 4-hour average utilization for this LP over the RMF interval.
3. The next two columns, entitled “Capping”, indicate the use of traditional LPAR capping and the new Soft Capping. The column entitled “DEF” indicates whether traditional LPAR was enabled for this LP. If this field is set to YES, this LP cannot be managed with WLM LPAR CPU Management. The column entitled “WLM%” indicates if Soft Capping was invoked for this LP for some of the time during this RMF interval. Soft Capping is described in more detail in 2.15, “Relationship to IBM License Manager” on page 42. A small value in this field indicates that for a small amount of time in this interval the rolling 4-hour average reached the defined capacity for this LP. This is good, and shows that you are getting a benefit from Workload Charging. However, if you

consistently see large values in this field, that is an indicator that this LP is severely out of capacity and that you should consider increasing the defined capacity for this LP.

4. The next two columns, entitled “Processor”, provide information about the CP types and numbers over the interval. The column entitled “NUM” shows the average number of online CPs in this LP over the interval. The column entitled “TYPE” indicates the CP type—CP (for use by an S/390 operating system), ICF (for use by a CF LP), or IFL (for use by a LINUX LP).

7.2.3 RMF Monitor I - LPAR Cluster Report



The RMF LPAR Cluster Report is a brand new report, created specifically in support of WLM LPAR CPU Management. This report repeats some of the information available in the LPAR Partition Report; however, it also provides additional information and gathers together information for the systems within an LPAR Cluster. The fields in this new report (a portion of which is shown above) are:

1. Cluster

   This field contains the name of the sysplex that the systems in the LPAR Cluster belong to. If the sysplex consists of two LPAR Clusters (on two CPCs), the cluster name will appear twice, once for each CPC.

2. Partition

   This is the name of the LP as defined in HCD.

3. System

   This is the name of the system, as defined in the IEASYM or IEASYS members, and as reported to the CPC by XCF.

4. Weighting Statistics

   These columns contain the actual and defined weights of each of the LPs in the LPAR Cluster. The columns entitled “Defined” report the Initial, Minimum, and Maximum weights for the LP as defined on the HMC. The columns entitled “Actual” report the average weight over the interval, the amount of time spent at the Minimum weight (%MIN) and the amount of time spent at the Maximum weight (%MAX).

5. Processor Statistics

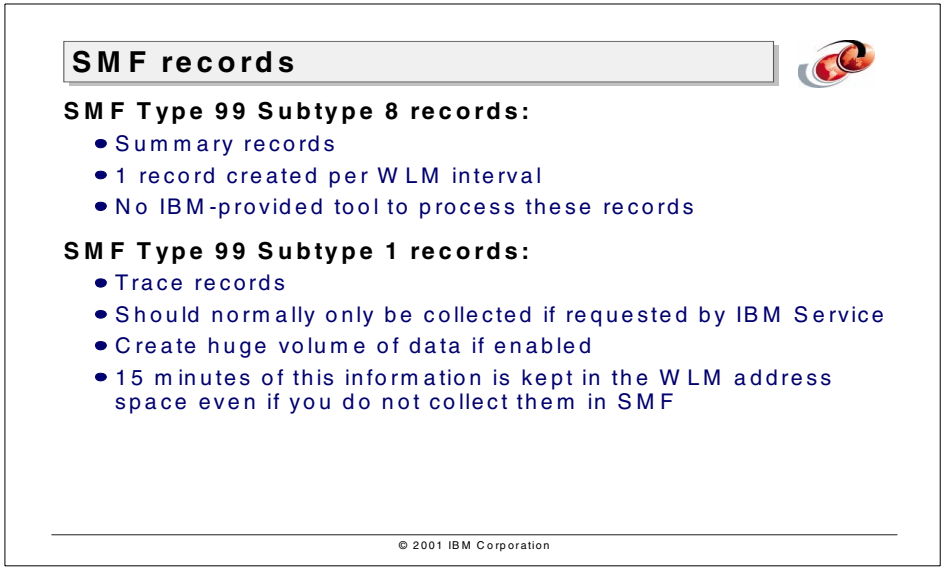
   These columns contain information about the number of logical CPs defined in the HMC for the LP, as well as the average number of online logical CPs over the interval.

   The columns entitled “%BUSY” show the logical and physical percent utilization for each LP.

7.3 Other RMF reports

There are no changes to the RMF Monitor II or Monitor III reports in support of WLM LPAR CPU Management in z/OS 1.1. Real-time information about the current weight of an LP can be obtained from the HMC, and information about the current number of online CPs can be obtained using the D M=CPU command. At the time of writing, it is planned to add some of this information to RMF Monitor III in z/OS 1.2.

7.4 SMF considerations



Information about the actions taken by WLM LPAR CPU Management is available in the SMF Type 99 records. The Subtype 8 records contain summary information about the decisions taken by WLM LPAR CPU Management. There is one Subtype 8 record written every WLM policy interval. The layout of the record is described in *z/OS System Management Facilities*. However, IBM does not provide any tool to process these records.

In addition, the SMF Type 99 Subtype 1 records contain detailed trace information about WLM processing. These records are very large, and will quickly fill up your SMF data sets if you collect them. As a rule, you should only enable the collection of these records (through the SMFPRM member) if specifically requested to do so by the IBM Service organization. However, even if you do not collect the Subtype 1 records, 15 minutes worth of the information in those records is kept in the WLM address space, and can be obtained by taking a dump of that address space.

7.5 A tuning methodology for WLM LPAR CPU Management

Based on early experiences with WLM LPAR CPU Management, we recommend that you enable this function, even if you are not currently encountering the scenarios that WLM LPAR CPU Management is designed to address, specifically:

- Critical SCPs are missing their goals because of CPU delays
- LPs in the LPAR Cluster are running at or above their Target LPx limit

While you may not be seeing these symptoms today, you probably will some day.

It may be wise to start with minimum and maximum weights that are 20% below and 20% above the current LP weight. Set the Initial weight to be equal to the LP weight that was specified prior to the introduction of WLM LPAR CPU Management. Gradually adjust the minimum and maximum as you gain experience, eventually removing the limits completely.

In relation to ongoing tuning of WLM LPAR CPU Management, there really is nothing to do. The function is specifically designed to be self-tuning. If you start reaching the point where important SCPs are missing their goals because of CPU delays, it may be necessary to increase the LPAR Cluster's share of the CPC, or to purchase additional capacity. Beyond that, there is really nothing you can do to tune the way WLM LPAR CPU Management functions.

Part 3

Dynamic Channel-path Management

8

Introduction to Dynamic Channel-path Management

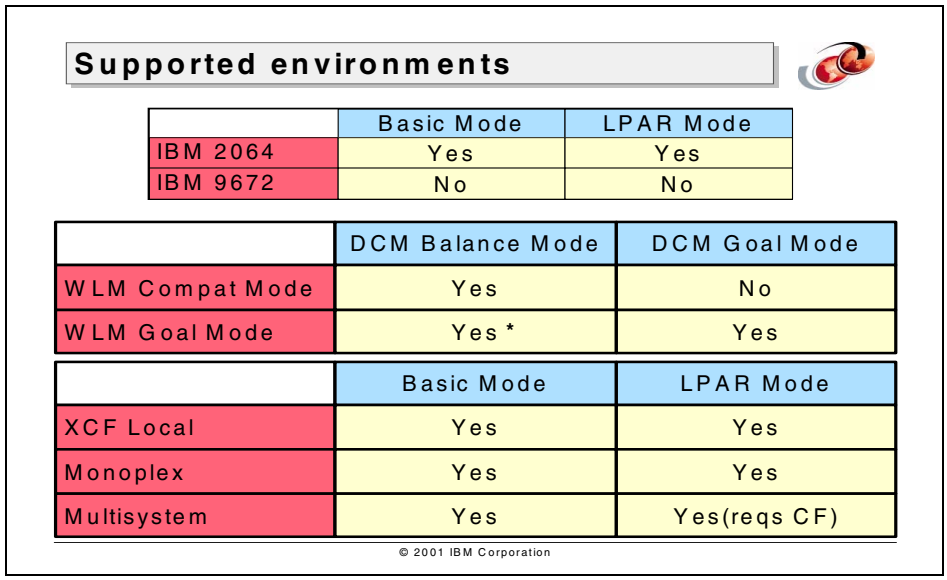
Dynamic Channel-path Management is a new capability, designed to dynamically adjust the channel configuration in response to shifting workload patterns. It is a function in Intelligent Resource Director, together with WLM LPAR CPU Management and Channel Subsystem I/O Priority Queueing.

Dynamic Channel-path Management (DCM) is implemented by exploiting new and existing functions in *software* components (in z/OS 1.1), such as WLM, IOS, HCD, Dynamic I/O Reconfiguration; and in *hardware* components, such as IBM zSeries 900 CPC, ESCON Directors, and DASD controllers.

DCM provides the ability to have the system automatically manage the number of ESCON and FICON Bridge (FCV) I/O paths available to supported DASD subsystems.

In this chapter, we review the benefits and applicability of DCM, and the remaining chapters help you understand how it works, and provide you with the information you need to implement DCM.

8.1 Supported environments



Before we get into describing the benefits of DCM, we will briefly describe its prerequisites, so you can determine whether you can use it in your environment.

We are just providing the main prerequisites here. Chapter 10, “Planning for Dynamic Channel-path Management” on page 257 provides detailed planning information.

In order to obtain the benefits of DCM, the system must be running on an IBM zSeries 900 or later CPC, with the required software levels (z/OS 1.1 or higher in z/Architecture mode). Both LPAR and Basic modes of operation are supported.

If you wish to share managed channels among z/OS images on the same CPC, the images must be members of the same LPAR Cluster, that is:

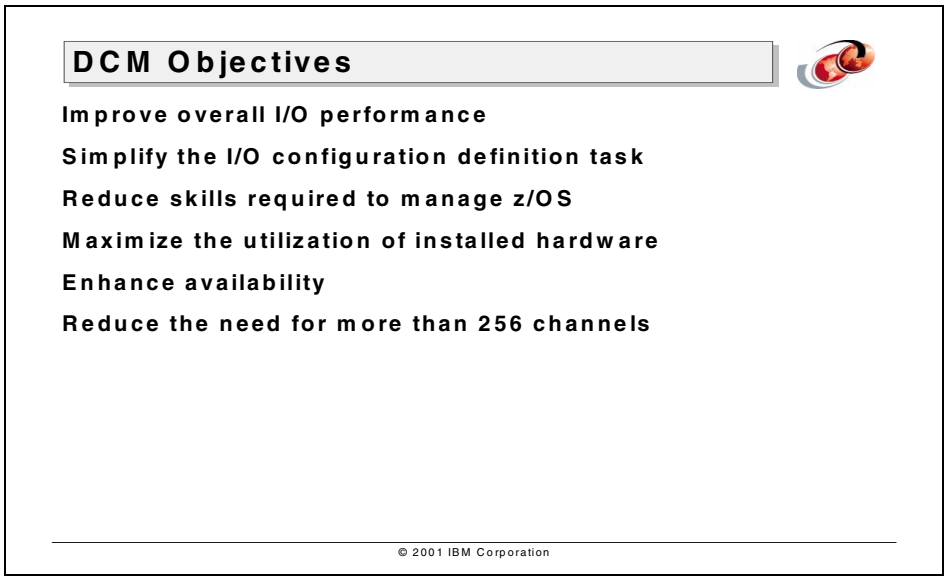
- They must be in the same Parallel Sysplex.
- They must all be running z/OS 1.1 or higher in z/Architecture mode.

WLM can be in either Goal mode or Compatibility mode. If WLM is in Goal mode, there is a greater benefit from DCM, but this is not a requirement.

DCM also has two modes, known as Balance mode and Goal mode. DCM always operates in Balance mode. In addition, if WLM is in Goal mode, DCM will also operate in Goal mode.

In relation to sysplex mode requirements, the system utilizing DCM can be IPLed in XCF Local, Monoplex, or Multisystem sysplex mode. If the system is in XCFLOCAL mode, you can use DCM and it will operate in DCM Balance mode. If the system is in MONOPLEX mode and WLM is in Goal mode, then you will have the benefits of DCM Goal mode in addition to those provided in DCM Balance mode. Finally, if the system is in MULTISYSTEM mode, you can use both DCM Balance and Goal modes, depending on whether all the systems in the LPAR Cluster are in WLM Goal mode or not. If the system is in MULTISYSTEM mode, it must be attached to a Coupling Facility in order to be able to use DCM, regardless of how many other systems are in the sysplex and regardless of how many of those systems are on the same CPC as that system.

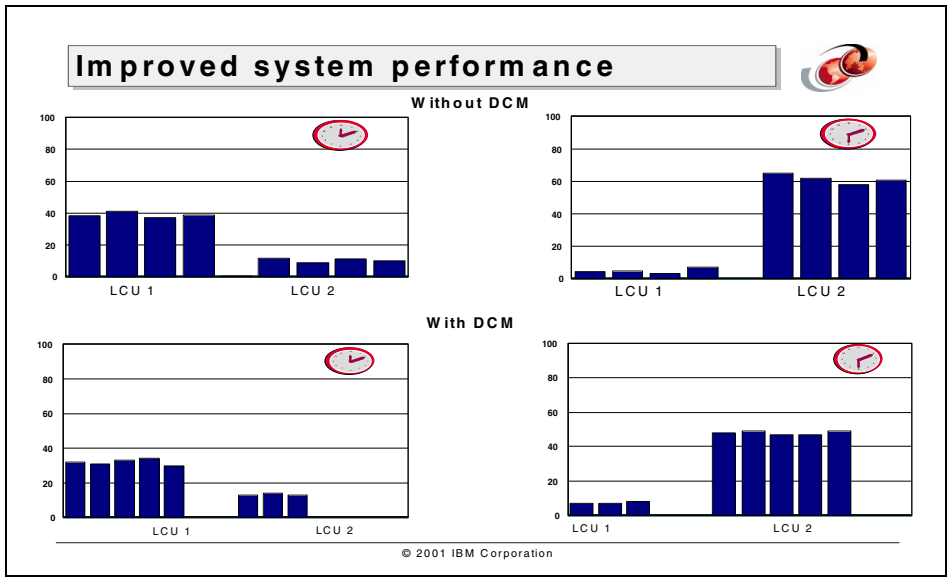
8.2 Value of Dynamic Channel-path Management



DCM provides value in a number of ways, including performance, availability, and reduced cost of computing. The specific benefits that DCM can provide for your installation depend on the profile of your environment.

The following pages describe the objectives and benefits of DCM in more detail.

8.2.1 Improved overall I/O performance



DCM can provide improved performance by dynamically moving the available channel bandwidth to where it is most needed. Prior to DCM, you had to manually balance your available channels across your I/O devices, trying to provide sufficient paths to handle the *average* load on every controller. This means that at any one time, some controllers probably have more I/O paths available than they need, while other controllers possibly have too few. Dynamic Channel-path Management attempts to balance the responsiveness of the available channels by moving channels to the controllers that require additional bandwidth, even when the system is in WLM Compatibility mode.

This chart shows the concept of how DCM operates. It shows the number of channels and their utilization when serving two logical control units (LCU) in two scenarios, one without DCM and another with DCM.

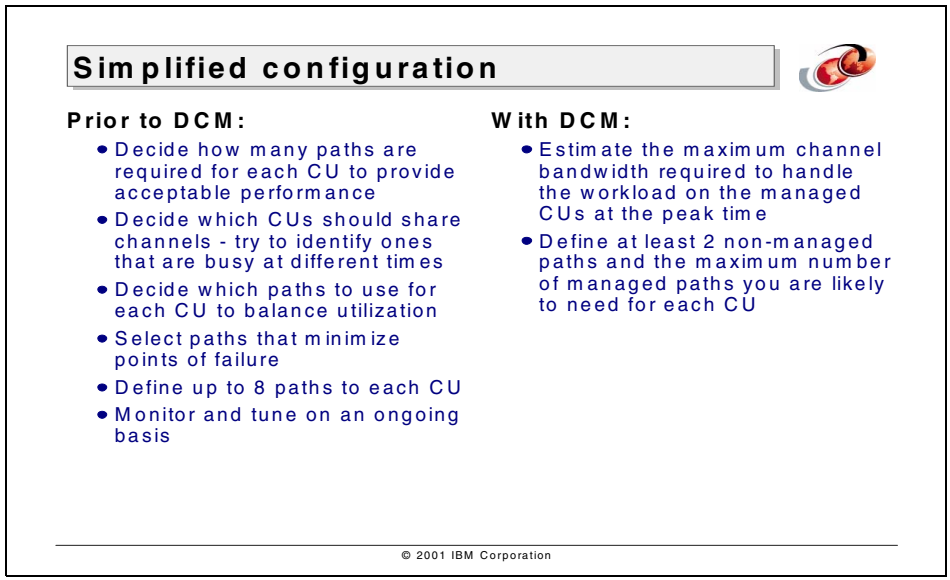
Without DCM, the channels are unbalanced in relation to their utilization due to I/O skew (uneven I/O activity in the LCUs) in the two LCUs. This is the normal mode of operation; it is unusual to see a configuration where all the control units are equally utilized at all times.

With DCM, the load on the LCUs is still uneven because DCM cannot do anything about that, but the channel utilization (and therefore the responsiveness) is more evenly balanced, resulting in decreased Pend times. In the prime shift, when LCU1 is much busier than LCU2, DCM moves two channels from LCU2 to LCU1 and also adds two more channels to LCU1. In the evening,

when the load on LCU1 largely disappears, and LCU2 gets much busier, the bulk of the channel bandwidth is moved to LCU2. In practice, DCM may not actually remove the channels from LCU1 in the evening time—this depends on the system workload at the time—but the chart is shown like this for illustration purposes.

Another advantage of Dynamic Channel-path Management is that you don't have to be as concerned about keeping different rules of thumb about how busy you should run your channels for every channel or control unit type you have installed. Instead, you just need to monitor for the signs of channel over-utilization (high channel utilization combined with high Pend times). This is made easier by changes in RMF whereby you now get a report showing the average aggregate utilization for all managed channels. This is discussed in more detail in 13.3, "Capacity planning considerations" on page 346.

8.2.2 Simplified configuration definition



Dynamic Channel-path Management also simplifies the task of defining your configuration. Without Dynamic Channel-path Management, you have to:

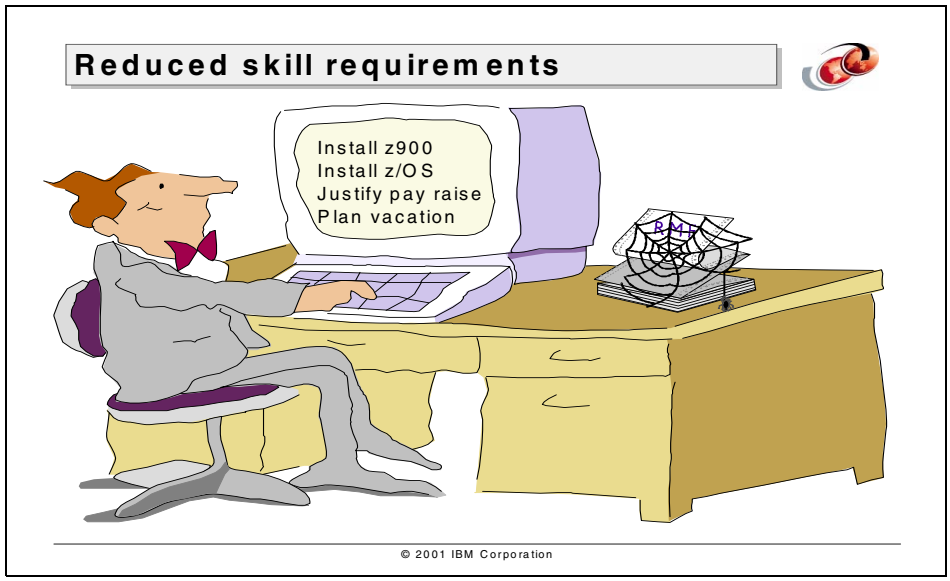
- Identify the bandwidth required for each LCU to provide acceptable performance at peak times.
- If you are forced to have more than one LCU per channel, you must identify LCUs that are busy at different times, and therefore are good candidates to share a channel. (You connect a single ESCON channel to several LCUs by using a Switch device, also called ESCON Director).
- Allocate your channels among LCUs in a manner that balances the overall channel utilization as much as possible.
- Select channels that provide the best availability (different channel cards in the CPC, different switches, and so on).
- Define the desired configuration (up to a maximum of 8 paths per LCU per LP[1]) to HCD.
- Monitor the configuration on an ongoing basis to ensure the performance and channel utilization remain within acceptable limits.

[1] Having more than 8 paths from a CPC to an LCU requires the use of Duplicate Device Number support. This is described in 10.4, “MIF considerations” on page 283.

Because Dynamic Channel-path Management can dynamically move I/O paths to the LCUs that are experiencing channel delays, you can reduce the CU/channel-level capacity planning and balancing activity that was necessary prior to Dynamic Channel-path Management.

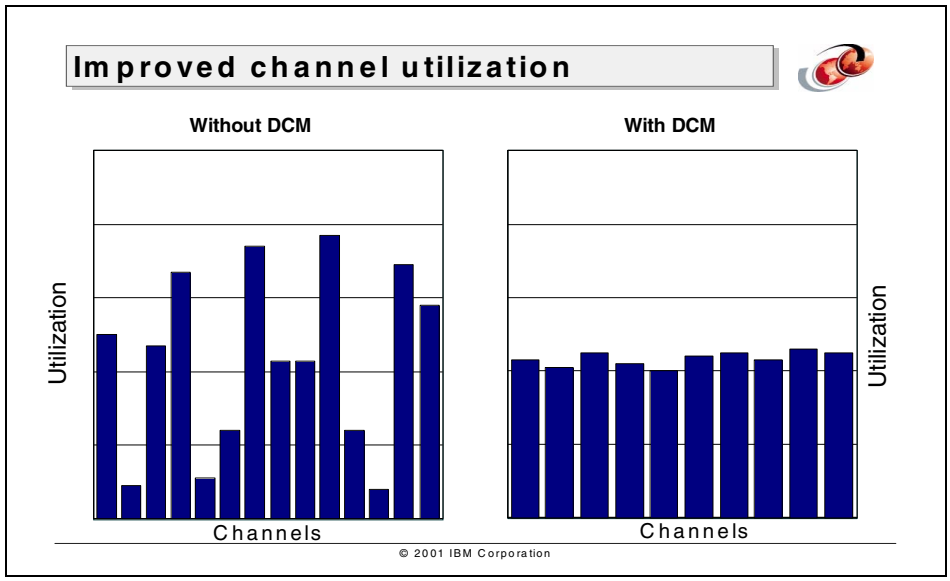
Using DCM, you are only required to define a minimum of one non-managed path and up to seven managed paths to each controller (although a realistic minimum of at least two non-managed paths are recommended), with Dynamic Channel-path Management taking responsibility for adding additional paths as required, and ensuring that the paths meet the objectives of availability and performance.

8.2.3 Reduced skills requirement



It is still necessary to understand the performance and availability characteristics of all your installed hardware. DCM can only work within the confines of the physical connectivity that you design and implement. However, to the extent that DCM is self-tuning, it should be possible to spend considerably less time on configuration management and monitoring and capacity planning. This allows you to free up scarce technical resources to work on other more valuable tasks.

8.2.4 Maximize utilization of installed resources

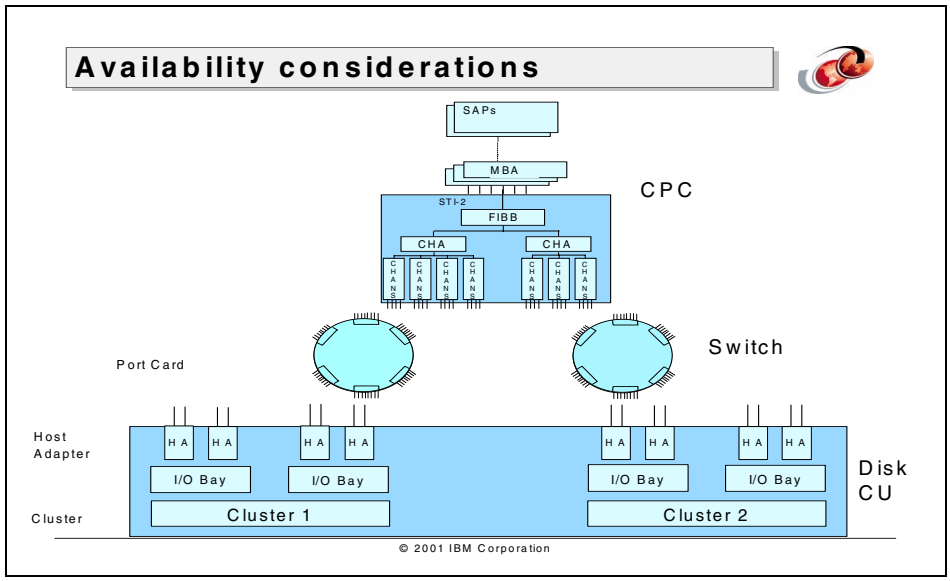


As one part of the continuing drive to reduce the cost of computing for the zSeries platform, DCM can help you drive more traffic to your DASD subsystems without necessarily having to invest in additional channels. DCM may let you increase the overall average utilization of your channels, without adversely impacting the response time for the connected subsystems.

For example, in the chart on the left, some channels are heavily utilized (resulting in channel contention for the attached control units), while others are nearly idle.

In the chart on the right, the overall average utilization is more consistent. This is because DCM will help a busy channel by adding an idle channel to the control units that it is attached to. This increases the utilization on the idle channel, and decreases the utilization on the busy one. The net effect is improved overall response times (because the peaks should not be so high), while driving more value from the installed channels (because idle ones will be put to work helping the busy ones). You will also notice that even though the peak utilization has been decreased, the number of channels in the DCM scenario has *decreased*, from 14 down to 10, thus allowing you to deliver similar service levels with fewer channels.

8.2.5 Enhanced DASD subsystem availability



When attaching a DASD subsystem to a zSeries CPC, there are a number of things you should take into account, if you wish to achieve maximum availability and performance from the configuration. Within the CPC, there are a number of redundant components provided (channel cards, STIs, MBAs, and so on). Ideally, you would attach a control unit to channels that are spread over as many of these components as possible, minimizing the number of single points of failure.

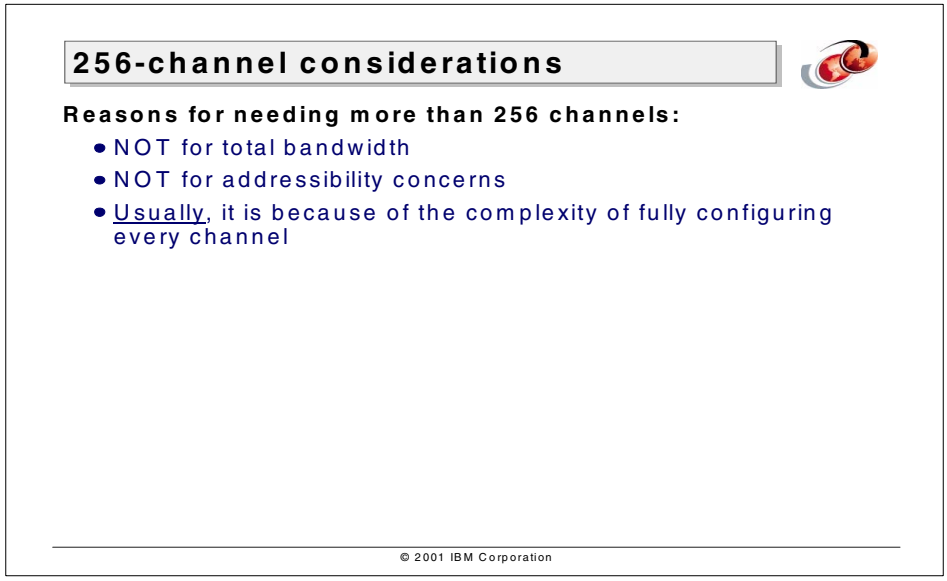
Similarly, in the switch and the DASD controller itself, there are various availability characteristics that you must be aware of when designing the physical connectivity.

If you currently go to the trouble of configuring every path to make the best use of the provided availability features, you are probably in the minority! DCM can help you make the best use of the installed availability features, without the time investment that was required prior to DCM. As we stated previously, you still have to take these hardware features into account when planning your physical connectivity, however, DCM can reduce the effort you have to put into deciding which paths should be used for each control unit, and which control units should be placed on the same paths. This reduces your workload and provides you with similar or better reliability and availability.

Another advantage of DCM is that it works all this out using support provided in the hardware components (CPCs, switches, control units), so every time you install a new device, you don't have to worry about reconfiguring your paths to match its characteristics. It should be sufficient to follow the connectivity recommendations provided in the installation documents associated with the device when you are designing the physical connectivity. IBM is working with non-IBM hardware vendors to ensure that they also provide information[2] in a format that DCM can use to identify potential single points of failure.

[2] The DASD vendor can provide a load module that contains this information.

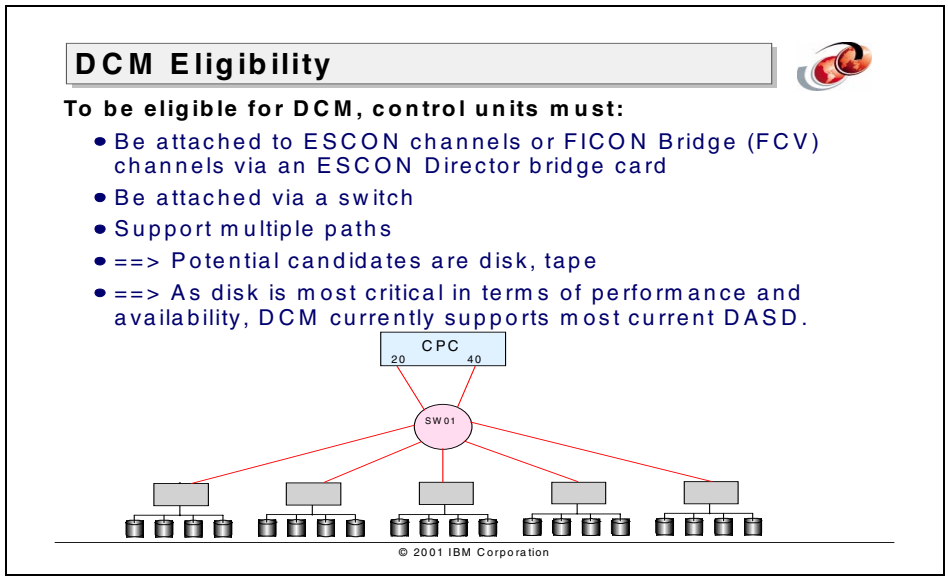
8.2.6 Reduced requirement for more than 256 channels



For most installations that require more than 256 channels, that requirement is rarely driven by a need for more than 256 channels worth of bandwidth—current processors do not have sufficient power to drive even 256 channels flat out, so providing more channels would not result in more work being done. Similarly, it is usually not driven by addressibillity considerations—for most installations, 256 channels provide connectivity for more devices than they will attach to a single CPC. In most cases, the requirement is driven by the complexity of configuring every channel with multiple control units, while also ensuring acceptable performance for every control unit.

DCM can address this last issue—configuring every channel with the maximum number of devices while still delivering acceptable performance across all the control units. Because DCM dynamically adjusts to changing workloads, if you happen to end up with multiple busy control units on a channel (a situation that can easily happen today and that will result in degraded performance), DCM will react to the resulting increased Pending time (more commonly known as *Pend time*) by adding idle or less busy channels to the control units that are suffering high Pend times.

8.3 Devices and channels that can be managed



DCM only supports DASD control units that operate completely non-synchronously[3] and are attached via ESCON or FICON Bridge (FCV) channels.

In addition, because DCM works by adding paths to a control unit, the control unit obviously has to support multiple paths. This effectively limits you to DASD and tape. Because DASD is typically more response time-critical, DCM only manages paths to DASD control units.

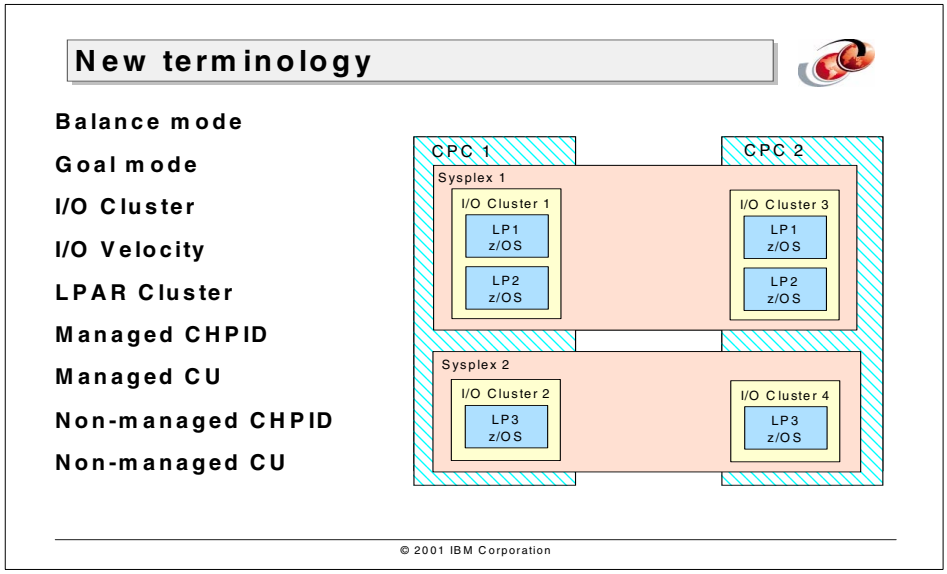
In order for DCM to be able to add or remove paths to a control unit, the control unit must be attached to a switch and that switch, in turn, must be attached to managed channels. Throughout this document, we use the generic term *switch* when referring to an ESCON Director.

For example, in the diagram above, the channel with CHPID 20 can have a path to any or all of the 5 control units. If the channel was not attached to the switch, it would be limited to attaching to a single control unit, and the path could not be “moved” if another control unit required additional bandwidth.

The supported and unsupported devices are discussed in more detail in 10.1, “Hardware planning” on page 258.

[3] A non-synchronous I/O operation is one where the control unit does not remain connected to the channel during transfers to the DASD device. All transfers to the channel are done from the control unit cache rather than from the device.

8.4 New terminology for DCM



There are some new terms that we use in this part of the book, specifically relating to Dynamic Channel-path Management. These terms, and their meanings, are:

I/O Velocity Measure of the responsiveness of a control unit. Defined as being the time the control unit is being used productively, divided by the time the control unit is being used productively plus channel contention delays.

DCM Balance mode Mode in which DCM attempts to balance I/O velocity across all managed DASD subsystems in the LPAR Cluster, irrespective of the importance of the work using a given Logical Control Unit (LCU).

DCM Goal mode Mode in which DCM takes service class periods objectives into account when deciding if a given LCU should have an explicit target I/O velocity set for it.

I/O Cluster An I/O Cluster is the group of z/OS systems, running in z/Architecture mode on a single IBM zSeries 900 CPC that are all in the same sysplex. It is possible to have more than one I/O Cluster per CPC if there are members of more than one sysplex on that CPC. At the time of writing, the term I/O Cluster is synonymous with LPAR Cluster.

LPAR Cluster The set of one or more LPs, running on one IBM zSeries 900 or later CPC, that are in the same sysplex *and* are running z/OS 1.1 or later in z/Architecture mode. The scope of an LPAR Cluster is currently the same as the scope of an I/O Cluster, however the term LPAR Cluster is also used in relation to WLM LPAR CPU Management and Channel Subsystem I/O Priority Queueing.

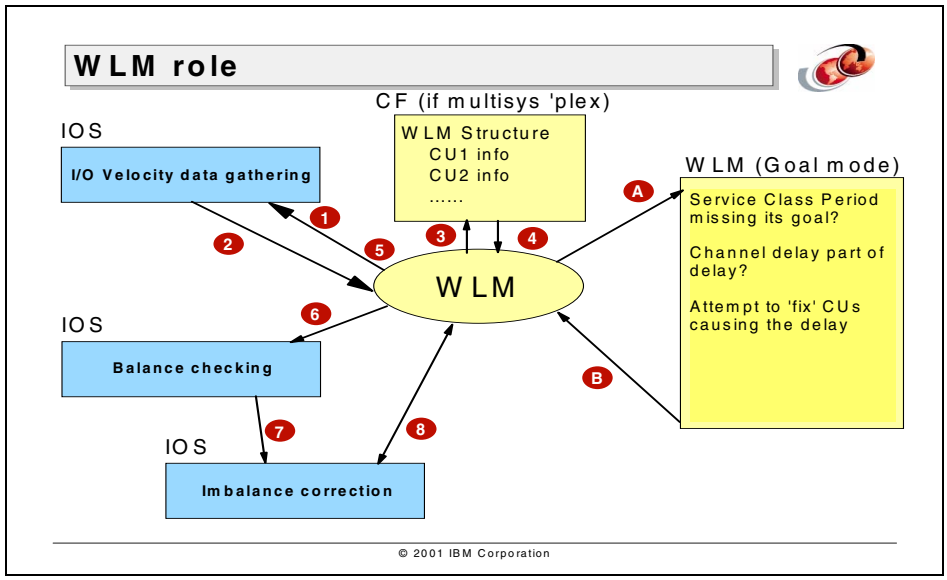
Managed Channel Any channel that is specified in HCD as being managed by DCM. That is, the channel is defined as being managed and no control units are explicitly defined as being attached to that channel. It must be connected to a switch.

Managed CU Any control unit that is specified in HCD as being managed by DCM. That is, some of the CHPIDs that the control unit is attached to are defined in HCD using an asterisk (*) rather than a specific CHPID number. '*' indicates that that path is to use a managed channel.

Non-managed Channel New term for traditional channels that are not managed by DCM. Non-managed channels are sometimes also referred to as static channels.

Non-managed CU New term for control units that are defined in HCD as having no managed channel defined to it. Non-managed control units are also sometimes referred to as static control units.

8.5 WLM role in Dynamic Channel-path Management



The z/OS Workload Manager (WLM) plays an active role in Dynamic Channel-path Management, regardless of whether the system is in WLM Goal mode or WLM Compatibility mode. The diagram above shows the role of WLM in both modes.

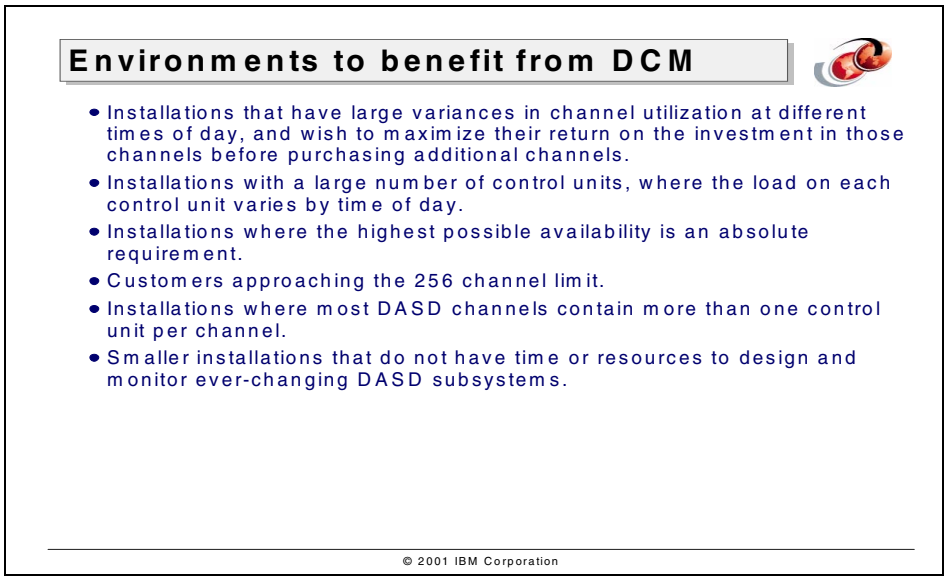
WLM is responsible for initiating the process of gathering performance information for all DASD subsystems (Step 1). Together with IOS, WLM calculates a metric, known as the *I/O velocity* (a value that is used to objectively measure the impact of channel contention), for every DASD subsystem (Steps 2 and 5). After this information has been gathered for all DASD subsystems, and optionally kept in a CF structure (Steps 3 and 4), IOS identifies the DASD subsystems whose channel delays fall outside an acceptable range - this process is known as balance checking (Step 6). Balance checking is followed by imbalance correction (Step 7), during which actions are taken to bring the identified DASD subsystems back towards a target I/O velocity. In addition, if DCM is operating in Goal mode, WLM may decide to set explicit velocities for control units that are impacting important workloads. This is shown in Steps A and B.

The I/O velocity is conceptually similar to the RMF Monitor III workflow percentage and the WLM Goal mode execution velocity in that it measures the impact of channel contention on the total I/O response time. This metric is the trigger that causes DCM to initiate a configuration change. I/O Velocity is discussed in more detail in 9.5, “I/O Velocity” on page 232.

If the system is part of a multisystem sysplex and is on a CPC running in LPAR mode, then any decisions relating to configuration changes by DCM must take into account the use of the channels and control units by *all* the systems in the LPAR Cluster (not in the whole sysplex). WLM uses a CF structure as the mechanism by which this information is shared between all those systems. This is shown as steps 3 and 4 in the diagram on the previous page.

Another function available when DCM is operating in DCM Goal mode is that if DCM needs to make a decision about a control unit that may affect another control unit with an explicit velocity, IOS will call WLM to decide which control unit is the more important. This is shown in Step 8 in the diagram.

8.6 Environments most likely to benefit



Based on the experiences of the ESP accounts and the design intent of Dynamic Channel-path Management, the environments that are most likely to benefit from DCM are those that meet one or more of the following profiles:

- Installations that have large variances in channel utilization at different times of day, and wish to maximize their return on the investment in those channels before purchasing additional channels.
- Installations where the load on different control units varies significantly by time of day.
- Installations where the highest possible availability is an absolute requirement. This is dependant on a well-configured underlying channel infrastructure—DCM cannot help your availability if all the channels attaching a control unit are connected through a single ESCON Director and a single channel card!
- Customers approaching the 256 channel limit per CPC.
- Installations with a large number of control units, and multiple control units per channel.
- Smaller installations that do not have sufficient technical resources to design and monitor ever-changing DASD subsystems.

If you are in the fortunate position of already having the maximum number of channels per control unit, and each channel only has one control unit on it, then there is not much that DCM can do to improve your current environment, certainly not from a performance point of view. Depending on how careful you were about avoiding single points of failure when setting up the configuration, it is possible that DCM may help you improve the availability of the configuration.

9

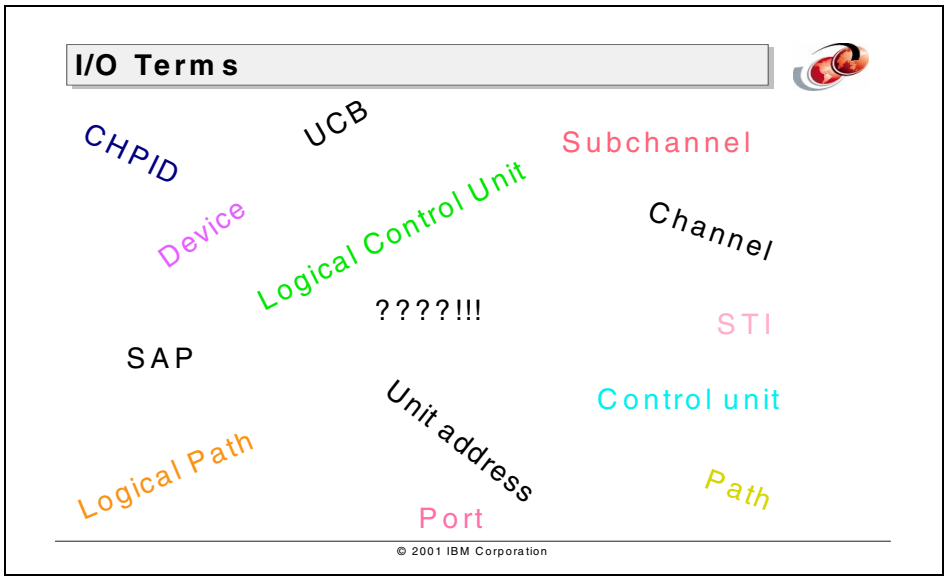
How Dynamic Channel-path Management works

So what *exactly* does Dynamic Channel-path Management (DCM) do, and how does it do it? Very briefly, it automatically adds and removes I/O paths to DASD control units in an attempt to provide the best performance and most effective resource utilization from the available hardware and configuration. It also coordinates these changes across the systems in the LPAR Cluster that are sharing the paths that are eligible to be reconfigured.

In this chapter, we describe how DCM does this. We start off by doing a level-set of how things worked prior to DCM, to provide a common level of understanding of the basics before we go on to describe the functions of DCM.

We explain how DCM knows which paths it can manage and what resources it has to work with. We also explain how it identifies which control units are in need of additional channel capacity and how it decides on and implements the changes required to provide that help.

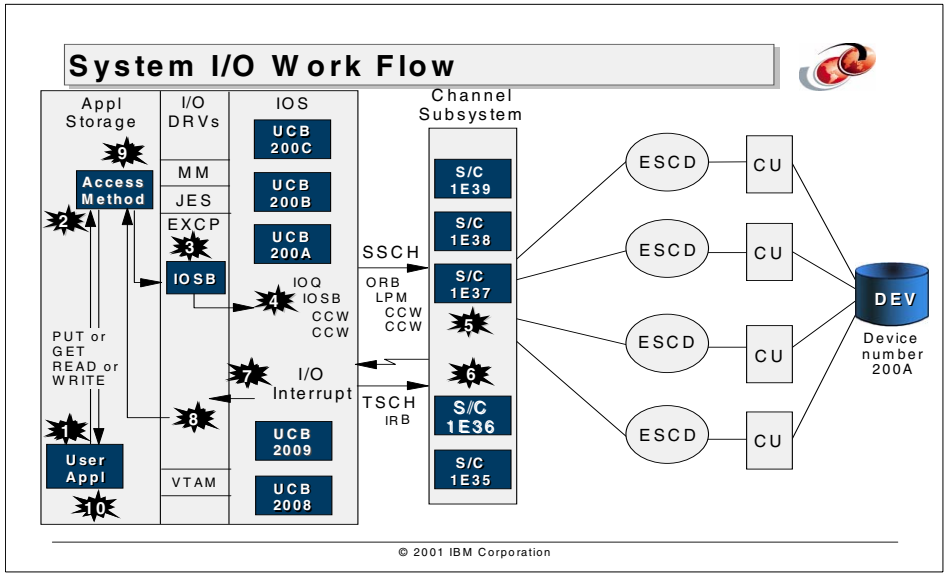
9.1 Understanding the basics



A plethora of terms is used when discussing I/O processing. As all of these might not be familiar to people new to the platform, we provide a description of each, and how they relate to each other. If you are already familiar with all these concepts, you can skip to 9.2, “Configuration definition prior to DCM” on page 216.

Note: As Dynamic Channel-path Management only applies to DASD subsystems at this time, we only discuss how DASD subsystem configurations work.

9.1.1 Life of an I/O



The diagram above shows the flow of an I/O operation from the request by an application until the operation completes. It introduces terms that will be explained in the next few pages. The steps in the flow are:

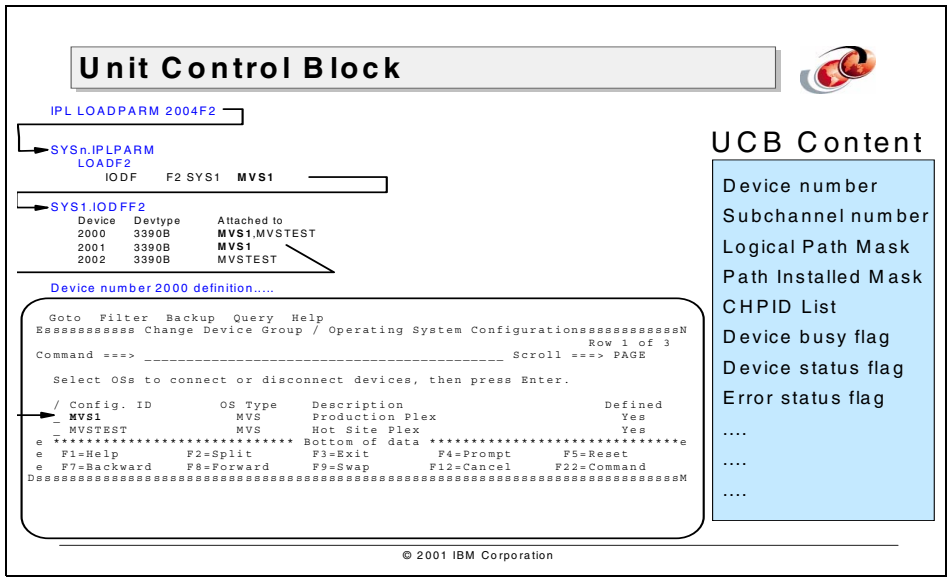
1. The user program connects to a data set by allocating it. The OPEN macro instruction causes the users access authority to be granted and the data set to be allocated to this program (assuming it has authority). The program uses an I/O macro instruction like GET, PUT, READ, or WRITE. The I/O macro instruction invokes an access method (through a branch instruction).
2. An access method has the following functions:
   - It creates the channel program (with virtual addresses) knowing the physical organization of the data.
   - It implements buffering.
   - It guarantees synchronization.
   - It automatically redrives an I/O request following a failure.

   The user program could bypass the access method, but it would then need to consider many details of the I/O operation, such as the physical characteristics of the device. There are several z/OS access methods (VSAM, BPAM, QSAM, and so on), each of which offers different functions to the user program. The access method that will be used depends on how the program plans to access the data (randomly or sequentially, for example).

3. To request the movement of data, the access method presents information about the I/O operation to an I/O driver routine (usually the EXCP driver) by issuing the EXCP macro instruction, which expands into an SVC instruction. The I/O driver, now in supervisor mode, translates the virtual storage addresses in the channel program to real storage addresses (a format acceptable to the channel subsystem), fixes the pages containing the channel programs and the data buffers, guarantees the data set extents, and invokes the I/O Supervisor (IOS). Media Manager (represented by an MM in the diagram) is the I/O driver of the VSAM access method.
4. If there are no pending I/O operations for the required device (from this system), IOS issues the Start Subchannel (SSCH) instruction to the channel subsystem. At this point, the task that initiated the I/O is generally placed in a wait state, and the CPU continues processing other tasks. When the channel subsystem indicates, with an I/O interrupt, that the I/O operation has completed, the task will be dispatched again. If there is already an outstanding I/O request to the device from this system, the request is queued on the UCB control block.
5. One of the System Assist Processors (SAP) processes the SSCH instruction and selects a channel to initiate the I/O operation. This channel executes the channel program, controlling the movement of data between device, control unit, and processor storage. Because a control unit does not have to reconnect to the same channel, it is possible that the I/O request will be completed by a different channel than the one the request was started on. There are more details available in 9.1.3, “Channel subsystem logic” on page 194.
6. When the channel signals that the I/O operation has completed, a SAP signals the completion to a CP by generating an I/O interrupt.
7. IOS processes the interruption by determining the status of the I/O operation (successful or otherwise) from the channel subsystem. IOS indicates that the I/O is complete by posting the waiting task and calling the dispatcher.
8. When appropriate, the dispatcher dispatches the task, returning control to the access method code.
9. The access method returns control to the user program (through a branch).
10. The user program then continues its processing.

Having shown all the steps in the I/O, we now discuss some of these, and other related terms, in more detail.

9.1.2 Unit Control Block



z/OS (and OS/390 before it) uses a Unit Control Block (UCB) to represent a device and its state. Systems programmers use HCD to define I/O devices in an I/O Definition File (IODF), and to connect the devices to an operating system configuration. During the IPL, the IODF is read and one UCB is built in SQA for each I/O device definition found in the IODF. The load parms used when the operating system is IPLed control which IODF is used.

If a device physically exists but is not defined in the IODF, then a UCB is not built and it is not possible for applications to access the device. Subsequent to the IPL, devices can also be dynamically defined, changed, or deleted using the OS/390 or HCD ACTIVATE command.

The UCB contains all the information necessary for z/OS to use the device for performing I/O requests, and records the last known status of the physical I/O device. The UCB describes the characteristics of a device to the operating system. The UCB is used by the I/O supervisor (IOS) when performing I/O requests, and by the job scheduler during data set allocation.

A subset of the information in the UCB can be displayed using the UCB option of the DEVSERV QDASD operator command.

Because traditional devices do not allow more than one I/O at time, IOS uses the UCB as a queue anchor block. If the device is already executing a previous I/O operation initiated by this system, a new I/O request is queued in the UCB. The time spent in this queue is called IOS queue time (IOSQ). IOSQ time is one component of the response time of an I/O, together with Pend time, Connect time, and Disconnect time.

As for any queue, the IOS queue time is a consequence of the service time of requests that arrived before you. Because DCM potentially decreases the response time (by lowering the channel busy and director port busy components of Pend time) of each request, each request completes faster, meaning that subsequent requests have to spend less time queuing. As a result, DCM can indirectly improve the IOS queue time. Another example of a way to decrease IOSQ time is to use the IBM 2105, which provides the ability to do multiple requests in parallel to a device behind the 2105 controller. The other components of I/O response time are discussed in 9.1.3, “Channel subsystem logic” on page 194. The impact of Pend time is discussed in 14.5, “Reasons for Channel Subsystem I/O Priority Queueing” on page 368.

9.1.3 Channel subsystem logic

Explaining the logic of the channel subsystem helps you understand the basic role of DCM. For completeness, we also include the logic of the SSCH instruction and the I/O interrupt.

Channel subsystem

The zSeries channel subsystem contains:

- One or more special processor units (PU) called SAPs. A SAP runs special I/O Licensed Internal Code (LIC). The SAP takes responsibility for some of the processing during the execution of an I/O operation, freeing up the operating system CPs to do other work. It schedules an I/O operation, checking for the full availability of the I/O path, and provides a queue mechanism if an I/O path is not available.
- Channels, which are special processors able to communicate with I/O control units (CUs) and manage the movement of data between processor storage and these control units.

  Specifically, the channels can:

  – Send channel commands from the processor to a CU via electrical or optical signals
  – Transfer data during read and write operations
  – Receive status at the end of operations

– Receive sense information from control units

An I/O operation starts when a Start Subchannel (SSCH) instruction is executed by IOS, which issues the instruction on behalf of a z/OS dispatchable unit (TCB or SRB). It ends when an I/O interrupt is received by the CPU (forcing the execution of IOS code again).

Start Subchannel (SSCH) logic

Start Subchannel (SSCH) is a privileged instruction issued by IOS to start an I/O operation. The SSCH instruction has two operands:

- Subchannel number, which is an index to the subchannel associated with the I/O device where the I/O operation will be executed. Refer to 9.1.9, "Subchannel number" on page 211 for more information about subchannels.
- Operation Request Block (ORB) address, which contains information about *what to do* during the I/O operation; among other fields it contains the channel program address.

The SSCH moves the ORB contents into the respective subchannel and places the subchannel in a specific Hardware System Area (HSA) queue named the *initiative queue*.

SAP logic

Depending on the processor model, there will be two or more SAPs in a channel subsystem. The SAP finds the subchannel in the initiative queue and tries to find a channel that succeeds in *initial selection* (connects to a control unit and starts the I/O operation). The SAP uses information in the subchannel to determine which channels and control units can be used to reach the target device.

Initial selection may be delayed if:

- One or more channels (serving the device) are busy.
- The Switch port connecting the control unit is busy.
- The control unit is busy.
- The device is busy, due to activity from another system.

All these delays are included in the I/O Pend time. During all of these delays, the request is serviced by a SAP without z/OS having to get involved. When the I/O operation finishes the SAP queues the subchannel (containing all the I/O operation final status information) in the I/O interrupt queue.

Once the channel is successfully able to start the I/O, the remaining time until the I/O is complete is called I/O Service_Time. This consists of two components:

I/O_Service Time = I/O_Connect Time + I/O_Disconnect Time

I/O_Connect time includes the data transfer and control protocol times between the channel and the control unit. As a result, if you can transfer the same amount of data with fewer SSCHs, thus reducing the amount of protocol time, you may see a significant decrease in the total Connect time to transfer that data. One way to decrease the number of SSCHs is to use larger blocksizes, thereby transferring more data in each I/O.

I/O_Disconnect time consists of all the time the channel is not involved during the I/O operation. Traditionally, this time is made up of:

- Processing a cache miss (having to access the DASD device)
- Doing the I/O to the secondary control unit of a remote copy pair, if using synchronous remote copy (Peer-to-Peer Remote Copy (PPRC), for example)
- Waiting for an available channel after the control unit has unsuccessfully tried to reconnect

With the more recent control units, Disconnect time may also include the following:

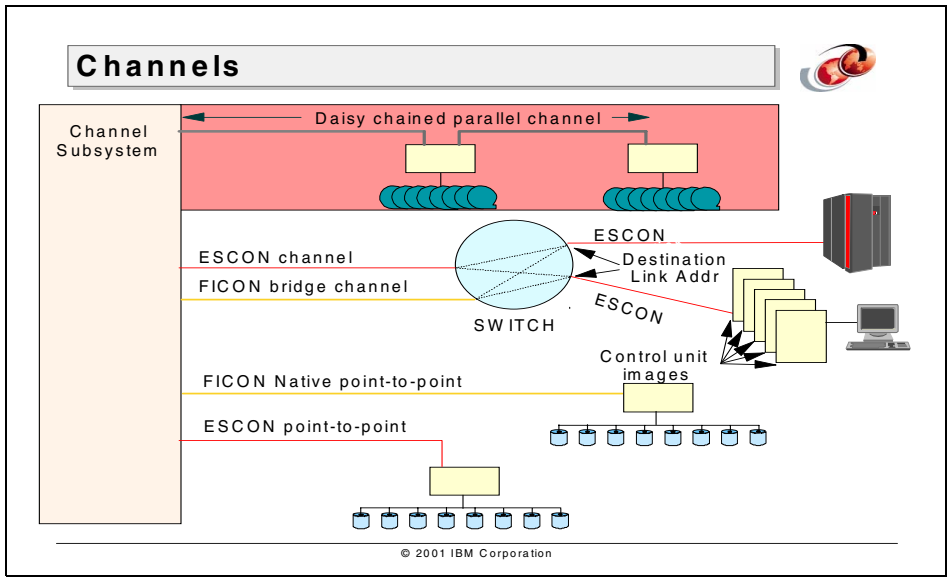
- Time when all the control unit internal data paths are busy and the control unit is unable to process the I/O request coming from the channel. The request is accepted, internally queued and the channel is disconnected. Previously, this amount of time was reported as control unit busy, and included in I/O Pend time.
- Time when the device is busy due to I/O from another image (shared DASD busy). The request is accepted, queued within the control unit and the channel is disconnected. Previously this time was reported as device busy delay, and included in I/O Pend time.

Interrupt processing

zSeries I/O processing is an interrupt-driven rather than a polling architecture. The only exception to this rule is the handling of the z/OS and Coupling Facility communication, where polling is used. When a CPU is enabled for I/O interrupts, and it detects a UCW in the interrupt queue, the I/O interrupt is accepted and control is passed to IOS. An Interrupt Response Block (IRB) describing the final status of the I/O operation is moved to storage. Another way to receive this interrupt is by IOS synchronously issuing the test pending interrupt (TPI) instruction (this is reported in the RMF CPU Activity Report). IOS will normally then post the z/OS process waiting for the I/O operation.

In certain error situations, when the I/O interrupt is not generated within an expected time frame, the Missing Interrupt Handler (MIH), a timer-driven routine, alerts IOS about this condition.

9.1.4 Channels



There is some confusion and ambiguity over the use of the terms *channel* and *CHPID*.

A *channel* is the piece of hardware with logic in the CPC to which you connect a cable in order to communicate with some outboard device.

There are many different types of channels. From the point of view of channels that can connect a DASD subsystem, you have a choice of:

- Parallel (copper) channels

  These channels were available going back to the S/360 generation of processors, and have a maximum channel speed of 4.5 MB per second. As can be seen in the figure above, parallel channels provide the ability to connect in a “daisy-chain” topology to a number of control units on a single channel.

- ESCON channels

  ESCON channels were introduced in 1990 on the 3090 J series of processors, and can run at speeds up to 17 MB per second utilizing fiber cables. Unlike parallel channels, ESCON channels are point to point topology (each channel connects to just one control unit); however, they can be used with a Switch (the ESCON Director) to give a single ESCON channel access to more than one physical control unit. ESCON channels can be shared between LPs in the same CPC using the Multiple-Image Facility (MIF).

- FICON Bridge (FCV) channels

  These were introduced with the 9672 G6 CPCs, and are available on the 9672 G5 and G6 and IBM zSeries 900 CPCs. FICON Bridge channels protect the investment in control units with ESCON adapters by allowing those control units to be used with the newer FICON channels. FICON Bridge channels are connected to the DASD controllers through the ESCON Director, using special bridge cards in the director. FICON Bridge channels provide higher bandwidth and greater distances than ESCON channels, and support up to 16K devices per channel compared to the ESCON implementation limitation of 1024. FICON Bridge (FCV) channels can be shared between LPs in the same CPC using MIF.

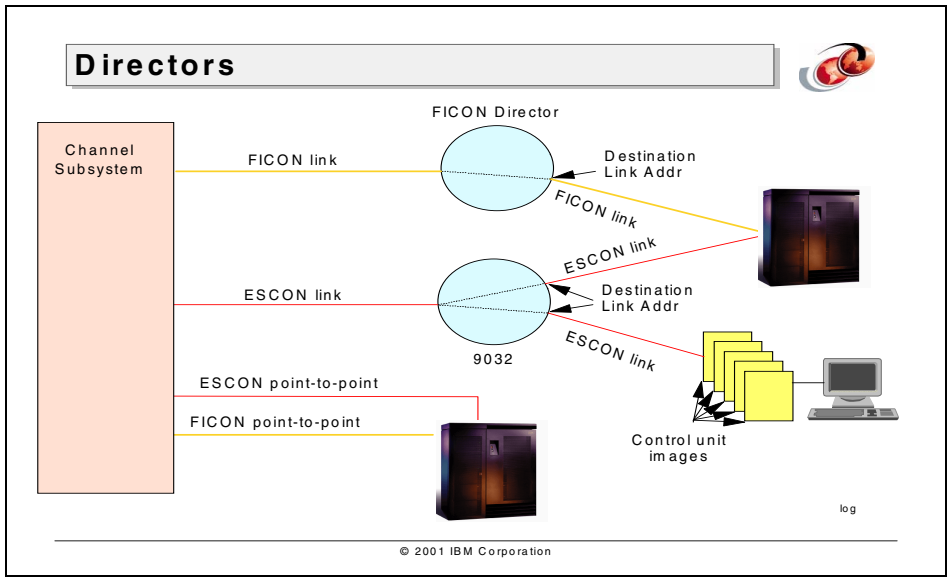
- FICON Native channels

  These are available on the 9672 G5 and G6 and IBM zSeries 900 CPCs. FICON Native channels are also point to point, and can connect either directly to a FICON port on a single control unit, or to multiple control units via a FICON director. FICON Native channels also support a maximum of 16K devices per channel. FICON Native channels can be shared between LPs using MIF.

Channel Path IDentifiers (CHPIDs) are the 1-byte *identifiers* for channels. You use a CHPID to identify a channel to the hardware and software in the HCD. The maximum number of CHPIDs in an IBM zSeries 900 CPC is 256. Even though the two terms are often used interchangeably, you should really talk about attaching a control unit to a *channel,* and using the *CHPID* in an z/OS CONFIG command to identify which *channel* you wish to bring online or offline.

In relation to Dynamic Channel-path Management, there are two new terms relating to channels. A channel can now be described as being *non-managed* (sometimes also referred to as *static)*. This is the type of channel we are all used to, where the channel's CHPID number appears on at least one control unit definition, meaning that that channel is always used to access that control unit, (at least until a configuration change is made). The other new term is *managed* channel. A managed channel is one that is defined with MANAGED=YES in HCD, has an *I/O Cluster* name associated with it (if the CPC is in LPAR mode), and its CHPID number does not appear on any control unit definitions. An I/O Cluster is effectively the same as an LPAR Cluster - the set of one or more z/OS LPs in z/Architecture mode in a z900 CPC that are in the same sysplex. Managed channels are dynamically added to, or removed from, a control unit by Dynamic Channel-path Management.

9.1.5 Directors



A *Director*, by which we mean an ESCON Director or a FICON director, provides a means of attaching more than one physical control unit to a single channel, and also allows you to effectively attach more than one channel to a single control unit port. It is also possible to connect a channel to another channel (as in a CTC) or a control unit to another control unit, as in a remote copy scenario.

ESCON Directors and FICON directors are high-speed switching devices that can dynamically connect two ports. Typically, a CPC channel would be attached to one port, and a control unit would be attached to the other. The ports can be connected on a permanent basis, known as a static connection, in which case those ports cannot communicate with any other port. Of more interest, in the context of this book, are the connections that are dynamic, providing the ability for a single host channel to communicate with more than one physical control unit and vice versa.

When the director is defined in HCD, it is given a 2-byte ID. This ID is used when you tell HCD which director a given channel or control unit is attached to. However, it is not used outside of HCD.

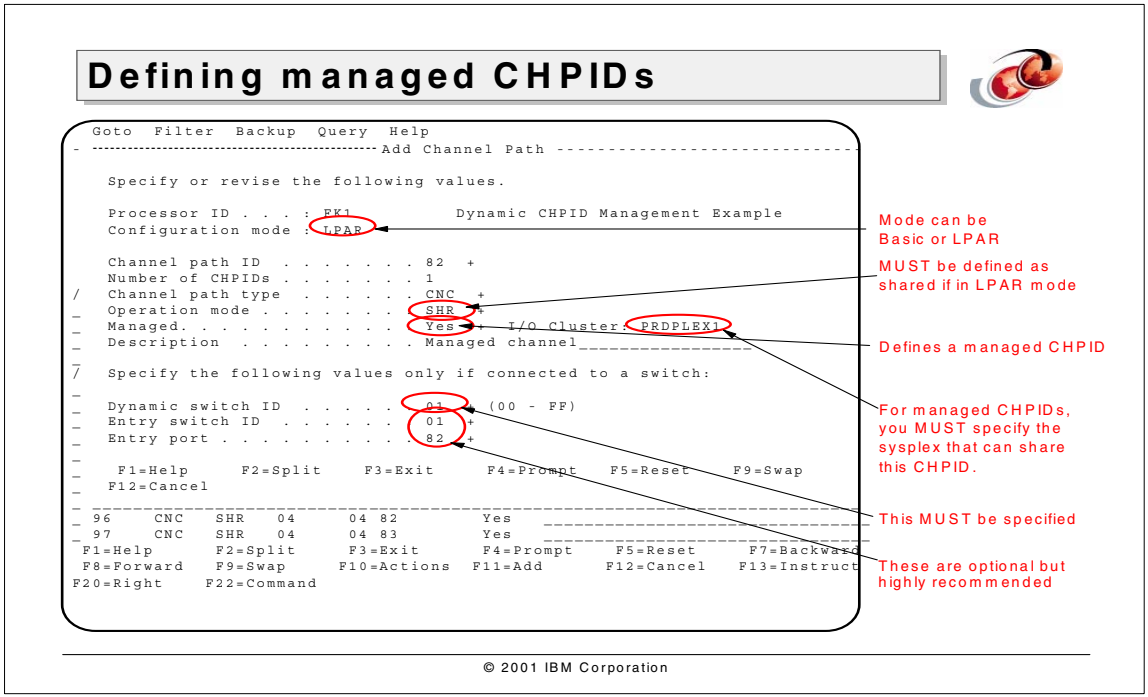
When you use HCD to define a control unit that is connected via a director, you must tell HCD, for each path on the control unit, which channel, which director (or “Switch”, in HCD terminology), and which port (link) on the director the control unit is attached to. This information is then used by the channel to build the address by which it communicates with the device (LINK.CUADD.UA). *Link*

means the port address on the director that the control unit is connected to. CUADD and UA are discussed in 9.1.7, “Unit address” on page 207. It is not mandatory to specify the port that the channel is connected to. The channel obtains that information at initialization in a dialogue with the director. However, in order to maximize the value of HCD as a repository of physical and logical connectivity, we recommend specifying this information when you define the channel.

On ESCON Directors and FICON directors, there is a reserved port called the Control Unit Port (CUP). It is used by the operating system, and the I/O Operations (previously called ESCON Manager) component of System Automation for OS/390 (SA/390), if it is installed, to communicate with the control unit function in the director. This capability is used by the director to report errors to the host, and also by the host to gather information about the director itself and about what elements (control units, CPC channels, or other directors) are attached to it. It is also possible to change the status of a port using commands issued from SA/390 to the CUP. While it was always *recommended* to define the ESCON Director CUP as a device in HCD, it was not *required*. However, for Dynamic Channel-path Management, it *is* a prerequisite that you do so. The device number that you assign to the CUP is how you will communicate with the director using z/OS console commands. See 9.4, “Initialization changes” on page 227 and 10.1.4, “Switch considerations” on page 267 for more information.

The device number of the CUP of the director that a channel is attached to is displayed when you issue a D M=CHP(xx) command in OS/390 V2R10 or later for that channel. Information about the director configuration can be displayed using the new D M=SWITCH command, and the DCM status of eligible ports can be altered with the new VARY SWITCH command. These are discussed in more detail in 12.1, “New operator commands” on page 318.

It is possible that two host channels could attempt to communicate with a control unit interface using the same director port at the same time. This results in a situation known as *Director Port Busy.* This time is reported as part of Pend time, and is one of the delays addressed by Dynamic Channel-path Management. Dynamic Channel-path Management also uses its knowledge of the Switch topology as one of the criteria when deciding which paths to add to or remove from a control unit.



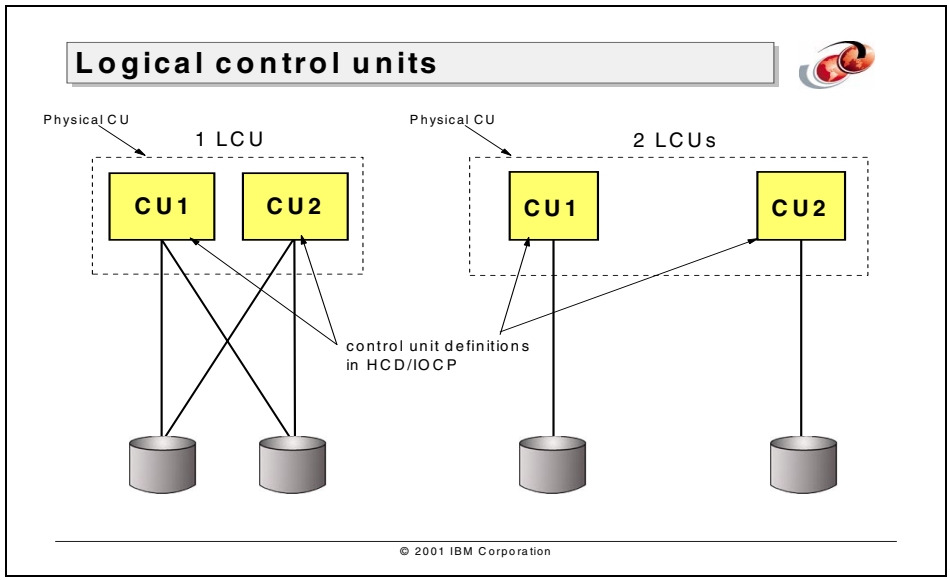
It is possible to have two ESCON directors in the path between a CPC and a control unit. If this is done, one of the directors must have a static connection between the entry port and the port the control unit (or the second director) is connected to. The following terms are used in the HCD application when you are defining a channel (as in the figure above):

Entry Switch ID: This is the identifier of the director to which the channel is physically attached. In the case of two chained switches, it is the first switch on the data path from the channel to the devices. The entry switch ID can be the same as the dynamic switch ID if the entry switch allows dynamic connections, or if it is the only switch in the path.

Dynamic Switch ID: This is the ID of the director that has the dynamic connection, if there are two directors in the path to the control unit. If there is only a single director, this is the ID of that director. If both Dynamic and Entry directors are being defined, the Dynamic director must have been previously defined to HCD before you specify its ID in this field. This field is required for dynamic CNC and CTC connections.

Entry Port: This identifies the port of the entry switch to which the channel is physically attached. This information is optional, however it is highly recommended as it allows tools like HCM to create a complete map of the I/O configuration.

9.1.6 Control units



There has been some confusion between the terms controller, control unit, logical control unit (LCU), and physical control unit. In this book, we use these terms as follows:

- *Control unit (CU)*. In all cases, an I/O-device operation is regulated by a control unit that provides the functions necessary to operate the associated I/O device. The control unit is also responsible for interfacing with the channel, talking to it in its specific protocol (parallel, ESCON, or FICON). The control-unit function may be housed with the I/O device or a separate control unit may be used. The term *control unit* on its own is used very loosely. Depending on the context, it can refer to either a physical control unit, a logical control unit, or a logical control unit image. To avoid confusion, from now on, in this book we avoid the use of the term *control unit* on its own, unless it is clear from the context which type we are referring to.
- A *physical control unit* (also called a controller or DASD subsystem) is the piece of hardware that sits on your computer room floor. Originally, this was actually a separate unit, which could be purchased independently of the devices that were attached to it. An example would be an IBM 3880 or 3990. More recent devices have integrated the control unit function with the attached devices into a single unit. For example, an IBM 2105 Enterprise Storage Server cannot be ordered as just a physical control unit.
- *Logical Control Unit (LCU)*. Over the years, there has been some confusion about the term LCU—depending on if you spoke to a hardware or a software specialist, you would get a slightly different definition. This was largely due to

the fact that the physical control unit was separate from the DASD devices, giving you some flexibility in relation to how you attached the DASD to the control units. Thankfully, a by-product of the move to integrated physical control units is that an LCU effectively now means the same thing to both hardware and software people.

An LCU is defined in the IOCP Users Guide as: “a channel subsystem concept representing a set of control units that attach common I/O devices”. When you define an I/O configuration using either HCD or IOCP, you define channels, control units, and I/O devices. When you define a device, you specify to which CU or CUs the I/O device is attached. When you do a Power on Reset (POR), LCUs are built in HSA, providing an anchor point to queue I/Os to one of the devices in that LCU.

The relation between the control unit numbers that you assign in HCD, and the logical control unit numbers as reported in RMF, depends on the physical controller internal design. In the 3390/3990, we had two physical control units sharing the same set of devices, so you had only one LCU, but in HCD it was defined as two control units: this is seen in the following RMF I/O Queueing Activity Report, where LCU 16A has two control units (200A and 200B) associated with it.

```
                    RMF - I/O Queuing Activity


15:02:41         I=  8%                CONT  DEL Q   %ALL   CHPID   %DP
CHAN PATHS CONTROL UNITS         LCU   RATE  LNGTH  CH BUSY TAKEN   BUSY

2F         200A                  016A   0.0   0.00   0.00    0.99   2.47
32         200A                  016A                         0.86   0.72
33         200B                  016A                         0.85   0.73
35         200B                  016A                         0.89   1.37
33         400A                  016D   0.0   0.00   0.00    1.69   0.00
35         400B                  016D                         1.69   0.00
```

Because modern DASD subsystems (such as the IBM 2105) usually contain more than 256 devices, the physical control unit in these subsystems actually consists of multiple Logical CU images. For example, the IBM 2105 contains 8 or 16 Logical CU images, each one addressing 256 devices. Each of these Logical CU images is defined in HCD as a separate control unit. The individual Logical CU images are identified using the CUADD parameter. Because each image has its unique set of devices, the channel subsystem creates either 8 or 16 LCUs per physical 2105 ESS (in ESS terminology, LCUs are called LSSs). The topic of control unit images is discussed further in 9.1.7, “Unit address” on page 207.

What exactly does CUADD represent? The ESCON I/O architecture allows control units to implement multiple images that can be addressed separately (logically, several control units are packaged in one physical control unit). The CUADD parameter is used to indicate the logical control unit image that contains the requested device. Modern DASD controllers use CUADD to increase the number of devices that can be accessed by channels (for example a FICON channel can reach 16K devices). To implement this, each physical control unit looks like several *control unit images,* each one supporting up to 256 devices. The unit address points to one of these devices (no bits indicating the specific logical CU) and the CUADD indicates the specific control unit image within the physical CU controlling the required device.

Strictly speaking, a logical CU image is not the same as an LCU, even though some IBM publications erroneously call these logical CU images logical control units. However, they are different. A logical CU image is a bypass to circumvent the architectural restriction on the maximum number of devices in a controller. Logical Control Unit is a mechanism to implement queueing in the SAP. However, in practice, in modern DASD subsystems there is a one-to-one correspondence between the two. When a fiber channel-attached control unit uses control unit image addressing (CUADD), each of these control unit images in the physical control unit would be represented by one or more control unit definitions in HCD, and each control unit image would correspond to an LCU. Each control unit image is used to access a different set of devices. For example, the IBM 2105 Enterprise Storage Server's 8 or 16 control unit images (LSS's in 2105 terminology) are potentially accessible from any of the host adapters.

An LCU cannot have more than 8 paths to a single CPC[1], a restriction enforced by HCD. If you attempt to "cheat" by defining a second LCU to access the same devices from the same physical control unit, you would have to specify different device numbers—and OS/390 would issue a message stating that you have duplicate volsers. As you told z/OS that the device numbers are different, it assumes that this is actually two different devices, both containing a volume with the same volume serial number, and therefore issues the message about duplicate volsers. It *is* valid to define the LCU and devices twice (the device numbers could be the same, but the control unit numbers must be different), but in this case the second LCU cannot share *any* paths with the first LCU, and the second LCU cannot be attached to any of the same LPs as the first LCU.

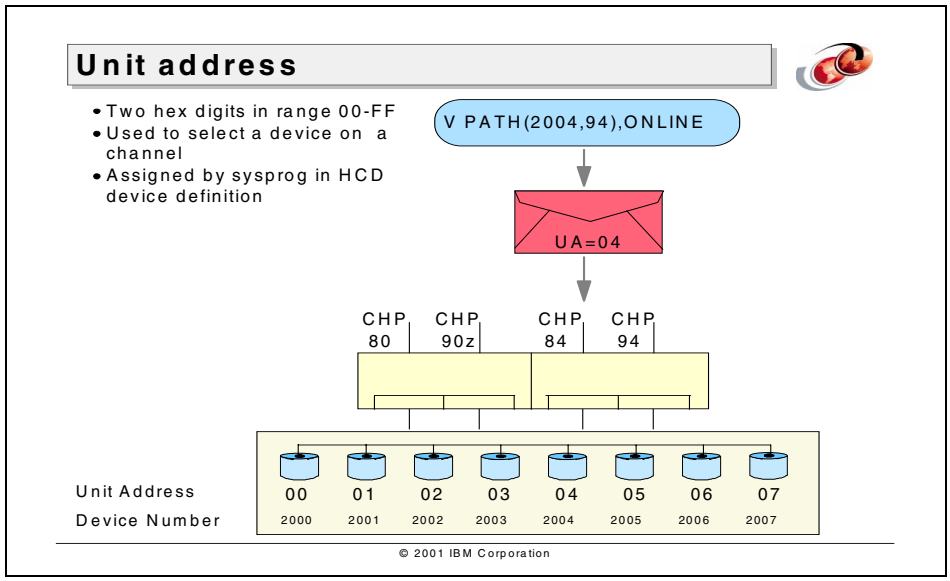
[1] It is possible, using a feature known as duplicate device number support, to have up to 8 paths per LP. This is discussed further in 10.4, "MIF considerations" on page 283.

LCU numbers get assigned sequentially, starting at 0000, when you run IOCP to create the IOCDS. If you produce an RMF Shared Direct Access Device report, you see the LCU number that is used by each system that is included in the report. You may also see that each CPC has assigned a different LCU number to the LCU; however, all the LPs on a CPC that share the LCU will use the same LCU number when referring to that LCU because all of them are described within the same IOCDS configuration.

z/OS does not provide a command to display information specifically about a control unit (either physical or logical). However, the DEVSERV PATHS command will provide some information about the characteristics of the physical control unit the specified device is attached to. The serial number of the physical control unit that a device is attached to, as well as the control unit type, can be displayed using the DEVSERV QDASD command. There was also an enhancement to the D M=DEV command in OS/390 V2R10 that provides the node descriptor of the physical control unit the device is attached to.

On modern DASD subsystems, it is unusual to see any amount of control unit busy time because the number of internal path tends to be equal to the number of host adapters. However, if a control unit is genuinely busy, Dynamic Channel-path Management cannot address that situation.

9.1.7 Unit address



An I/O device has at least three different ways of being named, depending on who is talking about it. They are: unit address, device number, and subchannel number. Please remember that when we talk about a DASD device, we are referring to the 3380/3390 sort of logical device, *not* the physical RAID disks used to map the 3380/3390 as implemented in modern controllers. First we discuss the unit address.

The unit address, or UA, is the name by which a channel knows a device attached to one of its control units. The unit address is two hex digits in the range 00-FF. It does not necessarily have to match any piece of the device number for a device, but care must be taken during device definition since the unit address defaults to the last two digits of the device number if not explicitly specified. Also, some control units (3990 control units, for example), attached to S/390 ESCON serial interfaces require a UA range starting at 00.

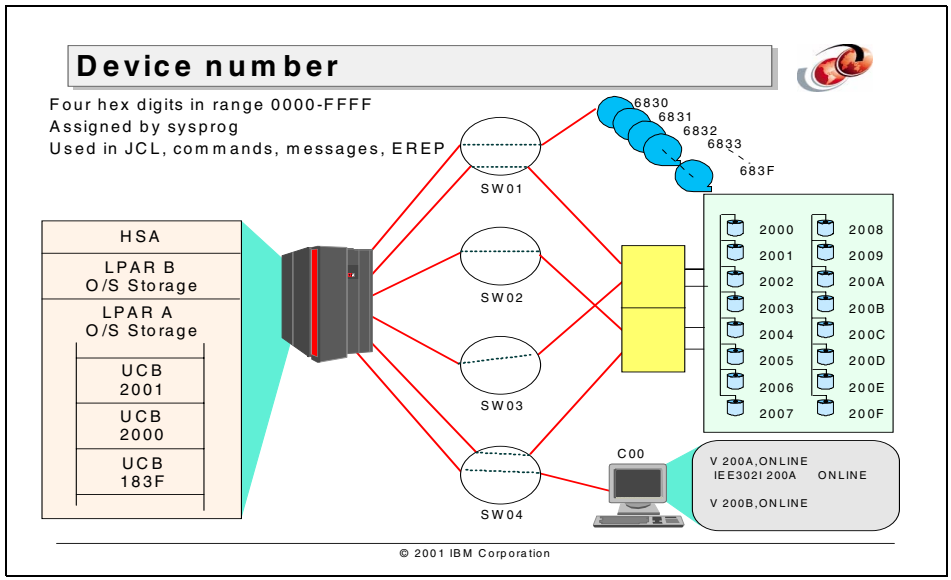
The unit address that is specified in HCD for a device must match the hardware UA on the physical control unit the device is attached to. The hardware UA is set by the engineer when the device is installed. The UA is not normally required by the operator during his/her daily tasks; however, it can be displayed using the DEVSERV command for control units supporting this function.

Parallel channels use the unit address to address the control unit (there can be more than one per channel) and the device (there are bits for the CU and bits for devices within that CU). As a result, each Parallel channel can support unit addresses from 00-FF, and so allows connection of up to 256 devices. ESCON-attached devices are addressed using a combination of the director link address (the address of the port that the controller is connected to), the control unit CUADD, and the unit address, in the format:

```
LINK.CUADD.UA
```

For a fiber channel (ESCON or FICON), each logical control unit supports the full unit address range 00-FF and may support up to 256 devices. By attaching a modern DASD controller to an ESCON channel on an IBM 9672 or IBM zSeries 900 CPC, you can attach up to 1024 devices to that channel (CUADD from 00 to 03). For a FICON Bridge or FICON Native channel, up to 16,384 devices can be attached (CUADD from 00 to 40).

9.1.8 Device number



The *device number* is used to identify a device in interactions between the system and humans—it is like a nickname.

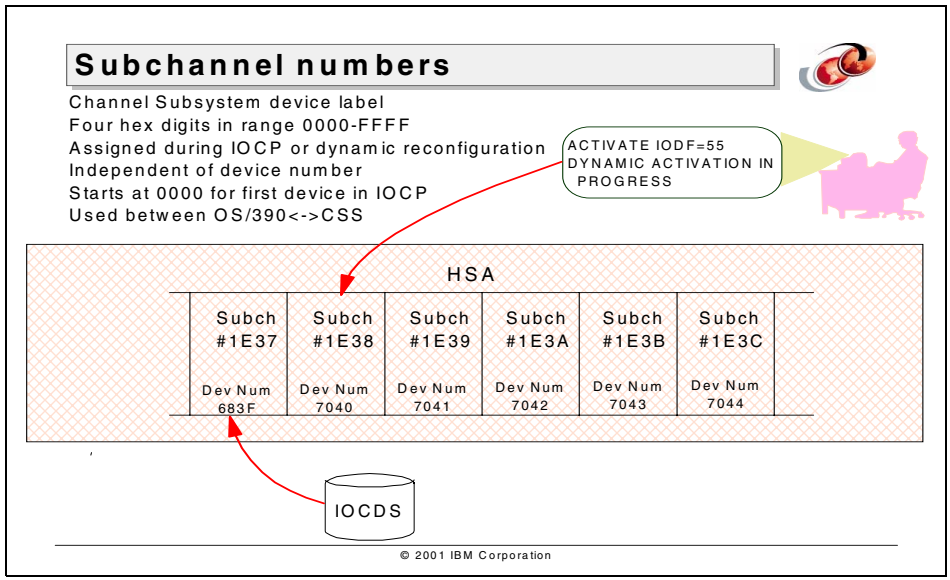
Every channel-attached device in an S/390 configuration is assigned a device number by the systems programmer. z/OS builds a control block called a Unit Control Block (UCB) for every device defined to it in the IODF, using the device number to uniquely identify each device. Device numbers are:

- Assigned by the systems programmer in HCD when defining a device to the channel subsystem (hardware) and z/OS
- An arbitrarily assigned number in the range '0000-FFFF', 16 bits, allowing for a maximum of 65,536 devices available to one copy of the operating system
- Used in JCL, console commands, console messages and error recording

If the device is shared between two systems in different CPCs, we strongly recommend that the same device number be used in each system to identify the same device.

Prior to MVS/XA, we did not have the concept of device numbers: the device address consisted of the (one digit) channel number and the (two digit) unit address. Starting with MVS/XA, the device number can be *any* hexadecimal number in the range 000-FFF (and later, 0000-FFFF), although most people stick to the convention of using the unit address for the last two digits of the device number.

9.1.9 Subchannel number



The *subchannel number* identifies a device during processing by the operating system and the channel subsystem; that is, during the execution of the SSCH instruction and the I/O interrupt processing.

Every device defined in the IOCDS (through HCD) is assigned a subchannel number. Subchannels are the hardware representations of devices. They are used by the channel subsystem to represent devices and to control I/O operations over channels to devices. A subchannel must exist in the channel subsystem to allow IOS to talk to a device. Subchannels are built at POR time in HSA, at LP activation, as a result of CHPID reconfiguration (if this adds the first path to the device for this LP), or when they are dynamically added using the z/OS ACTIVATE command. The characteristics of subchannels are:

- Subchannels are used in communications between IOS and the Channel Subsystem (CSS).
- The subchannel number acts as an index to the respective subchannel in the subchannel table. During the execution of the SSCH instruction the addressed subchannel needs to be located.
- Subchannel numbers are contiguous, starting at 0000, and are assigned when the IOCDS is created by the IOCP program.
- Subchannel numbers are usually not required by the operator, and are not displayed in the output from any operator commands.

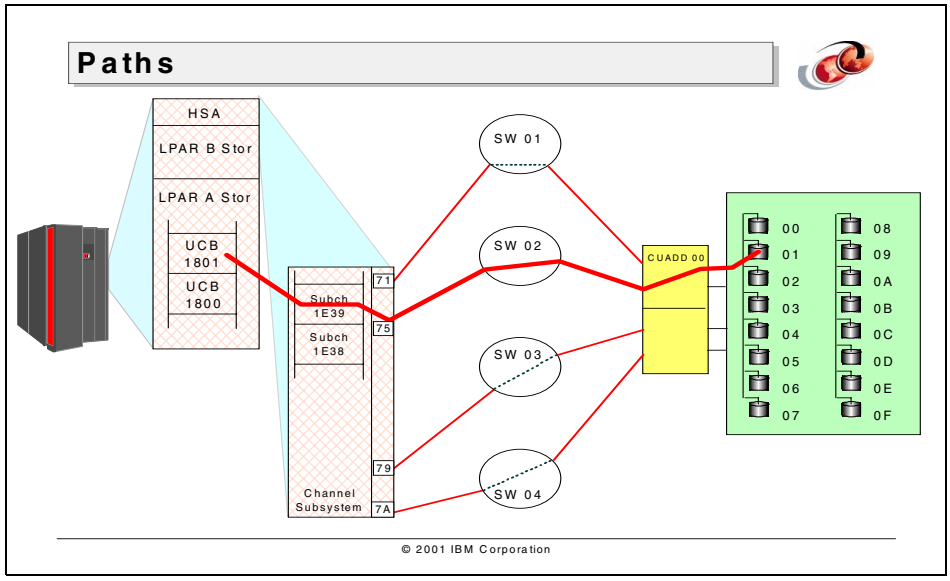
- The contents of the subchannel can be displayed on the HMC. There are two types of information in a subchannel:
  - Some of the information is static, such as which CHPIDs can be used to access the device. This information is obtained from the IOCDS during POR, dynamic reconfiguration processing, or from DCM.
  - The rest of the information is dynamic, and reflects the current state of the device and the current I/O operation.

There is a limit of 288K subchannels on an IBM 9672 G5 and G6, and 512K subchannels on an IBM zSeries 900. Each LP can have up to 36K subchannels. When a device is accessed by one or more shared channels, it is represented by one subchannel per defined LP (although the subchannel *number* for a given device is the same for every LP). The subchannels are kept in the HSA. When in either Basic or LPAR mode in a zSeries 900 or 9672 G5/G6 processor, there is just one HSA in the processor, and it is used by all LPs.

When subchannels are built (at POR time), they are left in a disabled state. They are also disabled during the initial stage of IPL. Later in IPL processing, z/OS attempts to match UCBs and subchannels. Any subchannel that has a matching UCB is enabled for that LP.

All the I/O data contained in IOCDS is used by the channel subsystem; control units are not aware of the IOCDS. There are also “devices” not defined in HCD that are represented by a subchannel number; for example, the ADMF function, crypto PCI-CC, and so on.

9.1.10 Paths



A path is simply a route to a device. From an operating system point of view, it flows from the Unit Control Block (UCB), which represents the device to z/OS, through the subchannel (which represents the device in the Channel Subsystem), over a channel, possibly through a director, and through the control unit to the device itself. In the diagram above, the path spans from the UCB for the device we wish to access (device number 1801), through the subchannel for that device (1E39 in this case), out to director SW02 via the channel with CHPID 75, through the control unit image with a CUADD of 00, and on to the device with unit address 01.

From a *hardware* point of view, a path runs from the channel subsystem, through the director, on to the control unit, and out to the device.

All those components must physically exist and be connected correctly in order for the path to exist. In order to avoid problems, the IOCDS should accurately reflect the actual physical connectivity. It is entirely possible (even likely) for a hardware path to exist without the corresponding software path existing; however, the software path cannot exist without the corresponding hardware components being in place.

Various IBM DASD control units support different numbers of host adaptors and logical paths. A logical path is a control block within the control unit. It is used to represent a channel instance actively interacting with the control unit. When a channel is initialized (at POR, CONFIG CHPID online, or LP activation) the

channel communicates with the control unit. At this point the control unit allocates the logical path control block that will be used for all the activities between the control unit and that channel instance. For a shared channel, a single physical channel can have several logical instances (and consequently uses several logical paths), one per each LP that shares that channel. If two different channels are communicating with the control unit by the same director port, and consequently by the same control unit interface, we also have two logical paths allocated. Note that each control unit has a limit on the number of possible logical paths.

The following table summarizes this information for the recent IBM DASD control units (note that even though some units only have 4 host adapters, it is still possible to have up to 8 logical paths to a single LP):

Control unit type	Host Adapters	Total logical paths
3990-3	16	16
3990-6	16	128
9345	4	64
9390 RAMAC 3	32	256
9393 RAMAC Virtual Array	16	128
9394 RAMAC 1	8	128
9395 RAMAC 2	8	128
9396 RAMAC Scalable Array	16	512
9397 RAMAC Electronic Array	16	512
2105 ESS	32	2048 (128 / CU image)

For non-managed channels, logical paths are established at LP activation for any control unit image that is on a channel that contains that LP in its channel access list. The channel access list is defined in HCD and contains a list of LPs that a MIF'ed channel is automatically accessible to when the LP is activated. By comparison, the channel candidate list is a list of LPs than can access a MIF'ed channel, but only by manual reconfiguration.

However, for managed channels, the logical paths are only established when a channel is added to the configuration for the control unit. This means that the introduction of Dynamic Channel-path Management does not, by itself, lead to an increase in the number of logical paths being used.

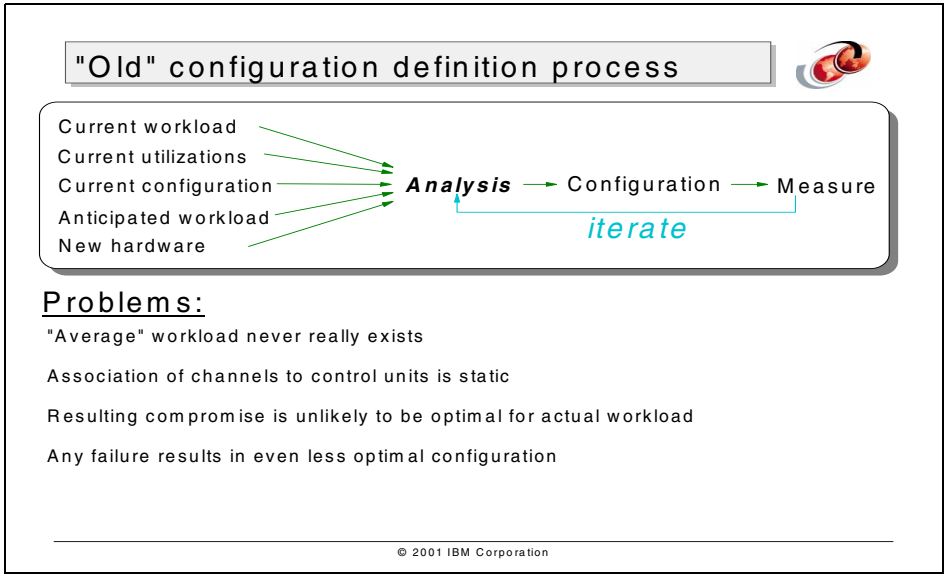
The paths to a device are initialized during the IPL by a process called *path validation (*regardless of whether DCM is active or not). In HCD, the Systems Programmer defines a number of non-managed CHPIDs as connecting to the control unit that the device is attached to. During path validation, the state of each path is determined. If a component of the supporting hardware is unavailable, that path is not brought online. For example, the CHPID may be configured offline, or the port in the director may be disabled.

To fully understand DCM, it is vital to understand the concept of paths to a device. From reading the preceding sections in this chapter, it should be clear that there are actually two requirements for an operating system to be able to access a device via a given path:

1. The hardware must be configured so that there is physical connectivity between the device and the CPC, *and*, that configuration must have been correctly defined using HCD.
2. The software must be aware of the paths, and must have varied those paths online. The z/OS VARY PATH command cannot cause a path to come online if the physical connectivity does not exist, or if the IOCDS does not contain the correct information about that connectivity. Similarly, you can use the z/OS VARY PATH command to stop z/OS using a path to a device, but that action does not change the fact that the hardware path still exists.

Dynamic Channel-path Management dynamically updates the underlying hardware configuration control information in HSA to add or remove paths to an LCU, and it updates the software to make it aware of the path changes and varying those paths on- or off-line accordingly.

9.2 Configuration definition prior to DCM

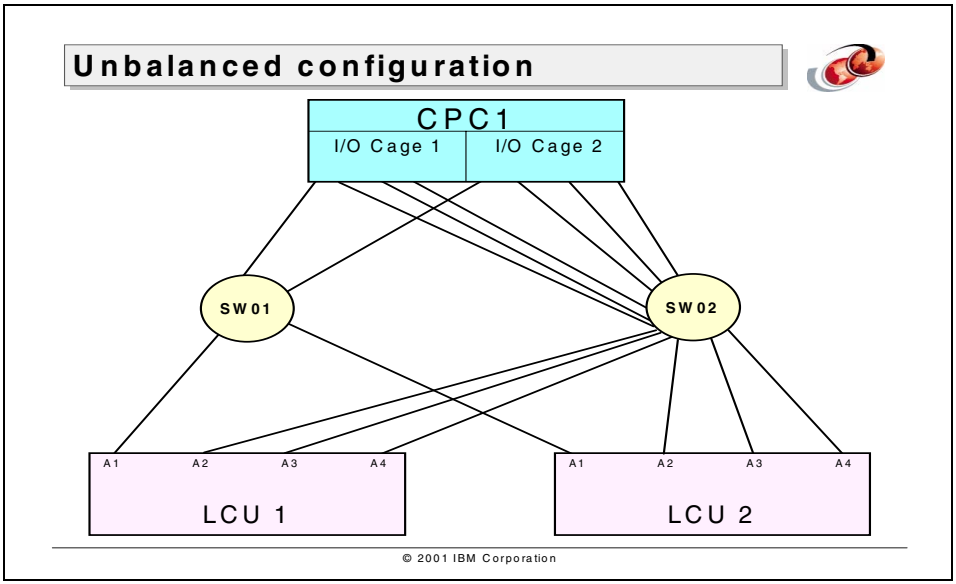


Now that we have described most of the terms relating to I/O processing, we move on to describing how the I/O configuration was defined prior to Dynamic Channel-path Management.

Prior to MVS/ESA V4.2 (which introduced Dynamic I/O Reconfiguration), if you wished to make a change, you had to do a power-on reset and IPL MVS to activate the changed configuration information.

This meant that you had to plan your I/O hardware topology in advance, and could not alter it without interrupting operations. Also, you had to plan your connectivity so that each control unit had sufficient channel bandwidth to accommodate peak workloads. The disadvantages of this approach were:

- You had to design a (relatively static) configuration that would support future (dynamic) workloads based on information about what had happened in the past.
- You probably had to install more channel bandwidth than you would require at one time. There are *very* few installations where the utilization across all channels is balanced and consistent over a 24-hour period!



- If you miscalculated, you might be affected by poor performance until you could either rework the configuration, or move some of the load off the busier channels.
- To ensure acceptable performance, you had to regularly monitor the load of all your channels, at all times of the day, to ensure that some were not becoming overloaded.

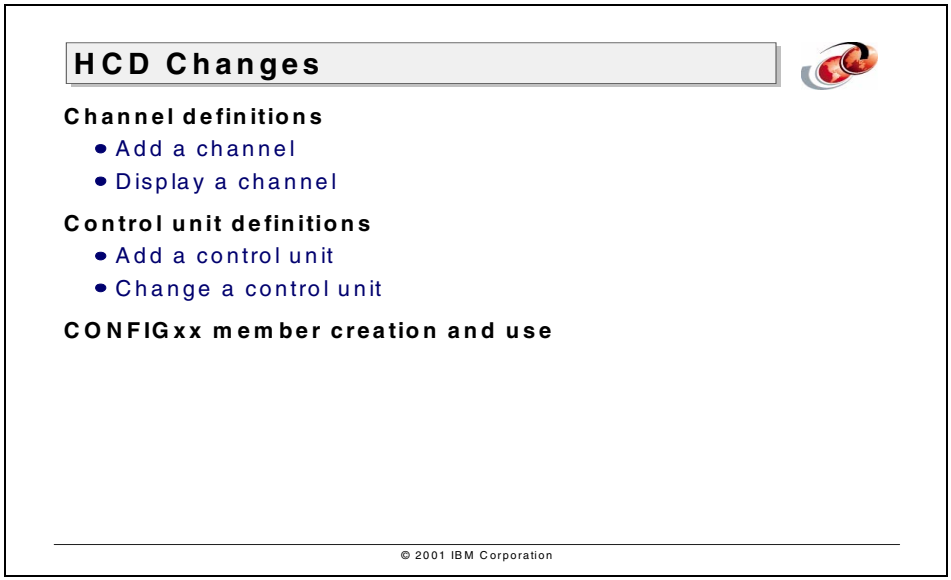
The introduction of the Dynamic I/O Reconfiguration capability in MVS/ESA V4.2 went some of the way to making configuration changes less intrusive. Most configuration changes could now be made dynamically, removing the requirement to interrupt the systems to activate a configuration change. However, it did not do anything to relieve the bandwidth-balancing considerations, or make it any easier to configure for maximum availability. DCM builds on the ability Dynamic I/O Reconfiguration provides to change the configuration without interrupting operations, and adds the benefits of enhanced performance, availability, and automation.

Another consideration was configuring for availability. When attaching a device to the CPC, you ideally want to have zero or very few single points of failure across the multiple paths. In a typical configuration, there are at least three entities in the path to a device: the CPC (including SAP and channels), a director, and the physical control unit. Each of these contain features to avoid single points of failure, however it is your responsibility to understand the features and configuration of each unit, to ensure that your topology exploits these features while at the same time delivering the performance you require. In a complex

environment, it can be difficult to arrive at a configuration that maximizes the availability features *and* provides the performance to an ever-changing workload. See the figure on the previous page for an example of an unbalanced configuration (many more paths routed through SW02 than SW01). Once again, Dynamic Channel-path Management takes all this information into account when deciding to activate a new path, and thereby removes a considerable amount of this responsibility from the Systems Programmer.

Note: It is still your responsibility to configure the physical connections in a manner that makes the best use of the availability features of your hardware. DCM can only work within the bounds of the physical configuration that you provide to it. However, as long as you provide balanced physical connectivity, DCM should avoid an unbalanced configuration like that shown in the diagram.

9.3 Configuration definition for DCM



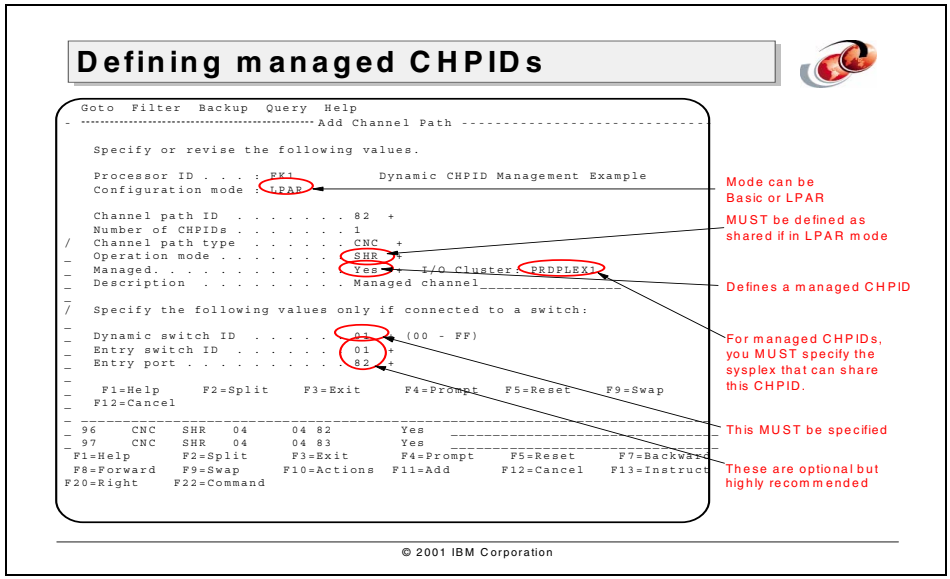
Now that we understand how the I/O configuration was defined and managed before DCM, we briefly describe how this is changed in a DCM environment. We then go on to describe in detail how DCM works.

Before you can start using DCM, you must define which control units and which channels you wish to have DCM control for you. This process is discussed in 10.5, “Identifying candidate control units” on page 287.

There are some changes to the HCD panels in support of this, and additional rules that HCD enforces.

There was also an enhancement to HCD (in OS/390 V2R6) that builds the CONFIGxx member of Parmlib for you, reflecting the configuration of the selected processor and operating system configuration. This function has been further enhanced to create information relating to DCM. This is discussed in more detail in 12.1, “New operator commands” on page 318.

9.3.1 Dynamic Channel-path Management channel definitions



There are two HCD definitions that are affected by the implementation of DCM: the channel definitions and the control unit definitions.

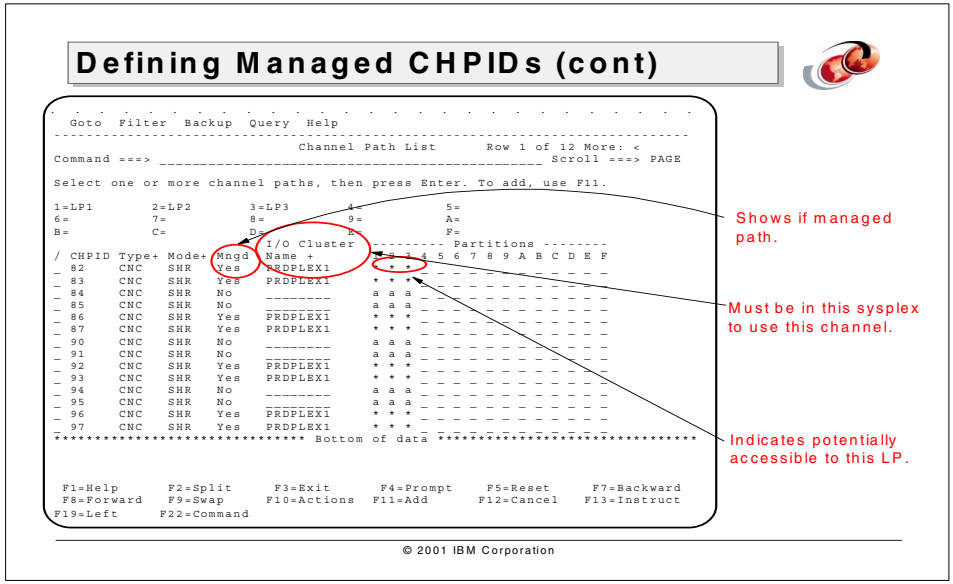
Starting with the channel definitions, the first thing to point out is that a managed channel can *only* be used to access control units that have been defined to use managed paths. You cannot specify the CHPID of a managed channel on any control unit definition. So, a channel is either a managed channel *or* a non-managed channel (one that is explicitly specified on a control unit definition)—it *cannot* be both. This restriction is enforced by HCD; the control unit itself has no idea whether a specific channel is managed or not.

As shown in the HCD dialog above, to define a channel as being managed, you have to specify the MANAGED attribute for that channel. As stated earlier, only ESCON or FICON Bridge (FCV) channels may be defined as managed. You also *must* specify a dynamic Switch that the channel is attached to. In addition, it is highly recommended, although not mandatory, to specify the entry Switch ID and port. This is discussed in more detail in 9.1.5, “Directors” on page 199.

In LPAR mode, if a channel is defined as being managed, you *must* also:

- Specify the sysplex name of the Parallel Sysplex that an operating system image must belong to in order to have this channel in its configuration. This name is specified in the *I/O Cluster* field. Later this information is used by the

hardware when it tells DCM in an LP which managed channels it is allowed to use.



- Define the channel as being shared. The reason for this requirement is that all LPs within the LPAR Cluster share the set of managed channels. If Dynamic Channel-path Management adds a channel to a control unit, that change is effective for all LPs in the LPAR Cluster.

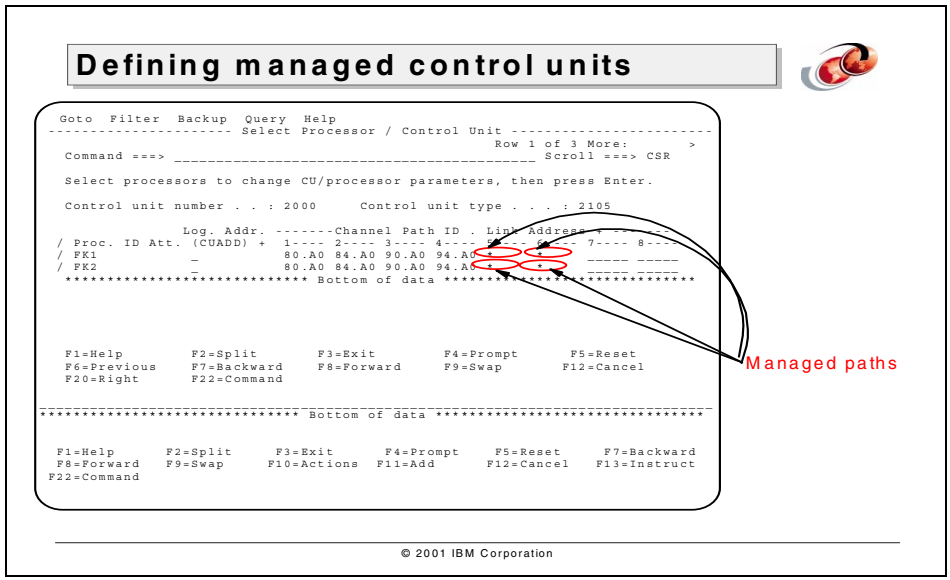
  However, unlike non-managed channels, which can potentially be shared by all LPs in a CPC (using MIF), managed channels can only be shared by LPs that are in the same LPAR Cluster. The reason for this is that an LP has to understand how all the paths to a control unit are being used before it makes a change; just because LP1 has very low utilization on a given path does not mean that LP2 is not using that channel intensively. The information about the use of each channel by the systems in the LPAR Cluster is kept in the WLM LPAR Cluster structure, and this is only accessible to systems in the same LPAR Cluster.

Because there is only one “Add Channel Path” panel for both LPAR and Basic mode, you have the option of defining the I/O Cluster name even if the CPC is in Basic mode. In this case, the I/O Cluster name is not saved in the IOCDS, however, it is stored in the IODF in case you later convert the CPC into LPAR mode.

You cannot define an access list or a candidate list for managed channels. The access to the channel is determined dynamically, based on the value specified in the I/O Cluster field. The HCD Channel Path List in the figure on the previous page shows an example of how a managed channel (which *must* be defined as shared) can only be accessed by other LPs that are in the same LPAR Cluster. The asterisk (*) is a new value that indicates that as any LP in the CPC could potentially contain a system in any (or no) LPAR Cluster, each LP must be eligible to use the shared channel—whether it is allowed to or not depends on the sysplex that it belongs to when it is IPLed. When you define a managed channel, it will automatically show this status for every defined LP, you do not have to define this manually as you do for access and candidate lists.

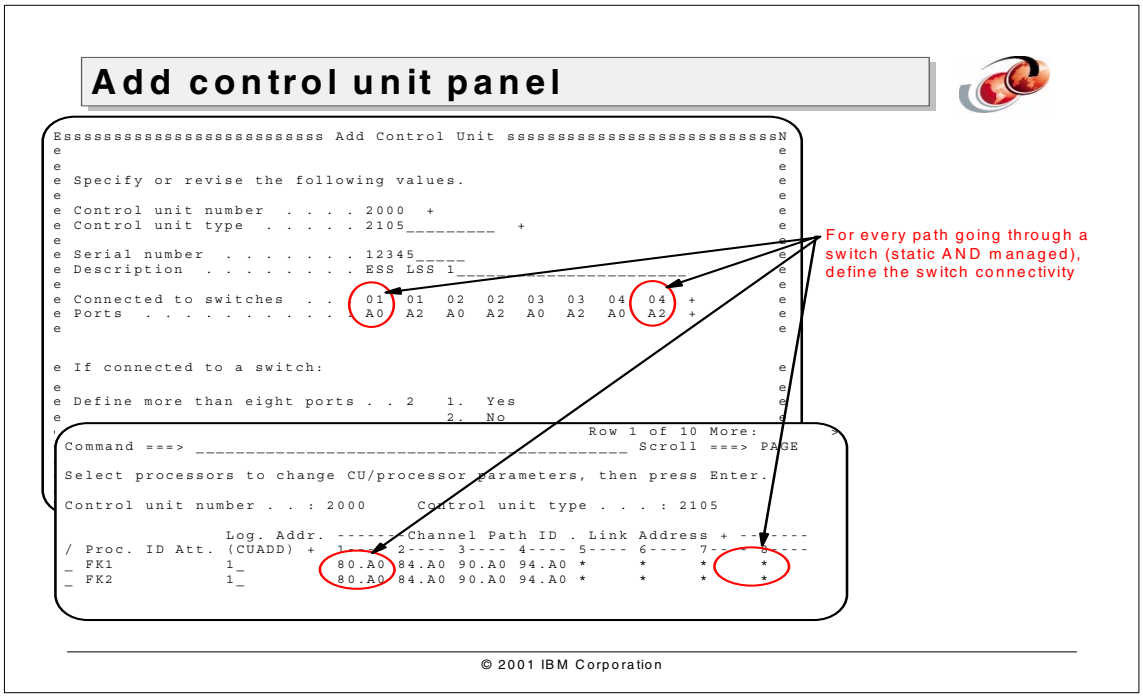
If you wish to change an existing non-managed channel to a managed channel, you must first remove the channel's CHPID from all control unit definitions before you change the channel definition to make it managed. 10.6, “Migration planning” on page 295 discusses how to manage HCD changes depending on your implementation approach.

9.3.2 Dynamic Channel-path Management control unit definitions



To define a managed CU, changes are required to the control unit definitions in HCD. You no longer have to specify the CHPID and link address of every path that is used to access the control unit. There are now two types of paths to a control unit: non-managed and managed. A *non-managed* path (sometimes also referred to as a static path) is a path where the CHPID number and link address are defined on the control unit definition. A *managed* path is one that is managed by Dynamic Channel-path Management. Managed paths are indicated by specifying an asterisk (*) rather than a CHPID number on the control unit definition. In the figure above, the IBM 2105 LCU 2000 is defined to have four non-managed and up to two managed paths from each I/O Cluster in CPCs FK1 and FK2; the IBM 2105 should be represented by just one control unit definition in HCD. However, if it was for an earlier control unit, such as an IBM RVA, there would typically be two control unit definitions in HCD for each control unit image in the physical control unit. Whichever way the control unit is defined, each LP can have no more than six paths to this LCU. If there were two I/O Clusters in each CPC, each one could use the four non-managed paths, plus two managed paths of their own, giving a total of eight paths from each CPC to the LCU.

In the HCD definition of the control unit, you specify the maximum number of managed channels *for an I/O Cluster* per CPC for that control unit. In the previous example, the control unit can be accessed by two managed channels from each I/O Cluster. Without knowing how many I/O Clusters there are on the CPC, there is no way to know the maximum number of paths that will exist in this example (within the limit of eight paths per CPC).



To be eligible for use with managed paths, a control unit must be attached to a director. In turn, the director must be attached to managed channels. This is because a director is the only way that a given serial channel can attach to more than one physical control unit.

Without the use of a director, all physical control unit connections using a fiber channel are point-to-point, rather than daisy-chained as was possible with parallel channels. If you wish, the *non-managed* paths can be direct, without going through a director.

If the CPC is in LPAR mode, *all* the managed channels used to access a control unit *must* be defined as SHARED. As with before DCM, you cannot mix shared and reconfigurable or dedicated channels on an LCU. Therefore, reconfigurable or dedicated channels may not be used to attach a managed control unit (because the managed channels *must* be shared).

You *must* define at least one non-managed path (which *must* be defined as shared) per HCD control unit definition. This is so the system can IPL (it only finds out about its managed paths after NIP processing), and also for Dynamic Channel-path Management to be able to identify the control units that it is to manage. For availability, we strongly recommend that you define at least two non-managed paths. This way, if one of the non-managed channels is offline at IPL time, you can still access the managed control unit through the other non-managed channel.

As shown in the figure on the previous page, on the first panel of the 'Add a Control Unit' dialog in HCD, you should specify the Switch and Switch port that *every* host adapter (also called channel adapter) on the control unit is attached to, regardless of whether that adapter connects to a non-managed or a managed channel. This is not an absolute requirement, but it is highly recommended. This is discussed in more detail in 11.1.1, "Managed Channel definitions" on page 303.

In order to avoid errors, HCD ensures that:

- When defining the paths to a control unit, you do not specify a non-managed path using a channel that has been defined as being managed. If you specify the CHPID of a managed channel when defining a control unit, you get a message similar to the following:

  ```
  Managed channel path 82 of processor FK1 can not be connected to control
  unit 2000.
  ```

- At least one non-managed path to the control unit still exists if you try to delete a path to the control unit. If you try to remove all the non-managed paths from a control unit, you receive a message similar to the following:

  ```
  Control unit 2000 can be connected to a managed channel path of
  processor FK1 only if there is at least one CHPID attached statically.
  ```

- Every managed control unit has at least one non-managed path. If you do not specify at least one non-managed path, you get a message similar to the following:

  ```
  Control unit 2000 can be connected to a managed channel path of
  processor FK1 only if there is at least one CHPID attached.
  ```

- Managed paths are only specified for DASD control units. If you attempt to specify a managed path for any other control unit type, you receive a message similar to the following:

  ```
  Control unit 6830 of type 3490 can not be connected to processor FK1 via
  a managed channel path.
  ```

- Preferred paths are not specified for any device connected to a control unit that has been defined to use managed paths. The preferred path feature tells

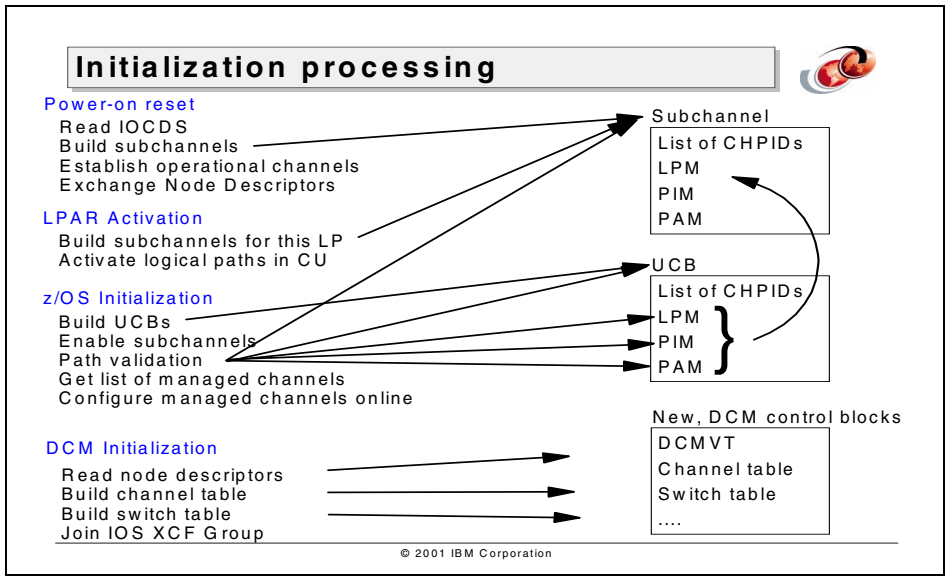
the SAP to always attempt to use the same “preferred” channel when trying to start an I/O operation to a specific device.

- Managed channels are not shown in the candidate channel selection list when defining a control unit. Rather, an asterisk (*) is shown at the top of the selection, indicating that managed channels are available for use by this control unit.

When Dynamic Channel-path Management adds an additional path to a control unit, it is effectively the same as a Systems Programmer adding another path to the control unit definition in HCD, building a production IODF, updating the IOCDS, activating the change in all LPs that are sharing that channel, and bringing the new path online in each of those LPs. Dynamic Channel-path Management exploits the dynamic I/O reconfiguration capability to add and remove physical paths to control units.

However, before we get into how this is achieved, we first have to discuss the process whereby DCM finds out what managed channels are available to it, and which control units and devices are potentially accessible through those channels.

9.4 Initialization changes



Having defined in HCD which channels and control units you wish to be managed, we now describe the processing that takes place from power-on-reset time through to when Dynamic Channel-path Management is ready to start managing the paths to control units. The diagram above summarizes the steps in this process, and the control information that is impacted by various steps.

It is not necessary for DCM to be active for this processing to take place (DCM can be turned on and off with an operator commend). This processing takes place on all z/OS systems (in z/Architecture mode) regardless of the current DCM status.

The IOCDS contains information about the configuration, and this information gets read at power-on-reset time. Also during power-on-reset processing, the subchannels are created in the Hardware System Area (HSA), with each one containing the subchannel number, unit address, and device number for each device defined in the IOCDS. Among other information, the subchannel data contains a list of the non-managed CHPIDs that are defined for the device in the IOCDS. Also, the PIM (Path Install Mask) field in the subchannel, indicating which of the fields in the CHPID contain a valid CHPID number (for example, does a value in the CHPID field of ‘00’ mean CHPID 00 or does it mean that there is no channel defined?), and the PAM (Path Available Mask) field, which indicates which of the paths are in the configuration for this device in this LP, are built. At this time, the subchannel indicates that the device is not available.

For devices behind managed control units, only the non-managed paths to the device are listed in the subchannel immediately after a power-on-reset.

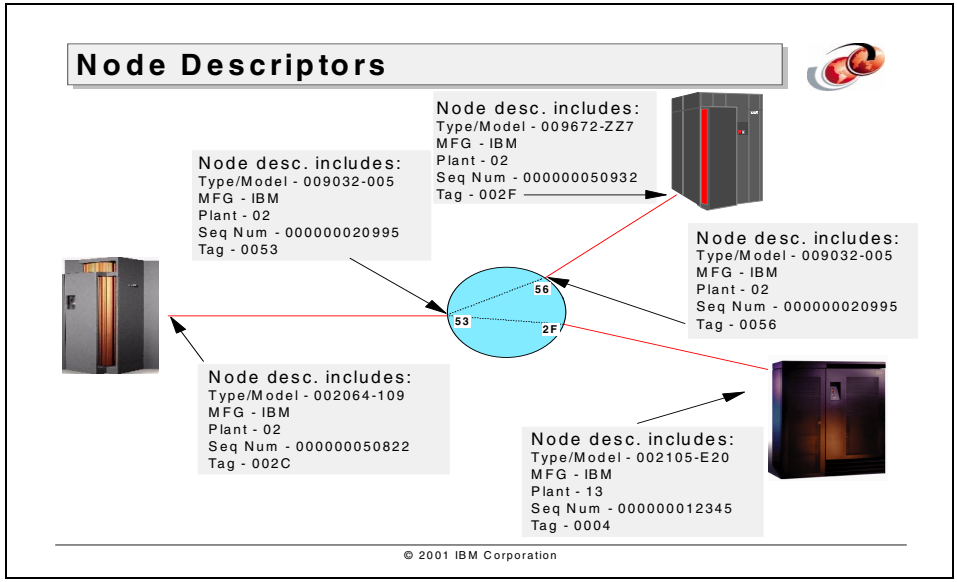
When an LP is activated, logical paths are established between the LP's channels and control units configured to the LP for all the non-managed channels in the configuration for that LP. No logical paths are established for the managed ones. For devices that are shared between LPs, there is a separate subchannel for each LP.

When an LP is reset (typically as part of the IPL process), all managed channels are removed from the configuration for that LP. This is one of the reasons you need at least one non-managed path to each control unit - if *all* paths were managed, it would not be possible to IPL! The reason for all the managed channels being removed is that at this time it is not known whether the system being IPLed in that LP is in the same LPAR Cluster as the system that was previously running in that LP, and the processor has to ensure that only systems in the same LPAR Cluster can access the managed channels belonging to that LPAR Cluster. Removing the channel from the LP causes the PAM bit in the subchannel to be reset, indicating that that path is no longer available.

When the LP is IPLed, part of Nucleus Initialization Processing (NIP) builds a UCB for every device defined in the IODF configuration in use by that operating system. The system then loads the UCBs with certain information from the corresponding subchannels. Any subchannel that has a corresponding UCB (by comparing the device number) is enabled, and the UCB is initialized with the list of non-managed CHPIDs and the PIM from the subchannel, and the subchannel number for the device (to be used as a parameter in the future SSCHs instructions). Subsequently, z/OS attempts to communicate with the device over each of the defined channels, and sets the Logical Path Mask (LPM) in the UCB and the subchannel to indicate which paths are really available.

All of the processing up to this point is common to OS/390 and z/OS. However, the processing that is described from here on is unique to z/OS, and takes place regardless of whether any managed channels or managed control units are defined or not.

At the end of z/OS Master Scheduler Initialization, z/OS (when running in z/Architecture mode) passes its sysplex name to LPAR LIC (if the CPC is in LPAR mode), to allow LPAR to determine which managed channels this LP can access. Having registered this information with LPAR LIC, z/OS then requests a list of the managed channels that this LP is permitted to use. LPAR LIC passes back a list of the managed channels that are defined in HCD as being usable by an LP in that I/O Cluster. z/OS then requests that each of the managed channels is brought into its configuration.



If this or other LPs have previously added any of the managed channels to a managed CU, those paths are brought online as part of the process of bringing the channel online. Unless this is the first LP in the LPAR Cluster to be IPLed after a power-on-reset, it is likely that there would be CUs already configured on the managed channels. Assuming other z/OS systems in the LPAR Cluster are active, bringing the managed channels online would result in this system having the same managed paths online as all the other systems in that LPAR Cluster.

Having brought all of its managed channels online, z/OS then requests the *node descriptor* for the neighbor of every channel. By neighbor, we mean the device the channel (or any element in the I/O path) is connected to.

A node descriptor is an unique set of data describing the element in the I/O path. It contains information such as: Type/Model, manufacturer, plant, and serial number that makes the element unique. It also contains (in the tag field) data that is used by DCM to avoid single points of failure.

For a managed channel, its neighbor should always be a Switch. This permits z/OS to start to create a map of what is physically accessible to each managed channel. The diagram above shows how each component in the hardware path has its own node descriptor, including channels, Switch ports, DASD controllers, and channels on other CPCs. The node descriptors help DCM understand the

topology of the configuration. Whenever an element of the I/O path is initialized, it exchanges node descriptors with its immediate neighbor. However, it is only during the topology build phase of z/OS initialization that a map of the whole configuration is built.

The next step (still in the topology gathering phase) is to request, from the Control Unit Port (CUP) on each Switch, the node descriptor for the neighbor of every port on the Switch. This returns the serial number and “device” type for every device (that is, either a control unit, CPC channel, or another Switch) that is directly attached to the Switch. In this context, device means any I/O element in the I/O path.

z/OS now has information about all the channels, and all the control units that could potentially be accessed via each of those channels, including which Switch port to use to access a given host adaptor (also called channel interface) on the physical control unit. This information is used to build the Switch Table. In addition (if the fix for APAR OW48164 is applied), if there are already other systems active in the LPAR Cluster, z/OS will get the DCM status of each Switch port from those systems and update its Switch Table accordingly. The information about the Switch is available using the D M=SWITCH command, as discussed is 12.1.2, “D M=SWITCH command” on page 320. The Switch Table contains the following information:

- Which channels connect to each Switch
- Which control unit’s host adapters connect to each Switch

It also obtains (through the tag field in the node descriptor) information about points of failure for the control units in the path to the device. It uses this information when making decisions about which paths to use when adding a new path to a device. This is discussed further in 9.9, “RAS benefits” on page 254.

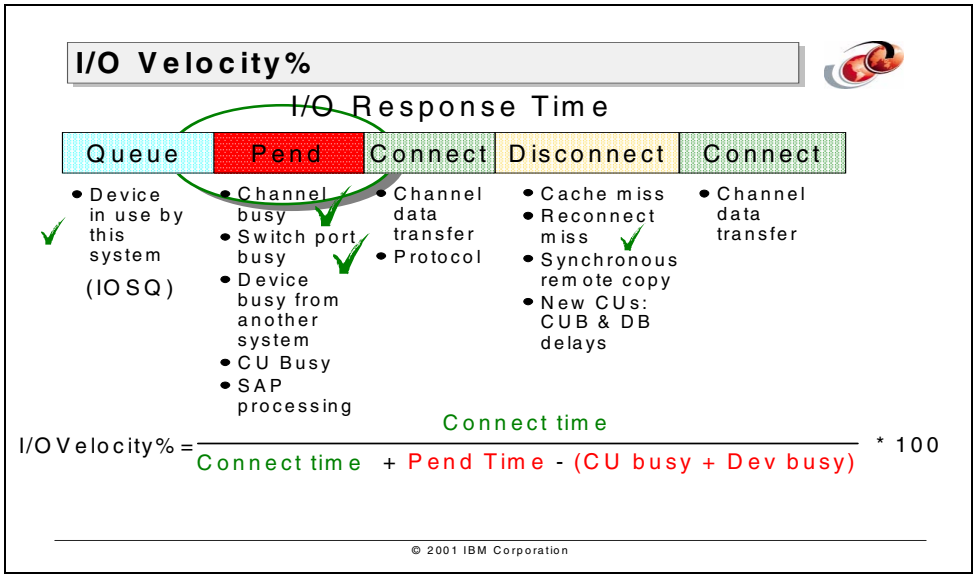
This topology information is gathered starting with OS/390 V2R10. Even though that level of OS/390 did not do anything with this information, the information is available to you via the new D M=SWITCH command. Obviously, the OS/390 system will not show any ports as being DCM-enabled since OS/390 does not support DCM.

At this point, LPAR LIC has made the managed channels available for use by the operating system, and z/OS is aware of the physical connectivity of the DASD devices. However, it has not yet started actively managing any of the managed control units.

As soon as DCM has information about the control units and managed channels, the first thing it will do is check to ensure that every managed control unit has at least two online channel paths. If it finds a control unit with only one online channel path, it will add one of the managed channels to the configuration for

that control unit. This is done to ensure that the failure of a single channel will not cause a total connectivity failure for that control unit. After this, it checks that every managed channel has a control unit on it. This is done to ensure that no channels are left idle when their capacity could be used to benefit a control unit.

9.5 I/O Velocity



At this point in the process, DCM has information about the connectivity options available to it. However, what prompts it to make a configuration change? The most common reason for DCM making a change is when it wishes to reduce the channel contention for a DASD control unit. To identify the effect of channel contention, DCM uses a new metric called I/O Velocity.

The I/O Velocity concept is based on the technique used by RMF Monitor III to derive the Workflow% and by WLM Goal mode for Execution Velocity metrics. This consists of using information about the amount of time the resource is being used and the amount of time it is being delayed. I/O Velocity is calculated for each LCU by dividing the amount of time the LCU is being used productively, by this amount of time *plus* an approximation of the amount of time the LCU is delayed by channel busy and Switch port busy. The formula is shown in the chart above.

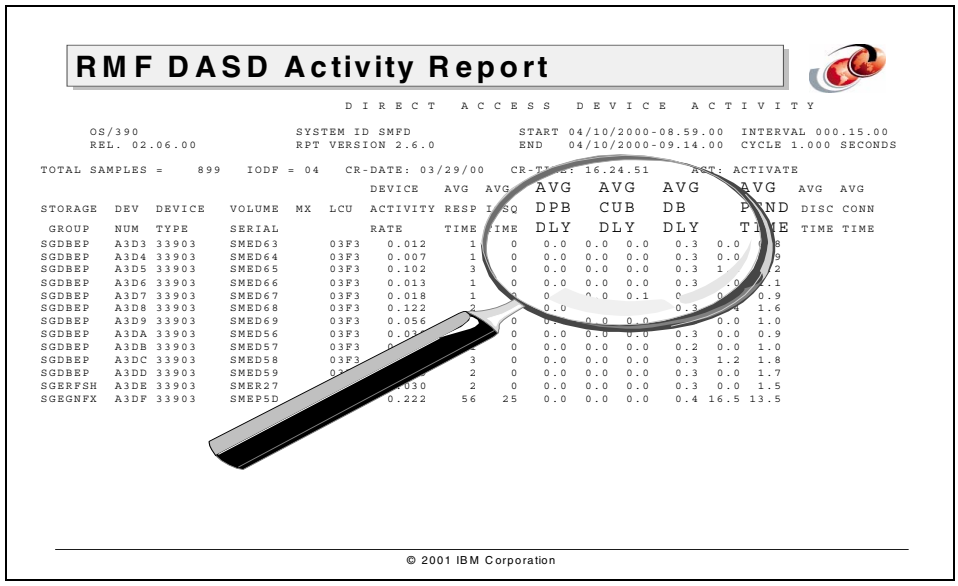
So, what constitutes “being used productively”? Dynamic Channel-path Management assumes that an LCU is being used productively when it is transferring data (to or from a channel), so the total device Connect time for all the devices behind the LCU is used for that part of the equation.

To calculate channel contention time and Switch contention delays, Dynamic Channel-path Management takes the total Pend time and subtracts the sum of CU busy time and device busy time. Pend time starts when the START

SUBCHANNEL instruction is issued, and ends when the I/O request is accepted by the device and the channel program starts execution. Refer to 9.1.3, “Channel subsystem logic” on page 194, for more information.

Pend time consists of:

- Time spent waiting because all the channels that connect the device are busy. This value is not shown directly in any RMF report.
- Time spent waiting because the director port on the Switch is busy (another channel is connected to the host interface in the physical control unit). This is shown in the AVG DPB DLY column in the RMF DASD Activity report on the next page.
- Time spent waiting because the CU is busy, that is, all the control unit internal paths are busy. This is shown in the AVG CUB DLY column in the RMF DASD Activity report.
- Device busy time—in most cases, this is time spent waiting because the device was reserved, or just being used by another system.
- The time the request is being served by the SAP. This is usually less than 0.5 ms. This value is not reported directly by RMF. RMF shows the SAP queue lengths in the I/O Queueing Activity Report.



Remember that one of the primary objectives of DCM is to ensure that DASD subsystems have the channel bandwidth they require. The impact of channel contention can be calculated as follows:

Pend Time - (CU Busy Time + Dev Busy Time)

It is important to note that the delays caused by channel contention also increase other parts of the I/O response time, such as:

- IOS queue time

  Any other delay (such as channel busy delay) causes previous I/O operations to take longer, thereby increasing the amount of time this request has to queue on the UCB waiting for them to complete.

- Disconnect time

  Channel contention may cause reconnect failures, consequently increasing the Disconnect time.

Using the formula for I/O Velocity, if the I/O Velocity% value is 100%, there is no channel contention. As channel contention (and therefore Pend time) increases, the I/O Velocity% drops below 100%.

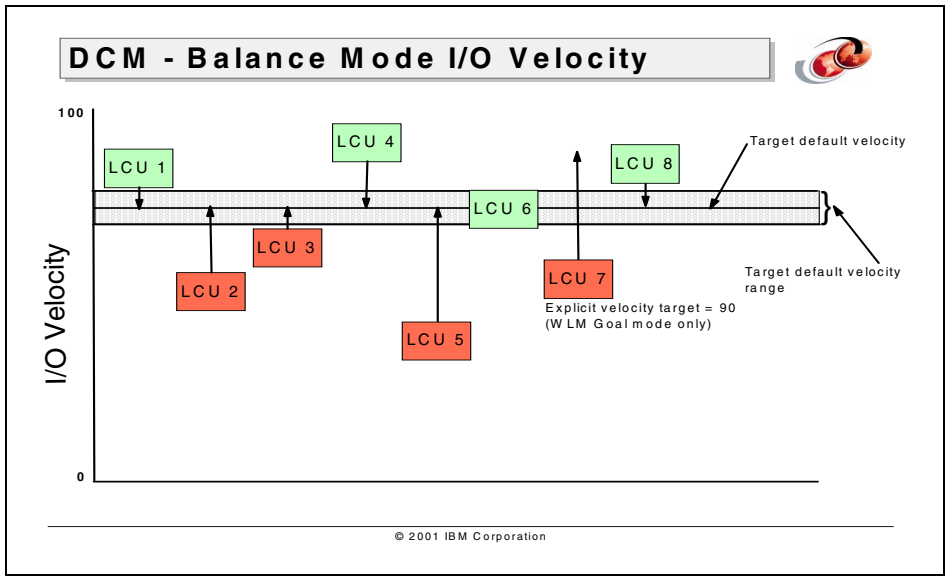
As a matter of interest, while we are talking about I/O response times, there is an interesting characteristic of Connect time that you should be aware of. Most people would expect that the Connect time to send a given volume of data to a given device will be fairly consistent. The benefits of larger buffers is generally

expected to be seen in reduced Pend and Disconnect times. However, in modern control units, the overhead in communicating between the control unit at the start of each I/O operation can be considerable, and this overhead is included in the Connect time. If you can reduce the number of I/O operations, you also reduce the number of times that this processing is invoked: the net result can be significantly reduced Connect times to transfer the same amount of data. The following table shows the numbers for two sequential loads of the same VSAM KSDS, with varying the number of data buffers.

Number of Data Buffers	Number of SSCHs	I/O Connect Time (secs)
Default (5)	39375	42.18
90	2821	20.5

As you can see, just decreasing the number of SSCHs (for the same amount of data transferred) resulted in cutting the I/O Connect time in half. This shows that good old-fashioned buffer tuning can still have a very positive impact on response times, channel utilization, and channel contention, even with the latest and greatest control unit technology.

9.6 Balance and Goal modes highlights



Dynamic Channel-path Management is now at a point where it can start to use the managed channels to provide more bandwidth to the DASD controllers.

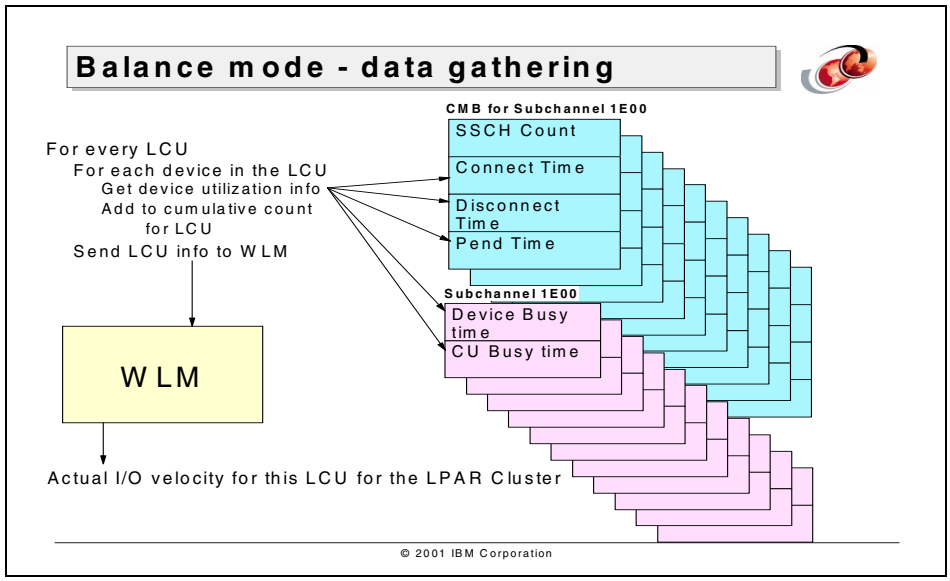
z/OS uses two algorithms when deciding on how much channel bandwidth should be available to a given controller. The first of these is known as *Balance mode;* it is used regardless of whether z/OS is running in WLM Goal mode or Compatibility mode. This algorithm is designed to attempt to ensure that all LCUs with managed paths have similar I/O velocities. Balance mode attempts to improve throughput for busy control units.

Dynamic Channel-path Management in Balance mode tries to move all managed LCUs into a band of similar velocities. As you can see in the figure above, there are LCUs above the range (the arrows are showing Dynamic Channel-path Management action to bring them down), below the range (the arrows are showing Dynamic Channel-path Management action to improve them) and one LCU, LCU 6, that is already within the target range.

The other algorithm, known as *Goal mode*, is only used if all the systems in the LPAR Cluster are running in WLM Goal mode. WLM investigates the cause of delays to service class periods (SCPs), and sets a specific target I/O Velocity for a specific controller, if it believes that action will result in improvements to the SCP that is missing its goals. If the controller is constrained, this action might result in taking channel bandwidth away from less “important” SCPs. In the figure

on the previous page, LCU7 has an explicit target I/O Velocity of 90% because WLM has determined that this is the velocity that is required to reduce I/O response times sufficiently to help the SCP meet its goals. Goal mode attempts to achieve better response times for the more important transactions.

9.6.1 Balance mode: data gathering and logic



The I/O Velocity of an LCU is the metric that is used by the Balance mode algorithm when deciding if it should add (or remove) a managed path to a particular control unit.

In the next few paragraphs we explain how the overall average LCU I/O Velocity is calculated. This is then used to calculate the range that DCM in Balance mode tries to move all managed LCUs towards.

On an interval basis, WLM asks IOS to collect and calculate the I/O Velocity for *every* LCU (both managed and non-managed). Because managed LCUs could potentially share non-managed paths with non-managed LCUs, DCM needs to know the I/O Velocity of *all* LCUs. The reason for this, is that removing a managed path from an LCU could increase the channel utilization on the remaining channels, and therefore indirectly impact the performance of other (managed *and* non-managed) LCUs on those channels. This is like a domino effect, where a change to one entity affects another, which in turn affects another, and so on. When DCM is projecting the impact of a configuration change, it also takes into account the impact on these other control units.

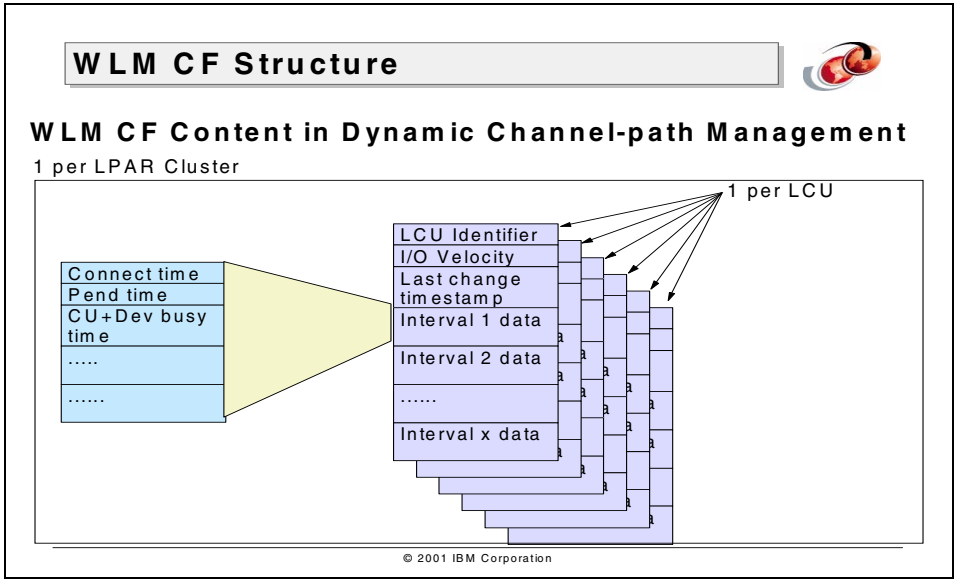
IOS collects utilization information from the Channel Measurement Block (CMB) and the subchannel for each device on an LCU. As they both contain information that is unique to each system, this data collection process takes place on every system in the LPAR Cluster. Each IOS gathers the required information for every device behind an LCU, sums it, and calculates the delta from the last

measurement interval. This information is then sent to WLM, which may update a CF structure with this information. A Coupling Facility is only required if the z/OS image is running in an LP and is part of a multi-system sysplex. The CF structure used by DCM is the same structure that is used by WLM LPAR CPU Management.

WLM maintains a set of control blocks, each of which contains a number of intervals worth of performance data for one LCU. The structure of these control blocks is shown in the figure on the next page. If the system is part of a multi-system sysplex, these control blocks are also maintained in the CF structure. When IOS passes LCU utilization information to WLM, WLM reads the entry for the current interval for this LCU from the CF structure, to get the equivalent information for the other systems in that LPAR Cluster. It adds the information for this system, and writes the updated information back to the CF structure, resulting in the CF structure containing LPAR Cluster-wide information for each LCU. At the same time, it extracts the information from the preceding intervals, calculates the actual I/O Velocity for those intervals and returns that information to IOS. IOS then saves these values for the LCU.

This process takes place for every LCU. If the data is for a managed LCU, IOS also uses this information to calculate an overall *default target I/O Velocity* for *all* the managed LCUs. This is used to calculate a range that DCM in Balance mode tries to move all managed LCUs towards.

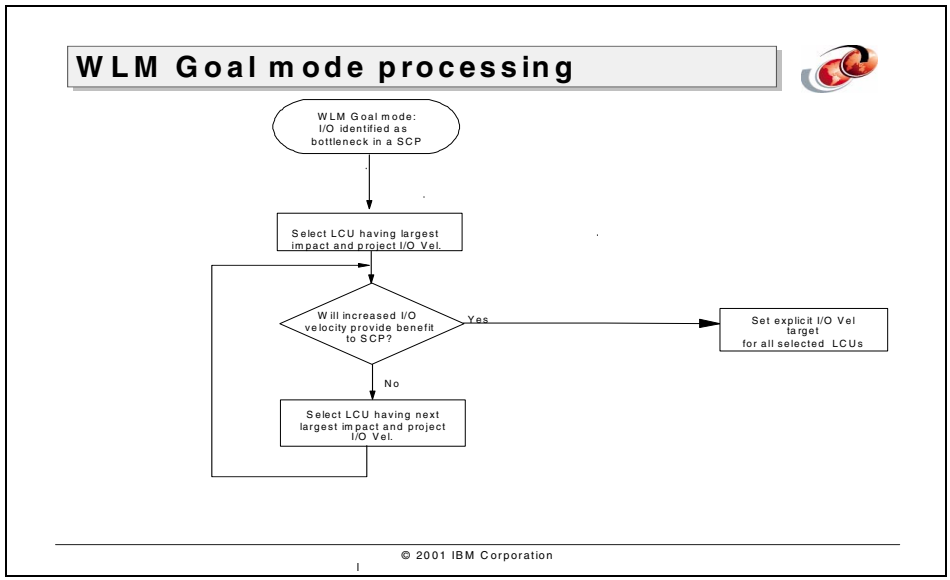
The use of the CF structure allows WLM to take the I/O activity on all systems in the LPAR Cluster into account when calculating the I/O Velocity for each LCU. The diagram on the next page shows the contents of the WLM structure: a single, consolidated set of information for each LCU, containing cumulative information from all the systems in the LPAR Cluster.



If one of the systems in the LPAR Cluster loses access to the structure, DCM in *all* the systems in the LPAR Cluster stops making any further adjustments until that system regains access to the structure, or is partitioned out of the sysplex. This ensures that adjustments won't be made that would impact a system when DCM can't see the activity of that system. For example, there might be a development system and a production system in the LPAR Cluster. If the production system temporarily loses access to the WLM structure, you would not want DCM on the development system (which no longer has information about the activity on the production system) to start taking paths away from an LCU that is being used by the production system.

If an LCU is shared by images in the LPAR Cluster and images that are not part of the LPAR Cluster (not z/OS for example), the non-managed channels are indirectly affected if DCM decides to add or remove a path to an LCU. If DCM adds a path, the load on the existing non-managed channels will probably decrease, helping the systems that are using those paths. If DCM removes a path, then the load on the non-managed channels will probably increase, potentially impacting the systems using the non-managed channels.

9.6.2 Goal mode



In addition to Balance mode, if all the systems in the LPAR Cluster are in WLM Goal mode, then DCM will also operate in Goal mode.

When WLM is in Goal mode, it runs its Policy Adjustment routine every ten seconds. During Policy Adjustment, WLM will select a Service Class Period (SCP) that is missing its goal. If I/O delay is the largest cause of the goal being missed, and the I/O requests for the SCP are suffering channel contention, WLM identifies the LCU causing the most delay. It then projects the impact of raising the I/O Velocity of that LCU. If the projection shows that this will provide a sufficient benefit, WLM sets an explicit target I/O Velocity for that LCU. If not, it projects the effect of *also* increasing the I/O Velocity for the LCU causing the next-most delay, and so on until a benefit is achieved. These explicit targets will override any attempts by Balance mode processing to bring those LCUs within the band of average velocities.

As the SYSTEM and SYSSTC service classes do not have WLM goals, WLM attempts to help those service classes if channel delay is a significant component of the delay encountered by those service classes.

As the I/O velocities apply across the LPAR Cluster, and WLM is aware of the Performance Index (PI) for the SCP for each system in the LPAR Cluster, WLM on one system may decide to set an explicit target for an LCU, even if the local PI of that SCP is acceptable. Because the changes made by DCM affect all the systems in the LPAR Cluster, WLM in one system is therefore capable of helping an SCP running in another system.

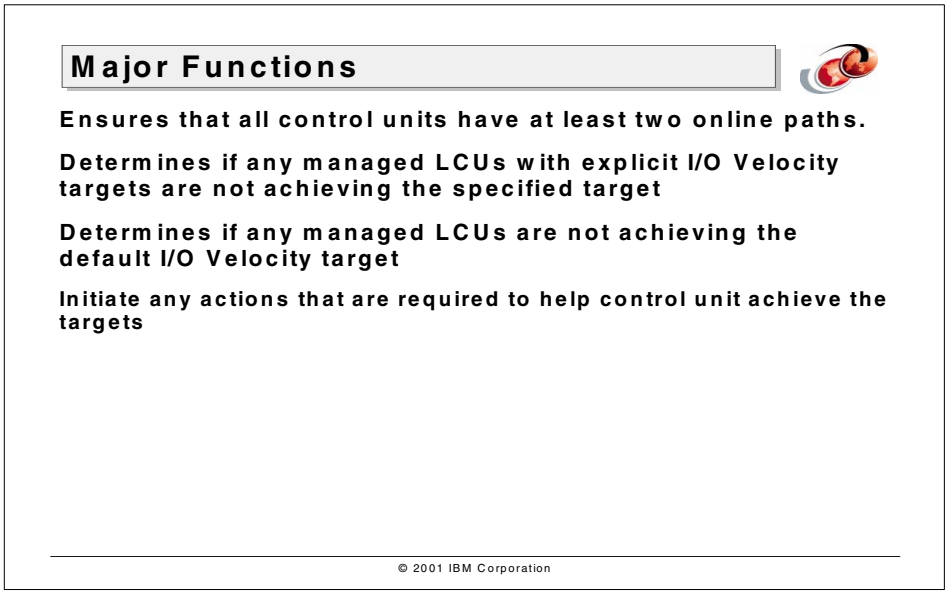
For normal WLM processing, WLM will help just one receiver per Policy Adjustment cycle. However, although WLM drives DCM processing, it does not happen as part of Policy Adjustment and it also is not limited to making one LCU change per interval. Therefore, DCM can potentially help a number of LCUs per interval. Having made a change to an LCU, DCM on any system in the LPAR Cluster will not make a further change to that LCU for a number of intervals. This is to prevent over-reaction to poor I/O velocities.

If an SCP that is using an LCU with an explicit target is no longer being delayed by that LCU, WLM gradually reduces the explicit target until it matches or is lower than the default target I/O Velocity. At that point, the explicit target is removed. This avoids drastic configuration changes.

It is important to understand that WLM just sets target velocities for the LCUs. The actual decision about which paths should be added or removed to meet that target is made by IOS during a process known as *imbalance correction.* However, WLM and IOS work together to assess the potential impact of the different possible configuration changes that IOS could implement.

If *any* system in the LPAR Cluster goes into WLM Compatibility mode, all the explicit targets are nullified by WLM as part of the transition.

9.6.3 Balance checking and imbalance correction



With each system in the LPAR Cluster having gathered the current I/O Velocity information for every LCU and calculating a default implicit target I/O Velocity (if in Balance mode) and potentially explicit I/O Velocity targets (if in WLM Goal mode), WLM then invokes a *balance checking and imbalance correction* service that executes four functions:

- Ensure that every managed LCU has at least two paths. This helps avoid single points of failure should a channel path break, or if the path is inadvertently taken offline manually.
- Determines if any managed LCUs with explicit I/O Velocity targets, as set by WLM, are not achieving the specified target. This function is also part of *balance checking*.
- Determines if any managed LCUs are not achieving the default I/O Velocity target, as determined by DCM Balance mode. This function is part of *balance checking*.
- Initiates any actions that are required to help control units achieve the targets. This function is part of *imbalance correction*.

Balance checking first searches for LCUs whose I/O Velocity is outside the default target I/O Velocity range (either below *or* above), or else those that have an explicit I/O Velocity target set. From this list, it excludes any LCUs that have been modified (that is, had a path added or removed) recently.

When in WLM Goal mode, IOS then calls WLM to prioritize the LCUs that have explicit targets. This is done to ensure that the most important LCUs are addressed first. IOS addresses all the LCUs with explicit targets before it moves to those with a default target.

Dynamic Channel-path Management (now in IOS) then decides what actions should be taken to move each LCU towards the target I/O Velocity. When making this decision, DCM takes the following things into account for each LCU:

- Is “Director Port Busy” causing the problem? If so, try to move one of the existing paths to another, currently unused, port on the same Switch.
- Will adding paths to the LCU raise the I/O Velocity? If so, which paths are candidates for adding to the LCU?
- If it were to add a path, would this conflict with any of the rules for the device? For example, would this result in more than 8 paths from an LP to the LCU? Or would it result in more managed paths to the LCU from the LPAR Cluster than the installation allowed for in the HCD definition?
- If it were to add a path to the LCU, what impact would that have on other LCUs already on that path?
- Will removing other LCUs from paths that this LCU is using raise the I/O Velocity? If so, which LCUs should be removed from which paths?
- It is were to remove one of the other LCUs from a path being used by this LCU, what would the impact be on that LCU?
- When deciding whether to add or remove a path from an LCU, Dynamic Channel-path Management takes single points of failure into account. It prefers to add a path that does not have any single points of failure in common with those paths that are currently being used to access the device.

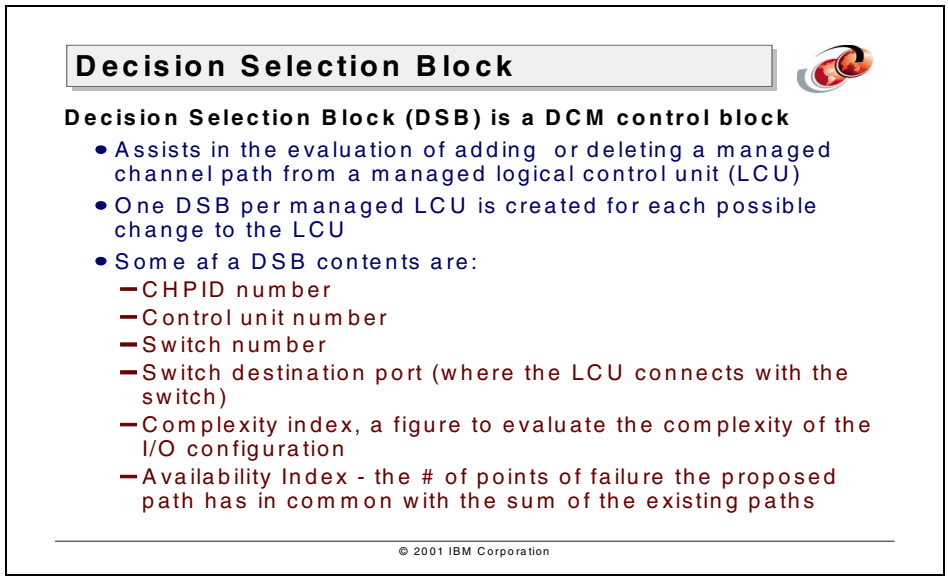
  DCM also prefers a path that is not currently being used by any other LCUs. This ensures that all available bandwidth is used before DCM initiates a change that could potentially impact another LCU.
- If an LCU with an explicit target is going to be negatively impacted in order to help this LCU, DCM asks WLM to decide which action should be taken. This is to ensure that the most important workload is not impacted by the decision.

Having evaluated all these (and other) criteria, DCM implements its decision.

As part of implementing the decision, a timestamp is set, indicating that a change has been made in relation to that LCU. This timestamp ensures that no further changes will be made (by *any* of the systems in the LPAR Cluster) for a number of intervals, to give the change a chance to affect the I/O Velocity.

DCM produces a combination of Component Trace and SMF Type 99 records to document its actions. These are discussed in more detail in 12.5, “Problem determination” on page 340.

9.7 Decision Selection Block

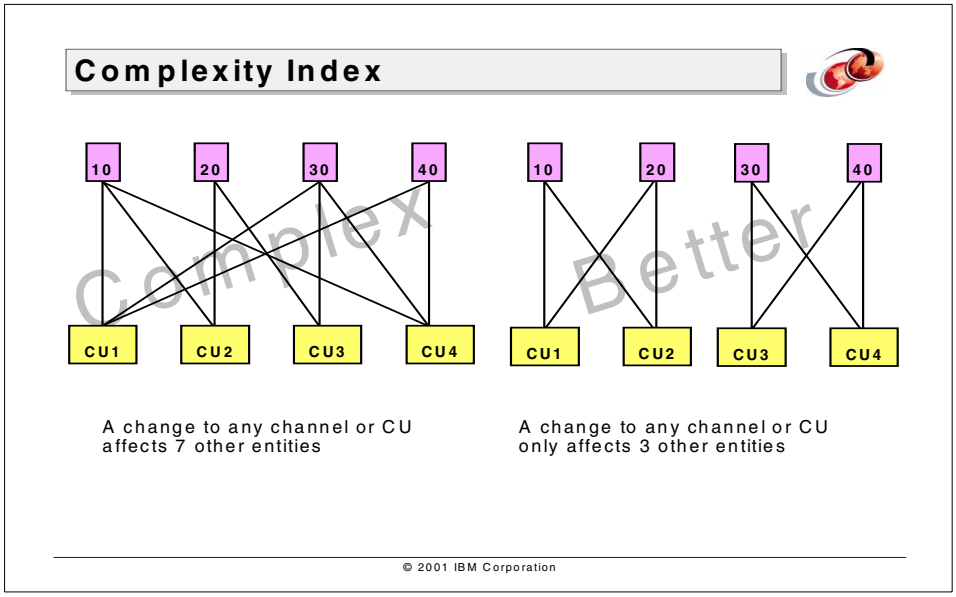


A new term that you may encounter when working with Dynamic Channel-path Management is *Decision Selection Block* (DSB). This is a new control block that is built during the process of deciding whether to make a change to the number of paths in an LCU configuration (during the imbalance correction routine).

A DSB contains all the information required by DCM when deciding what is the best action to take to improve the I/O Velocity of a given LCU. There is one DSB created for each possible action (adding or deleting a path) for each LCU:

- For each path (CHPID.link) that can be added, there is one DSB.
- If there are other LCUs on any of the managed paths in use by this LCU, one DSB is created for each LCU that might be removed from any of those paths.

When building the DSB, DCM checks to see which other LCUs and CHPIDs could be indirectly impacted by each possible action, and that information is added to the DSB. This ensures that DCM is able to assess the impact on other LCUs when deciding which is the most appropriate action to take to help an LCU.



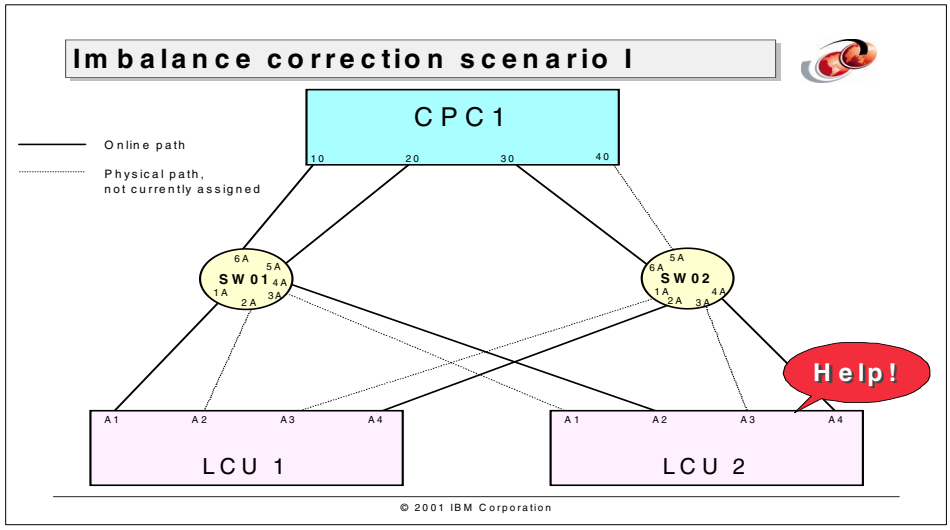
DCM completes all the processing associated with one LCU before it moves on to the next. So, DCM identifies an LCU that needs help, builds the DSBs for all possible actions to help that LCU, identifies the best course of action, and initiates that action before moving on to the next LCU.

In order to help keep the configuration as simple as possible (thus minimizing the domino effect), DCM calculates an index (which we call the *complexity index)* to represent the number of interconnected channels and control units that would result from the different actions done to them.

As more channels and LCUs become interconnected, each change that DCM makes potentially affects a larger number of other entities (channels and LCUs), as shown in the figure above.

In the configuration on the left in the (simplified) diagram above, if DCM adds or removes a path to any of the four LCUs, it affects seven other entities. In this case, the complexity index is eight (1+7). This makes it more difficult to identify any change that does not have an impact on one of the other LCUs. It also increases the amount of processing that DCM must do to calculate the impact of each potential change.

On the other hand, if a path is added or removed from one of the LCUs in the configuration on the right in that diagram, only two LCUs and two (possibly three) channels would be interconnected (and, as a result, affected). In this case, the complexity index is 4 (2 channels plus 2 LCUs). For this reason, Dynamic Channel-path Management attempts to select an action that would result in the lowest complexity index, all other things being equal.

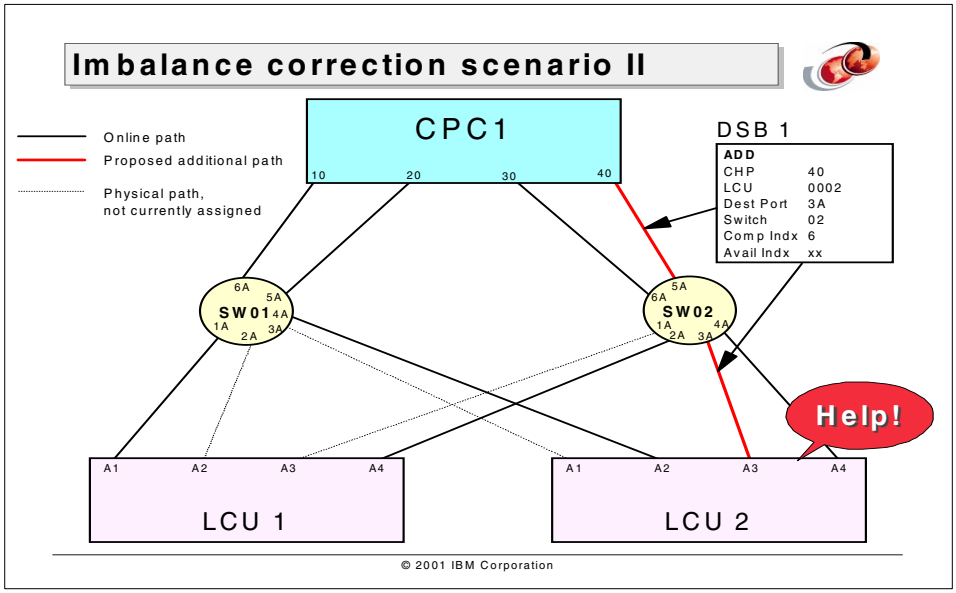


In the diagram above, there are two LCUs: LCU 1 and LCU 2. LCU 1 has two physical connections to Switch SW01 and two physical connections to Switch SW02. Similarly, LCU 2 has two physical connections to each of the Switches. Each Switch in turn has two physical connections back to the CPC (channels).

Note: The diagrams used for this illustration are not meant to represent a recommended configuration—they are simplified to make it easier to understand the actions of DCM. Obviously, we would not recommend having just two paths to an LCU.

At the beginning of this scenario, LCU 1 has two paths configured and online: path 10.1A (CHPID 10 and link 1A) through SW01 and path 30.2A (CHPID 30 and link 2A) through SW02. LCU 2 also has two paths: CHPID 30.4A through SW02 and CHPID 20.4A through SW01.

During DCM data gathering and the subsequent balance checking, it is determined that the I/O Velocity of LCU 2 is below the default target I/O Velocity. This causes DCM to start building DSBs to identify the best course of action to take to improve the I/O Velocity of LCU 2.



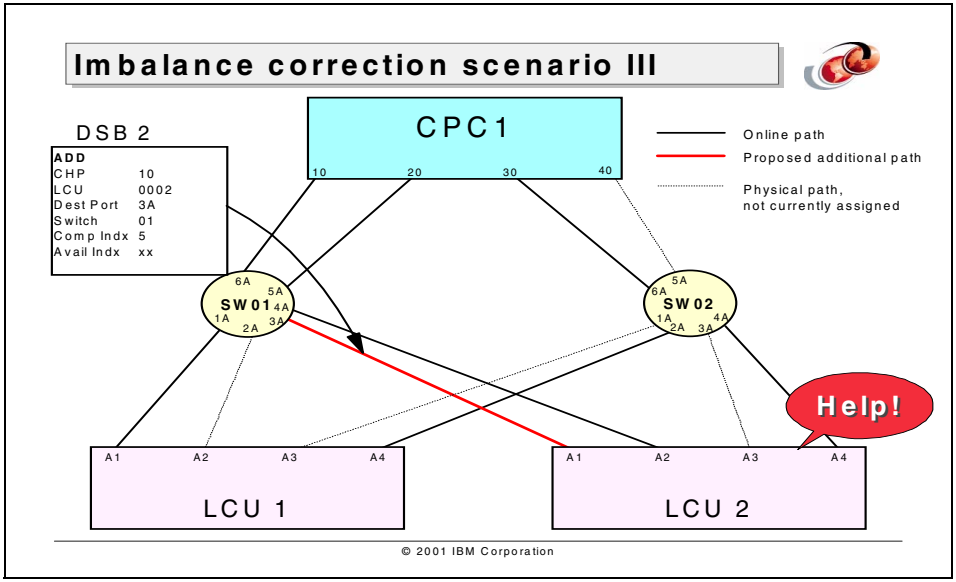
The first option that DCM builds a DSB for is to add a path using CHPID 40 and link address 3A to the configuration for LCU 2.

The DSB indicates that the proposed action is to ADD a path using CHPID 40, Switch SW02, and director port 3A to the configuration for LCU 2. It also contains information about the current and projected I/O Velocity for that LCU. The *complexity index* (represented by 'Comp Indx' in the diagram above) is '6', indicating that this change would result in five entities (channels and LCUs) being interconnected (CHPIDs 10, 20, 30, 40, and LCUs 1 and 2).

DCM would also calculate an Availability Index that would represent the number of points of failure that the proposed path has in common with the sum of the existing paths.

In summary then, DCM calculates the following:

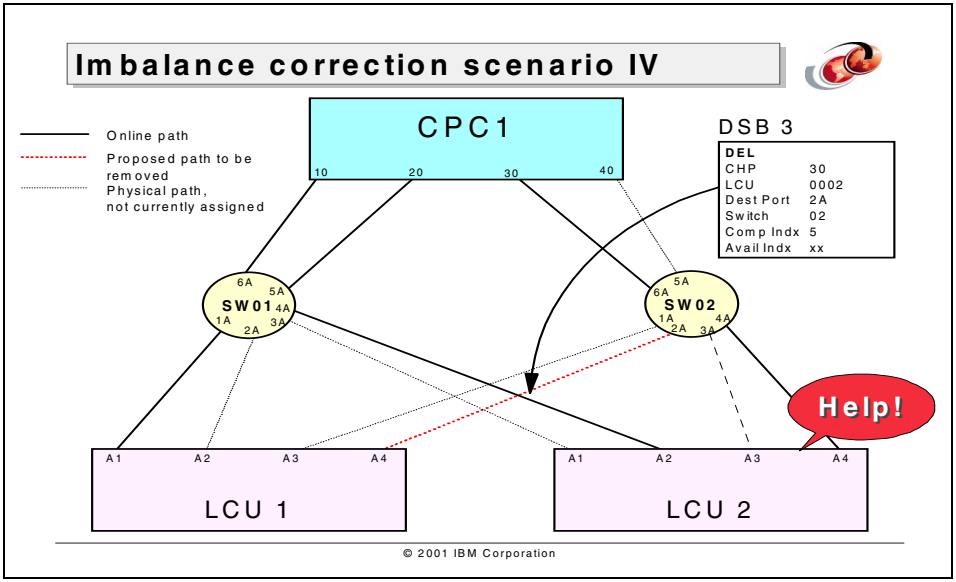
- The performance impact of making this change (in I/O Velocity terms)
- The complexity index
- The availability index



The next option might be to add CHPID 10 to the configuration for LCU 2. DCM builds a second DSB (DSB2 in the diagram above) to represent this option.

In this case, the DSB will indicate that the proposed action is to ADD a path using CHPID 10, Switch SW01, and director port 3A to the configuration for LCU 2. Once again, it calculates:

- The performance impact of making this change (in I/O Velocity terms).
- The complexity index (in this case, '4': three CHPIDS (10, 20, and 30) and two LCUs (1 and 2)).
- The availability index.



The third and last option that we describe is to remove CHPID 30 from the configuration of LCU 1. DCM builds a third DSB to represent this option.

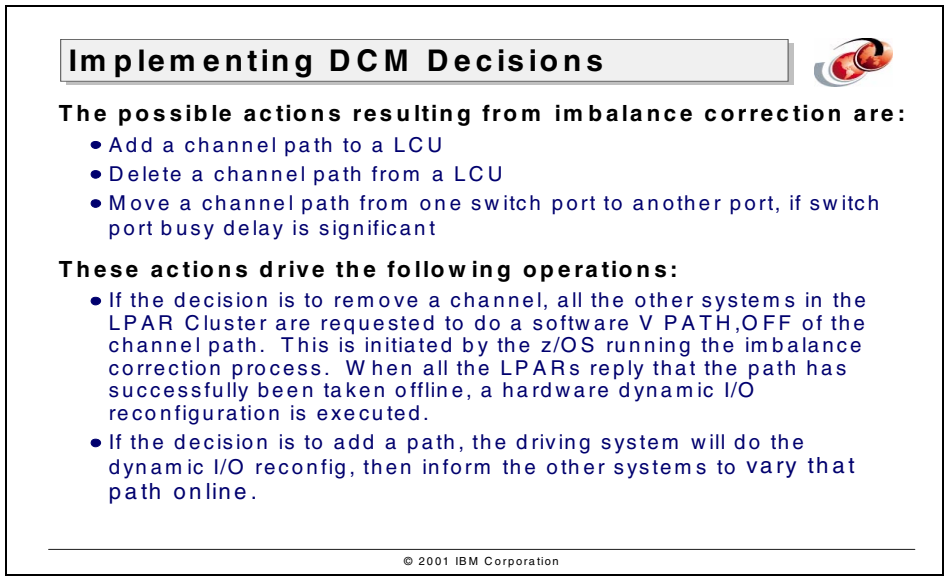
In this case, the DSB indicates that the proposed action is to DELETE the path using CHPID 30, Switch SW02, and director port 2A from the configuration for LCU 1. The DSB still contains the name of the LCU that is being assisted by this action—LCU 2. The reasoning behind this proposal is that removing LCU 1 from CHPID 30 would reduce the load on that CHPID, thus reducing the channel contention being suffered by LCU 2. Once again, it calculates:

- The performance impact of making this change (in I/O Velocity terms).
- The complexity index (in this case, ‘5’ (three CHPIDs (10, 20, and 30) and two LCUs (1 and 2)).
- The availability index.

In fact, this option will be discounted by DCM. The reason for this is that if DCM were to implement this, LCU 1 would only have one path, and DCM will never leave a managed control unit having only one path.

There are also numerous other possibilities in this scenario, and DCM builds a DSB for each one. When a DSB has been built for all scenarios, DCM identifies the “best” one and implements that.

9.8 Implementing DCM decisions



Having identified the potential actions that can be taken, DCM must select one of the actions (add a path, remove a path, or do nothing), and implement that decision. The decision is made by IOS, using the information in the DSBs, and the implementation is achieved through a combination of XCF services and the Dynamic I/O Reconfiguration capability.

In the list of criteria considered by imbalance correction, the first criteria was “Director Port Busy”. If this is a significant component of Pend time for this LCU, DCM attempts to alleviate the problem by moving one of the existing managed paths to another, currently unused (by this CPC) port on the same Switch.

For all other changes, DCM either adds a path to the LCU or removes an LCU from one of the managed paths being used by this LCU. This involves two sets of actions:

- The hardware must be requested to do a dynamic reconfiguration to add or remove the path from the configuration for that LCU.
- The operating system in every LP that is using the affected path must be requested to VARY ON or VARY OFF its path to the devices behind the LCU in question.

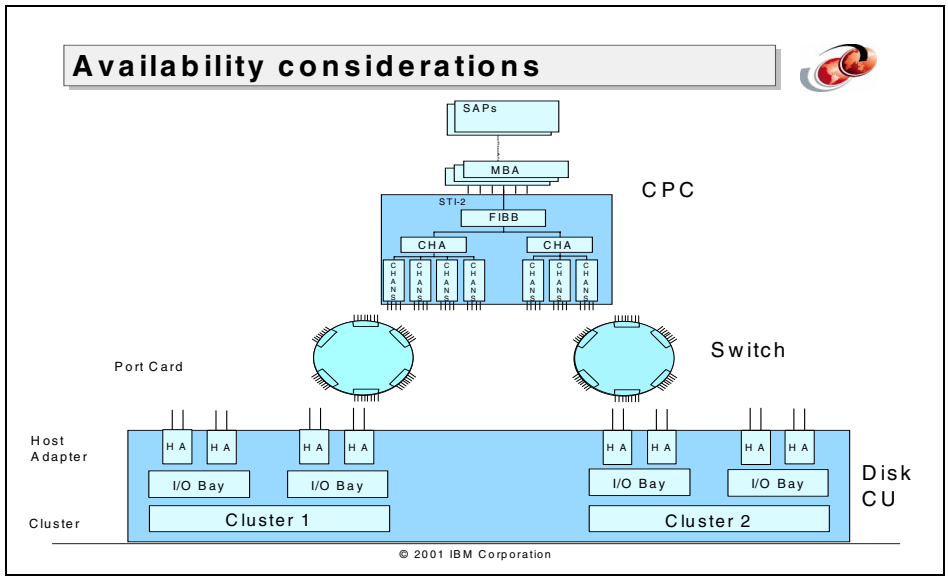
Before DCM attempts to remove a path from an LCU, it must first be sure that all the other systems in the LPAR Cluster have varied that path offline. The system that is running imbalance correction uses XCF to communicate with the other systems, indicating the required action. Those systems must then vary the paths offline and inform the initiating system of successful or unsuccessful completion of the command. If any of the systems cannot vary the path offline, the initiating system aborts the reconfiguration and informs all the other systems in the LPAR Cluster that the path should be varied online again.

The XCF group name used by DCM is SYSIOSnn. This is discussed in more detail in 10.2.2, “Other software requirements” on page 274.

Similarly, after DCM adds a path to the configuration for an LCU, all the systems in the LPAR Cluster must vary that new path online. Once again, DCM uses XCF to send a signal to the other systems in the LPAR Cluster to vary that path online. Those systems must then vary the paths online and inform the initiating system of successful or unsuccessful completion of the command. If any of the systems cannot vary the path online, the initiating system aborts the reconfiguration and informs all the other systems in the LPAR Cluster that the path should be varied offline again, after which it removes the path from the configuration for that LCU. This mechanism ensures that channels do not end up in an indeterminate state where they are neither online nor offline.

To initiate the hardware side of the change, DCM utilizes the Dynamic I/O Reconfiguration capability. After the change has been implemented, a machine check is presented to every other LP in this CPC that has access to the affected LCU.

9.9 RAS benefits



Whenever Dynamic Channel-path Management investigates adding or removing a path to an LCU, it tries to select the path with the best availability characteristics that delivers the required performance. It does this by comparing the points of failure of all potential paths with the points of failure of all existing paths to that LCU. The diagram above shows an example of the various points of failure, and the built-in redundancy, that exists in a typical S/390 or zSeries configuration.

As well as providing additional paths for performance reasons, DCM attempts to ensure that there are always at least two paths configured from a z/OS image to an LCU, and that those two paths have a minimum number (zero if possible) of common points of failure. In fact, if an LCU only has one path, DCM selects a second path based solely on the availability characteristics of that path compared to the existing path. For subsequent paths, DCM uses the availability and performance characteristics and the complexity index when deciding which path to add or remove.

Using a combination of information which is obtained from each system component whenever that component is added to the configuration, and programmed knowledge about the component, DCM is able to determine the points of failure used by each path.

For example, in a 9032-5 ESCON Director, there are 8 ports per port card. And each path through a director uses two ports: one for the connection to the CPC, and the other for the connection to the control unit. So, when DCM is deciding which channel (path) to add to a control unit, it attempts to choose one that is not using any of the port cards that are already being used by any of the existing channels to this control unit.

Depending on the component (CPC, Switch, or physical control unit), there are differing numbers of single points of failure. For DASD control units, DCM uses information from the tag field of the node descriptor and combines this with information from a new load module (IOSTmmm) to identify common points of failure within the device. IOSTmmm (where mmm identifies the manufacturer) is built and provided by your DASD vendor. The table for the IBM devices is shipped with z/OS.

Similarly, the channel subsystem in the CPC consists of a number of components, each of which are used to process an I/O request. To give an example, the channel subsystem in a IBM zSeries 900 CPC contains the following components:

- ESCON channels are attached to a Channel Card (“CHANS” in the diagram on the previous page). All critical devices (like DASD) should be connected to more than one channel. And the channels that you choose should be spread across multiple Channel Cards.
- The Channel Card is in turn associated with a Channel Driver Card (“CHA” in the diagram on the previous page). The Channel Cards connecting a given DASD LCU should be distributed over a number of Channel Driver Cards.
- The Channel Driver Card is associated with an STI (Self-Timed Interconnection). The Channel Driver Cards connecting a given DASD LCU should be spread across the available STIs.
- The STI is associated with an I/O cage. Depending on your CPC configuration, your CPC might have more than one I/O cage. In this case, you should attempt to spread the STIs associated with a given LCU across more than one I/O cage.

The IBM zSeries 900 CPC adds a new function that indicates the single points of failure among a set of CHPIDs that you provide. DCM uses this function to identify the CHPID with the fewest channel subsystem points of failure in common with the existing paths to the device. DCM takes this information, along with control unit and Switch information, into account when deciding which is the best path to add to an LCU.

There is also another aspect to the RAS benefits of DCM. If a path is lost for any reason (hardware failure, operator CF command, and so on), the I/O Velocity of any affected control units will probably drop as a result. This drop in I/O Velocity will result in DCM taking action to restore the I/O Velocity to the previous level. Therefore, DCM can be viewed as a way of automatically addressing the impact of hardware failures.

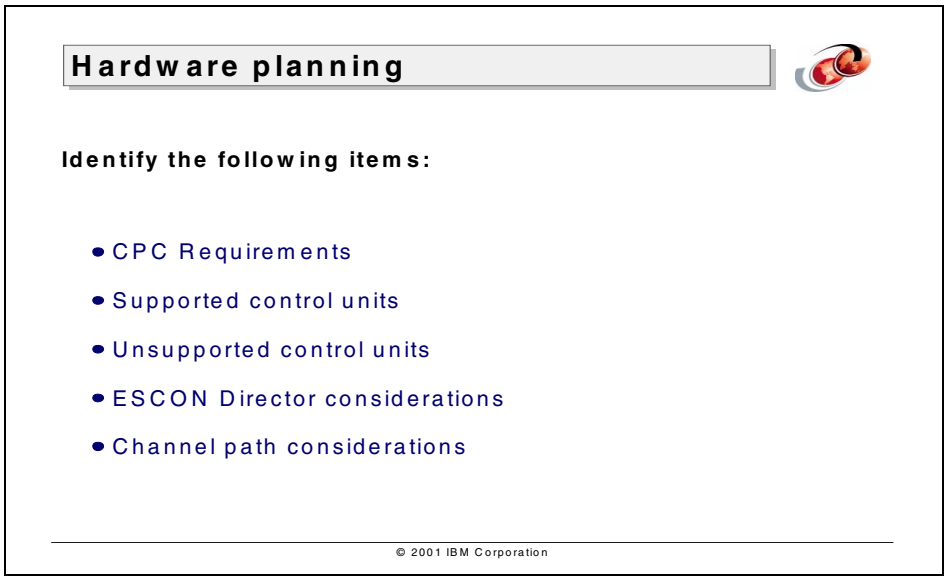
10

Planning for Dynamic Channel-path Management

As with any project, the key to a smooth and successful implementation of Dynamic Channel-path Management (DCM) is careful and comprehensive planning. For DCM in particular, actually turning it on is easy—the more time consuming part is planning for exactly how you want to use it. Since DCM can potentially have a significant impact on the channel capacity available to a given control unit, and a corresponding impact on the applications using that control unit, you must be sure that your planned configuration will be capable of delivering the performance and availability you require.

In this chapter, we discuss the hardware, software, and operational changes that must be made as part of the implementation of DCM. We also introduce a methodology for identifying channels and control units that are good candidates for DCM. The approaches you can use to start using DCM are covered in 10.6, “Migration planning” on page 295.

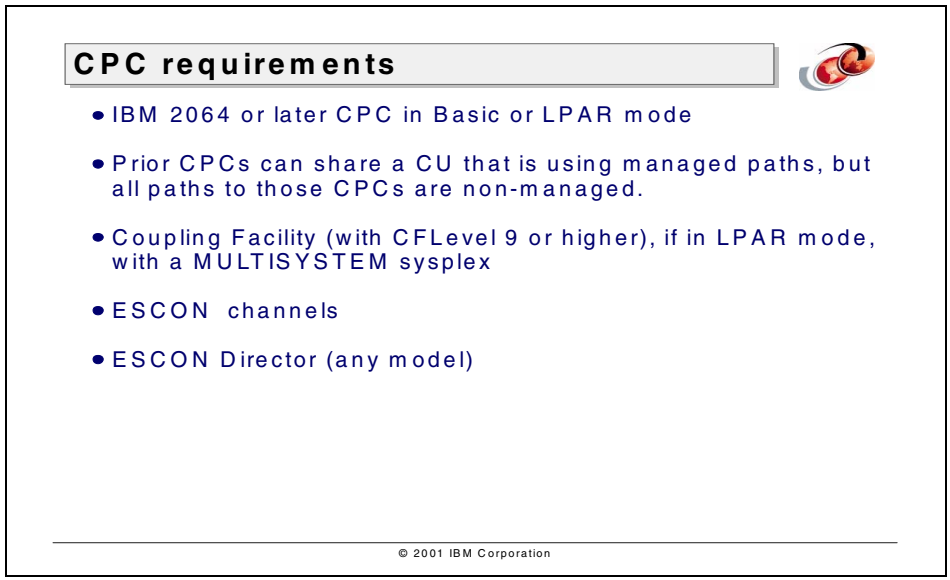
10.1 Hardware planning



In this section, we cover the hardware requirements for an environment that contains at least one system that is exploiting Dynamic Channel-path Management. During this phase, the following hardware items need to be addressed:

- The CPC requirements
- Supported control units
- Unsupported control units
- ESCON Director considerations
- Channel path considerations

10.1.1 CPC requirements



The most basic hardware requirement for implementing Dynamic Channel-path Management is that you must be running on an IBM zSeries 900 or later CPC. It *is* possible to share a control unit that is using managed paths with an earlier CPC; however, that CPC can obviously only have non-managed paths. Also, any host adaptor on the control unit can support a mixture of managed and non-managed paths. So, adapter 1 could be attached (via a switch) to a non-managed channel on an IBM 9672 G6 and a managed channel on an IBM zSeries 900 at the same time. The control unit itself has no knowledge of whether a given path is managed or non-managed.

When you define channels in HCD, HCD does not allow managed channels to be defined on a CPC prior to a IBM zSeries 900.

DCM is supported in both Basic and LPAR mode.

There should be no HMC changes required to support DCM. The only things you have to ensure are:

- That Dynamic I/O Reconfiguration support is enabled in the CPC Reset profile. This capability has existed on IBM processors since Dynamic I/O Reconfiguration was introduced in MVS/ESA 4.2.
- That the “automatic input/output (I/O) interface reset” option is enabled. This is specified on the Options tab of the CPC Reset profile.

This function allows an operating system to issue a System Reset for another LP. This would normally be used by SFM, to partition a failed system out of the sysplex. However, in a DCM environment, it is also required to free up the locks held by an LP that was in the middle of making a configuration change when it failed.

- That the LPs in the LPAR Cluster are all allowed to update the IOCDS and the current configuration. This is controlled in the “Security” tab of the LPAR Image Profile, by enabling the “Input Output Configuration Control” button.

On a related topic, you do *not* have to increase the percent that you expect the HSA to grow by, as a result of the changes made by Dynamic Channel-path Management. The reason the HSA grows for existing Dynamic I/O Reconfigurations is that you are normally adding or removing devices when you make a configuration change. However, DCM is not adding or removing any devices—all it is doing is changing the contents of existing fields in existing subchannels. As a result, DCM has no impact on the size of the HSA.

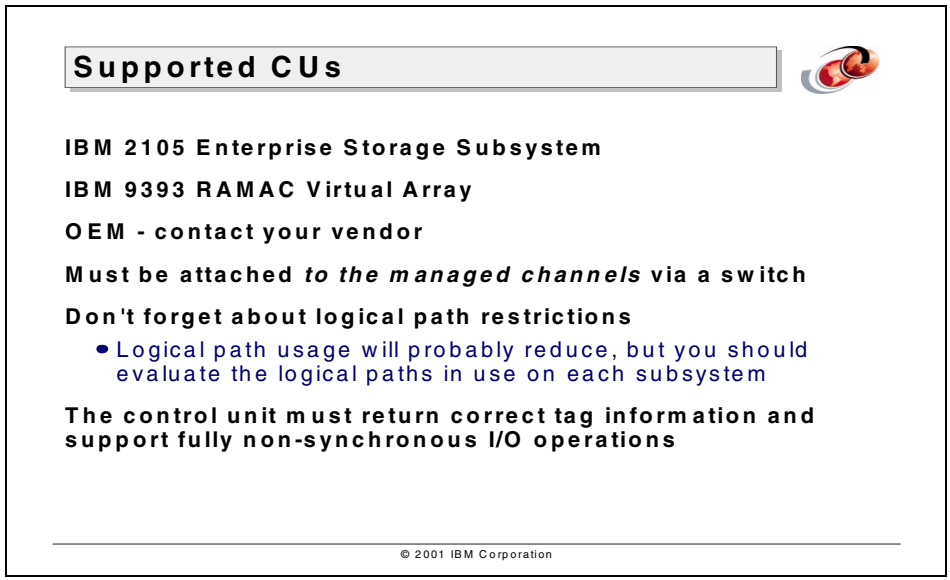
If running in Basic mode, a Coupling Facility (CF) is not needed for DCM. This is because DCM only uses the CF to share information between LPs that are in the same LPAR Cluster. As no LPAR Cluster exists on a CPC in Basic mode, there is no need for a CF.

If you are running in LPAR mode, a CF is required if you wish to use DCM in any LP containing a system that is a member of a multi-system sysplex, even if the LP is the only member of that sysplex on this CPC. The WLM structure that resides in the CF does not have any specific failure isolation requirements; however, the CF must be using CFLEVEL 9 or higher. CFLEVEL 9 is only supported on IBM 9672-Rn6 (including 9672-R06), Yn6, Xn7, and Zn7 (that is, 9672 G5 and G6), or any model of IBM zSeries 900. It is perfectly acceptable to place the structure in the same failure domain as the connected LPs. There are no specific considerations for a double failure; that is, a single failure taking out both the CF and one or more connected operating systems.

You do not need a CF if all the LPs using DCM are running in XCFLOCAL or Monoplex mode. Because these systems are not in a multisystem sysplex, there is no other LP that they will share the managed channels with, and therefore no need for the WLM structure.

As stated previously, DCM currently only works with ESCON and FICON Bridge (FCV) channels. The managed channels *must* be connected to an ESCON Director. The requirements for the director are discussed further in 10.1.4, “Switch considerations” on page 267.

10.1.2 Supported control units



To support Dynamic Channel-path Management, a DASD control unit must:

- Support fully non-synchronous I/O operations. This means that the device should *never* transfer data directly to or from the channel. Fully non-synchronous devices always transfer data to and from the channel from the cache.

  Not all ESCON-attached control units are fully non-synchronous—for example, there are some circumstances under which an IBM 3990 will operate synchronously. When a control unit operates synchronously, the transfer rate to and from the channel is much lower—in the region of 4 MB/second compared to 17 MB/second when operating non-synchronously. This varying transfer rate upsets the DCM calculations. As a result, control units that can sometimes operate synchronously are not supported by DCM.

- Return valid node descriptor information. The information that the control unit places in the tag field of the node descriptor is used to identify single points of failure. There is no pre-defined layout for the contents of this field, so it is possible that a given control unit may not provide information in the format expected by DCM. Check the IRD PSP subset of the 2064DEVICE PSP bucket for information about any IBM microcode upgrades that may be required. Also, the IOSCUMOD macro must be used to create an IOSTmmm load module for any non-IBM DASD subsystems. These modules provide control information for determining service boundaries in a control unit, to identify single points of failure in the control unit, and are created using

information provided by your DASD vendors. This is discussed in more detail in 11.4, “Building the IOSTmmm module” on page 314.

At the time of writing, the IBM DASD control units that are supported by and have been successfully tested with DCM, are:

- 2105 Enterprise Storage Subsystem with microcode level SC01208
- 9393 RAMAC Virtual Array

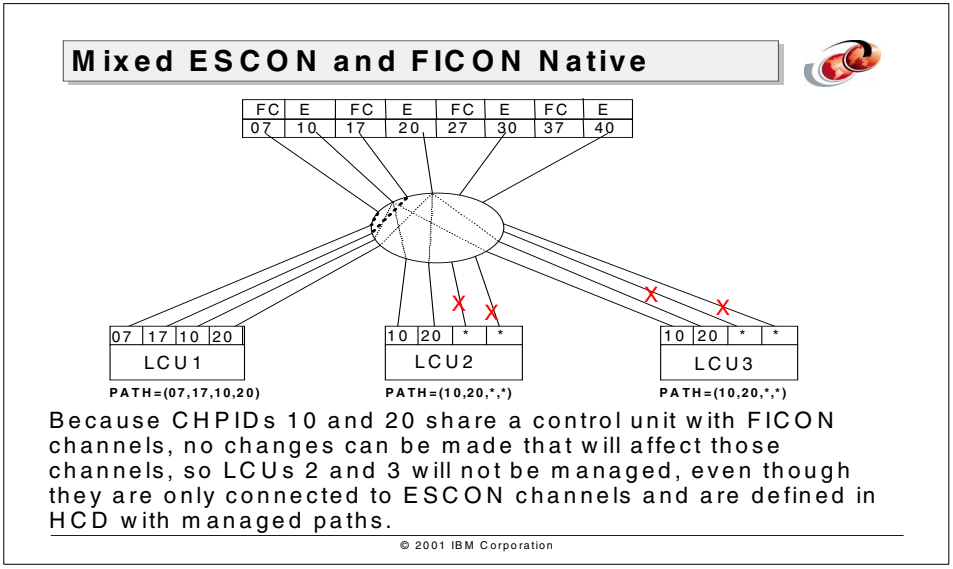
It is interesting to note that the control unit itself is unaware of the fact that some of its paths are being managed by DCM. When paths are added to, or removed from, the configuration for the control unit, it does not know if it was an operator or DCM that made the change.

All the managed paths to a control unit *must* be attached via a switch. If you wish, the non-managed paths can be connected point-to-point rather than through a switch. However, we recommend that, where possible, *all* DASD paths are routed through a switch, since this provides the optimum flexibility and connectivity.

Another thing to consider is the mixture of ESCON or FICON Bridge (FCV) and FICON Native (TYPE=FC) channels on a control unit. IBM recommend that you should only mix ESCON and FICON Native channels on a control unit during a transition period. It is not intended that such a configuration would be used on an ongoing basis.

At the time of writing, Dynamic Channel-path Management does not manage FICON Native channels. In addition, HCD will not allow you to define a control unit with both managed channels and FICON Native channels. Also, DCM will not manage a control unit that shares any channels with a control unit that has FICON Native channels, nor will it manage any affected control units.

To illustrate this, we will use the simplified configuration shown in the figure on the following page. In this case, you have a control unit (LCU1) that is defined in HCD with both non-managed ESCON and FICON Native channels. The two non-managed ESCON channels (CHPIDs 10 and 20) are also in the configuration of LCU2 and LCU3. In this case, even though LCU2 and LCU3 have been defined with managed paths, DCM will not configure any channels to use those managed paths. Remember that any time DCM makes a change to a control unit, it projects what the impact will be on all the channels attached to that control unit, *and any other control units attached to those channels*. Because DCM does not manage FICON Native channels, it cannot project the impact of any change that in any way will affect a FICON Native channel. Therefore, it cannot add paths to LCU2 or LCU3 because that change would have an impact on CHPIDs 10 and 20, which in turn would have some effect on LCU1, thereby affecting the two FICON Native channels on that LCU.



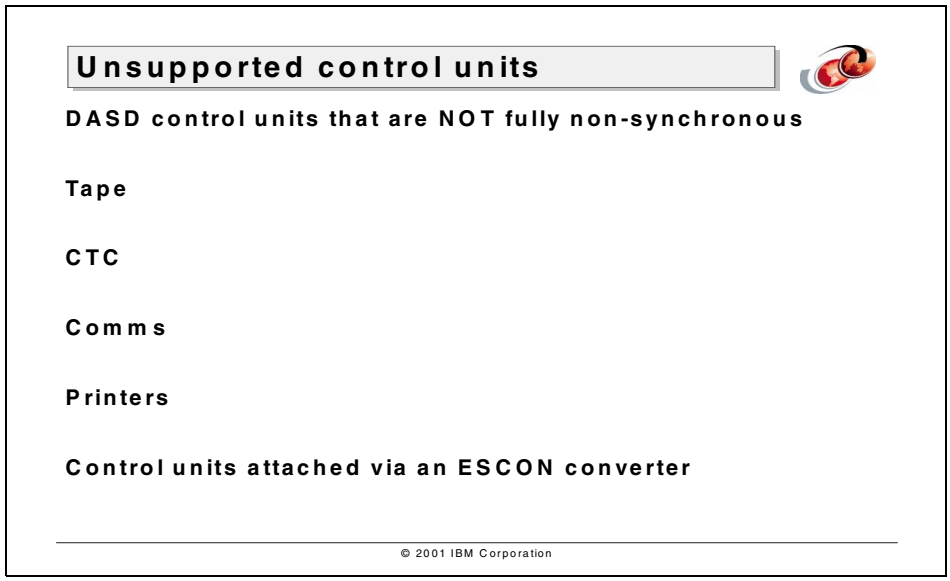
One thing that you should bear in mind is the number of logical paths in use to the control unit. This is discussed in more detail in 11.1.2, “CU definitions” on page 304; the table in 9.1.10, “Paths” on page 213 provides a list of the number of logical paths supported by the various IBM DASD control units. The introduction of DCM should not, by itself, increase the number of logical paths in use. In fact, it may even reduce the number of logical paths in use.

Consider a configuration prior to DCM: each LP that uses a given LCU probably is on the access list of all channels that connect to that LCU. So, if you have four LPs and an LCU with eight paths, that totals 32 logical paths. If you introduce DCM, and move to 4 non-managed and 4 managed paths, and 2 of the 4 LPs are in the LPAR Cluster, the number of logical paths would now be 16 for the non-managed paths (4 paths by 4 LPs) plus a maximum of 8 for the managed paths (4 paths by 2 LPs), giving a total of 24 logical paths compared to 32 previously.

We recommend that you identify the number of logical paths in use before you introduce DCM, and calculate the number that is in use after you enable DCM, just to be sure that you do not exceed the number of supported logical paths to each control unit. Remember that this must be done for each control unit, and that different control unit types support differing numbers of logical paths. The ICKDSF ANALYZE NODRIVE NOSCAN command can be used to display logical path information for a controller. See the *DSF User’s Guide and Reference* for more information.

At the time of writing, we do not have a comprehensive list of non-IBM control units that support DCM. You should contact your vendors to determine the control units that support Dynamic Channel-path Management, and what microcode levels are required.

10.1.3 Unsupported CUs



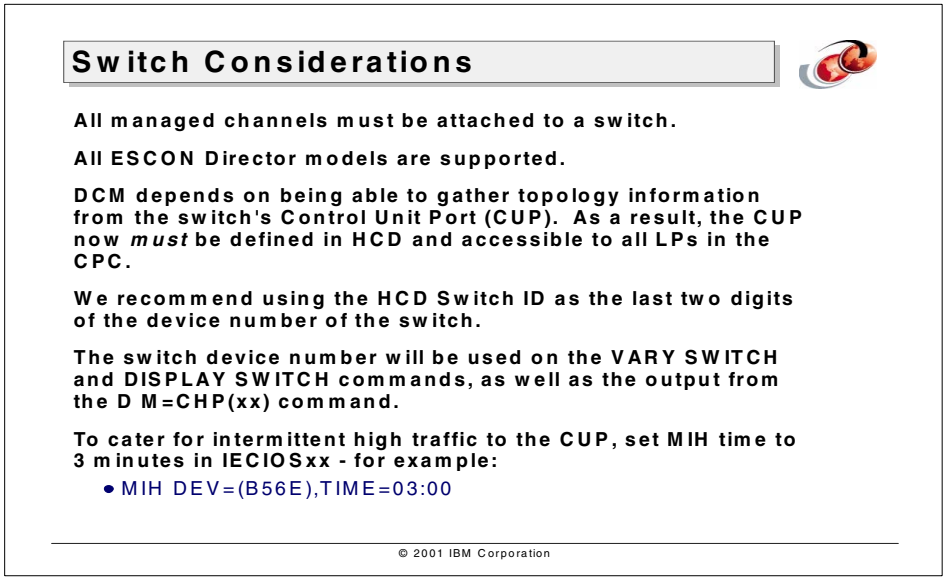
At this time, the only control units supported by Dynamic Channel-path Management are those used to attach the more recent DASD devices. There are a number of reasons for this, including:

- DASD tends to be more response time-sensitive than most other device types.
- Modern DASD control units can potentially deliver huge volumes of data, so very large channel bandwidths are required. However, the same control units can have extended periods with relatively little activity, thereby tying up valuable channel capacity during these idle periods.
- DASD control units support multiple paths. The whole concept of Dynamic Channel-path Management—moving paths to where they are needed—only works with control units that support concurrent attachment to multiple channels.
- The DCM algorithms depend on the attached subsystems being *fully* non-synchronous. Earlier DASD subsystems, even the ESCON ones (like the 3990) were largely, but not completely, non-synchronous. This is the reason why earlier DASD subsystems are not supported by DCM.

While tape devices support multiple channels, the impact of insufficient channel capacity for a tape drive is generally not as severe as lack of channel capacity for a DASD device.

Control units that are attached via an ESCON Converter cannot be attached via a shared channel, and thus cannot be used with Dynamic Channel-path Management, since all channels connected to a managed control unit must be shared.

10.1.4 Switch considerations



As we have said before, all managed channels *must* be defined as shared and must be attached to a switch. All models of the ESCON Director are supported. The following table contains the required microcode levels for each of the ESCON Director models:

ESCON Director Model	Minimum Required microcode level
9032-2	Version 4 Release 1
9032-3	LIC 04.03.00
9032-5	LIC 05.04.00

Before we get into discussing how to manage the switch, it is important to understand that there is no such thing as a “managed port” on the ESCON Director. There are ports that will be used by a static path. However, it is also possible, even likely, that those same ports will also be used by managed paths. *Any* port that is connected to a control unit that is supported by Dynamic Channel-path Management, and has been defined in HCD as being a managed control unit, is potentially usable by Dynamic Channel-path Management. It is important to remember this when we discuss the new D M=SWITCH command, and whether a specific port is “DCM Allowed” or not in 12.1.2, “D M=SWITCH command” on page 320.

In order for DCM to gather the topology information it needs, it is *required* that the Control Unit Port (CUP) on every Switch that is attached to managed channels is defined to HCD. This is a new requirement in z/OS; in prior levels of the operating system it was recommended that you define the CUP, but it was not an absolute requirement. As described in 9.4, “Initialization changes” on page 227, DCM communicates with the CUP on each director to get the connectivity topology and port status for all the ports on the director. If the CUP is not defined in HCD, and accessible to every LPAR in the LPAR Cluster, this information cannot be obtained.

It is a good idea, when defining the control unit and device for the director in HCD, to use the Switch ID as the last two digits of the device number. For example, if the Switch ID is 6E, the device number might be B56E. This makes it easier to relate the device number back to the Switch definition in HCD.

Another thing you should do in relation to the directors is set the MIH time for the director devices to 3 minutes. All of the OS/390 V2R10 or higher systems that are connected to a director will check with the CUP on a regular basis to see if there have been any changes in the director configuration (for example, if a port has been blocked). This can lead to elongated response times from the CUP if there are a large number of connected systems. Therefore, it is best to increase the MIH setting in the IECIOSxx member as shown in the previous figure. The device numbers of the director must be specified since there is no MIH class for Switch devices.

There are two new commands provided in z/OS V1R1 and later that are related to managing switches. The D M=SWITCH command displays information about what is connected to each switch port, and whether a given port can be used by DCM or not. The VARY SWITCH command allows or disallows Dynamic Channel-path Management from using a specified port. These commands are discussed in more detail in 12.1.2, “D M=SWITCH command” on page 320 and 12.1.7, “VARY SWITCH command” on page 328 respectively.

If a port is Prohibited (that is, you have told the ESCON Director that port xx cannot communicate with port yy) or if a port is Blocked (no port can communicate with this one), then DCM will understand this, and not try to use that port.

There is nowhere in HCD that you can actually save information about whether you want Dynamic Channel-path Management to be able to use a specific port on the ESCON Director. We recommend that you define the Switch and Port that *every* adaptor port on a managed control unit is attached to on the first panel of the “Add a Control Unit” panel in HCD. If you do this, when you use HCD (option 2.6.6) to create a CONFIGxx member, it will present you with a complete matrix of every port on every Director that DCM could use. At this time you can specify which ports you do not want DCM to use. The result of this specification is a

CONFIGxx member that contains a set of SWITCH statements that specify the desired DCM state of each Director port. The default is for all the ports connected to managed control units to be DCM-enabled. If you wish to stop DCM from using some of those ports, you must either issue V SWITCH commands or use the CONFIGxx member with the CONFIG MEMBER(xx) command to change the port status. This information is not saved anywhere in HCD: you must re-enter it each time you create a new CONFIGxx member.

Caution with use of CONFIGxx member

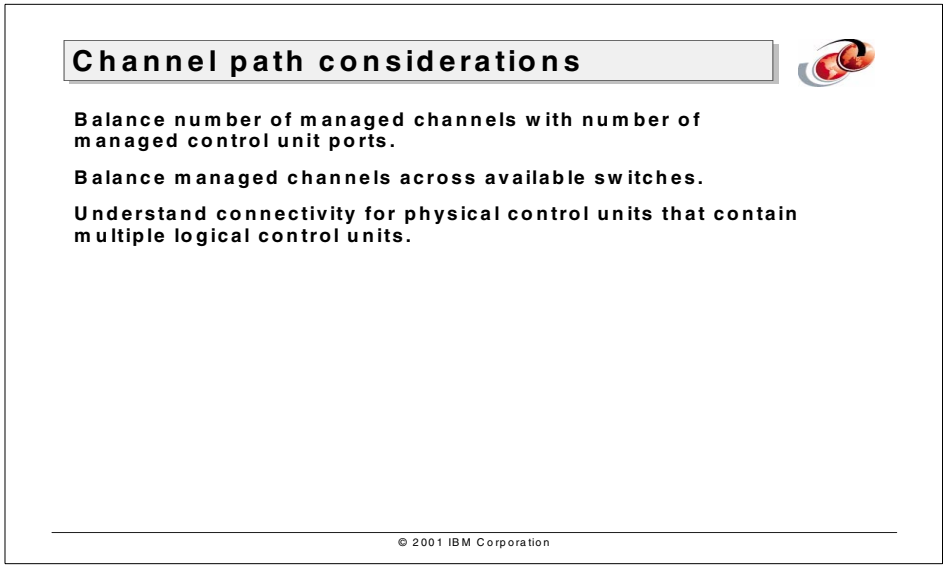
You must use the CONFIG MEMBER(xx) command with care. This command is designed to automatically adjust your configuration to bring it into line with the model configuration you specified in the CONFIGxx member.

If you have made any changes to the configuration since the CONFIGxx member was built, for example, if you took a CHPID offline for service, issuing the CONFIG MEMBER(xx) command will reset the configuration to match the definition in the CONFIGxx member.

You should never issue the CONFIG MEMBER(xx) command without first issuing a D M=CONFIG(xx) command to find out what changes will be made when you issue the CONFIG MEMBER(xx) command, and confirming that no unexpected changes will be made.

When the first system in the LPAR Cluster is IPLed, this is not an issue. However, if one system is IPLed while the other systems in the LPAR Cluster are already running, and you have made configuration changes in the interim, issuing the CONFIG MEMBER(xx) command will reset those changes and this could potentially result in performance or availability problems.

10.1.5 Channel path considerations



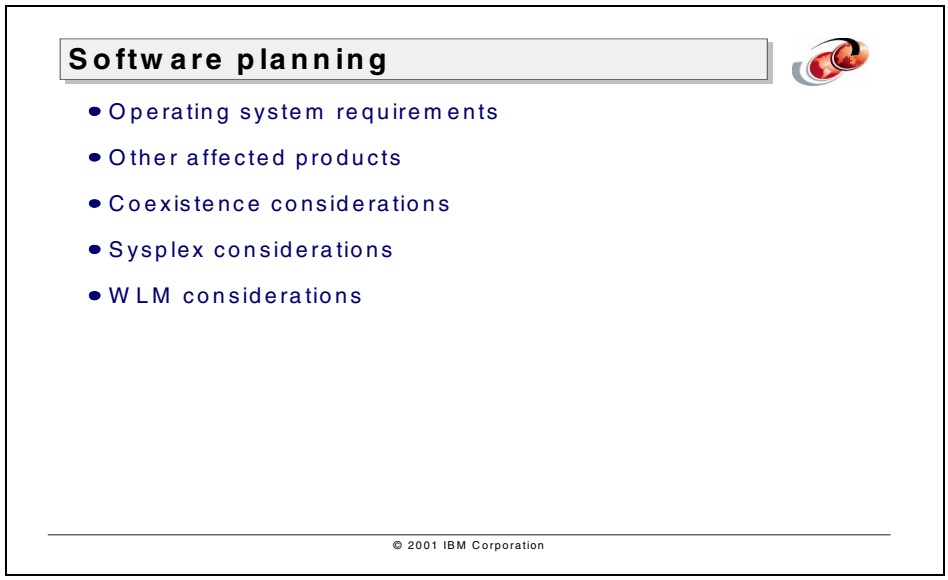
When you are setting up your configuration, you should attempt to have a balance between the number of Host Adaptor Ports on the managed control units, and the number of managed channels connected to the Switches. There is little point in providing 8 managed channels but only having one managed control unit that only has a total of 4 Host Adaptor Ports. You would potentially end up with two channels connected to each port, with minimal performance improvement.

When deciding which channels should be managed, you should also attempt to spread the managed channels across as many Switches as possible. This gives DCM the ability to select paths that will provide the best possible availability.

Similarly, you should understand how paths are used for a physical control unit that contains multiple logical control units. For example, you can have up to 32 Host Adaptor Ports for S/390 use in an IBM 2105. You can also have up to 16 LCUs for S/390 use per 2105. Multiple LCUs are accessed through a single Host Adaptor Port. However, because you have a Switch as part of the channel path for each managed LCU, it is possible that LCU 1 will be accessed via managed CHPID 10, using port 18 on Switch 01, connected to Host Adaptor Port 01. LCU 3 in the same physical 2105, might be accessed via managed CHPID 20, using the same port on the switch, and obviously also connected to Host Adaptor Port 01. So, even though we have a single connection between the control unit and the Switch, we actually have two managed channels using that connection. You

would have many more than one Host Adaptor Port on the 2105, of course, but this shows how each might be configured. Whenever DCM is allocating an additional path to an LCU in a multi-LCU control unit, it evaluates all the options, considering availability, performance, and complexity, so it is possible that two LCUs in the 2105 might be configured on the same managed CHPID, or they might be configured on different managed CHPIDs. To provide a configuration with the optimum performance and availability, the non-managed paths should be set up to follow the normal configuration guidelines for the control unit (that is, one set of channels should be used for the even-numbered LCUs, and a different set should be used for the odd-numbered LCUs).

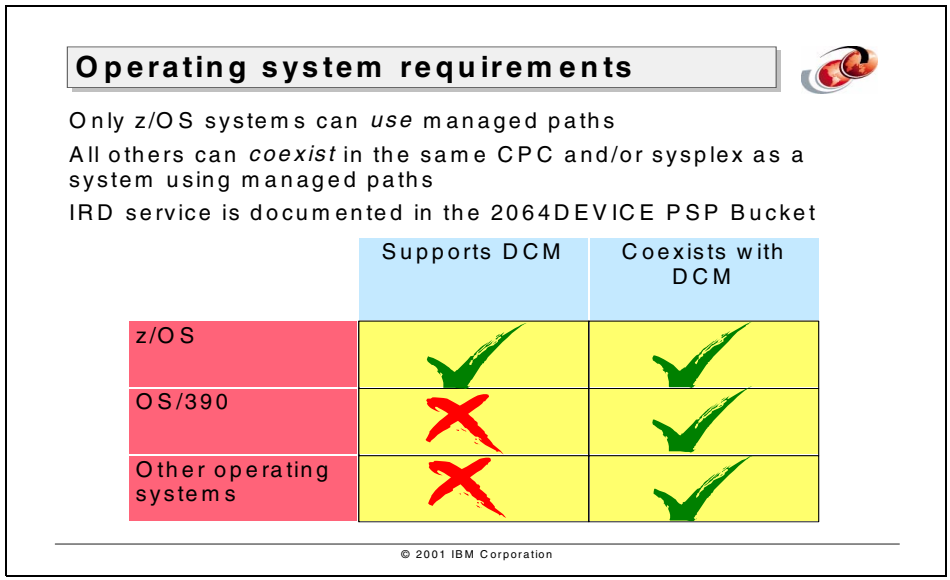
10.2 Software planning



In this section, we discuss the planning requirements for software products that provide support for, or coexist in, a Dynamic Channel-path Management environment. This includes:

- Operating system requirements
- Other affected products
- Coexistence considerations
- Sysplex considerations
- WLM considerations

10.2.1 Operating system requirements



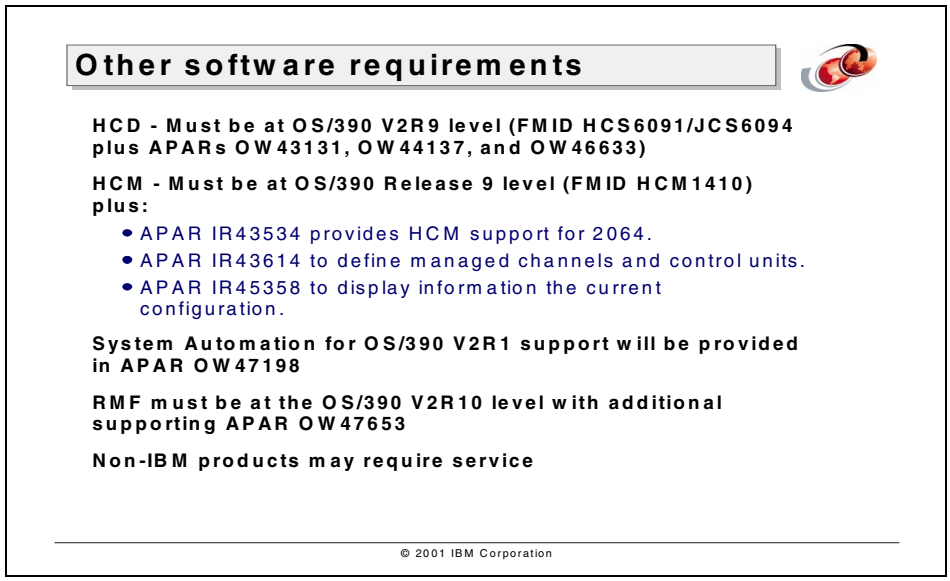
The only operating system that can *use* a managed path is z/OS 1.1 or higher (in z/Architecture mode). There is some support contained in OS/390 V2R10 that allows you, for example, to use some of the new DCM-related display commands; however, the OS/390 V2R10 system itself cannot use a managed path.

You should review the IRD subset of the 2064DEVICE PSP bucket for the latest service information. Because DCM consists of code in both the IOS and WLM components of z/OS, there is service to both of these components. Also, the WLM Web site contains planning and implementation for IRD at the following URL:

`http://www.ibm.com/servers/eserver/zseries/zos/wlm/documents/ird/ird.html`

As we stated previously, the control units themselves are not aware of the existence of DCM. The ability to return tag information existed prior to DCM, so this is not a new DCM requirement. Therefore, there are no software pre- or co-requisites to share a control unit that has managed paths to another LP.

10.2.2 Other software requirements



Because Dynamic Channel-path Management is basically responsible for adding and removing paths to control units, the only products that are impacted by its presence are those that are used to define, monitor, or control the configuration.

APAR OW44137 added support to the OS/390 V2R9 level of HCD to allow you to define a Dynamic Channel-path Management environment. This support allows you to prepare an IODF in advance of the installation of an IBM zSeries 900 CPC, and also allows you to continue to maintain an IODF that contains Dynamic Channel-path Management definitions without having to do that work on a z/OS system. To create an IBM zSeries 900 IOCDS, you must use the IYPIOCP version of the IOCP program, delivered by APAR OW43131. In addition, to define managed channels and control units, you need HCD APAR OW44137. If you are using the I/O Operations component of System Automation for OS/390, you should also apply HCD APAR OW49738.

The support for DCM in HCM consists of three separate enhancements. The first of these, APAR IR43534, provides HCM support for the IBM zSeries 900 range of processors. The next is provided by APAR IR43614. This APAR provides the ability to use HCM to define managed channels and control units. The other enhancement, provided by APAR IR45358, is the ability to show information about the managed channels and control units in the current configuration. This support is especially beneficial when doing I/O problem diagnosis in a DCM environment.

System Automation for OS/390 V2R1 (SAFOS) is being updated to indicate whether a particular channel or port is managed in the appropriate messages. It also includes changes so that, if you want to take a managed path offline, it will issue a VARY SWITCH command instead of a VARY PATH command. This support is delivered in APAR OW47198. If you do not have this APAR applied, attempts to bring a managed path offline using SAFOS will fail. Similarly, if you use SAFOS to block or prohibit a port, it will issue a VARY PATH rather than a VARY SWITCH for the managed paths. It will still go ahead with the switch change, but the path will not be taken offline in advance of the change. This may result in the console being flooded with messages—a condition that is resolved by SAFOS APAR OW48237. Once you have the support, the ideal situation is to use SAFOS to manage the varying online and offline of all channel paths: SAFOS will automatically figure out the correct command to use, eliminating the need for you to use different commands depending on whether the path is a managed path or not.

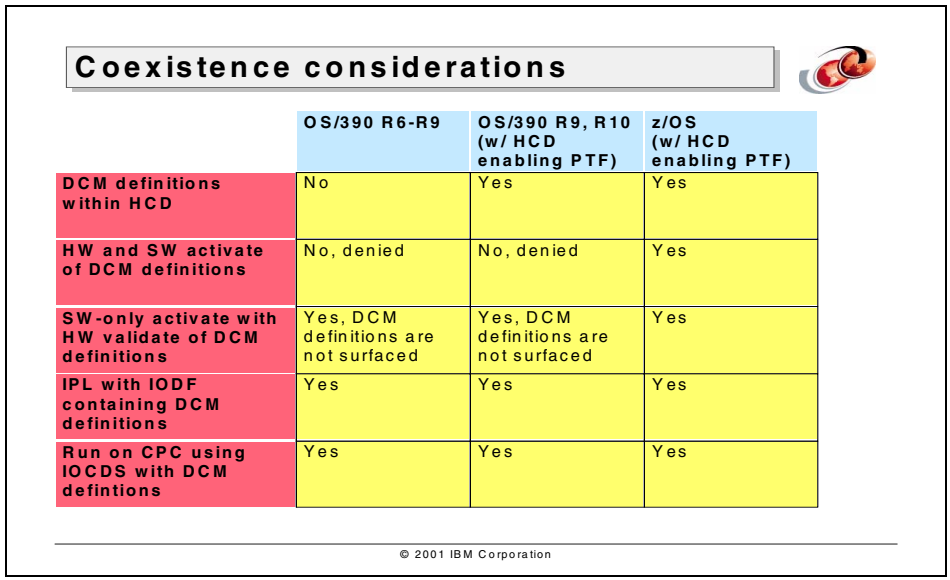
Support has also been added to RMF. To get this support, the OS/390 V2R10 level of RMF, together with APARs OW47653 and OW48950, is required. The specific changes to RMF are described in 13.1, “RMF considerations” on page 342.

It is possible that non-IBM products may need specific support. You should discuss this with the vendors of any products that have knowledge of the underlying hardware configuration.

At the time of writing, no other IBM products have been encountered that require specific support for DCM. Most products have no knowledge of the underlying configuration—and if they do, the changes being made by DCM look just like there is an operator constantly issuing ACTIVATE commands to implement HCD changes.

As time goes on, other IRD-related service may become available. Refer to the IRD subset of the 2064DEVICE PSP Bucket for the latest information.

10.2.3 Coexistence considerations



While it is necessary to be running z/OS on a IBM zSeries 900 or later CPC to exploit Dynamic Channel-path Management, there will almost certainly be a period during which older versions of both the CPC and the operating system are coexisting in the sysplex.

In order to handle this situation, you need to be able to do the following:

- Create an IODF containing DCM definitions from a level of the operating system prior to z/OS.
- Run levels of the operating system prior to z/OS on a CPC whose IOCDS contains DCM definitions.
- Run levels of the operating system prior to z/OS using an IODF that contains DCM definitions.
- Perform software and hardware activates using an IODF that contains DCM definitions.

As noted in 10.2.2, “Other software requirements” on page 274, it is possible to create and maintain an IODF containing DCM definitions using the level of HCD that is provided with V2R9 or later, plus the supporting APARs.

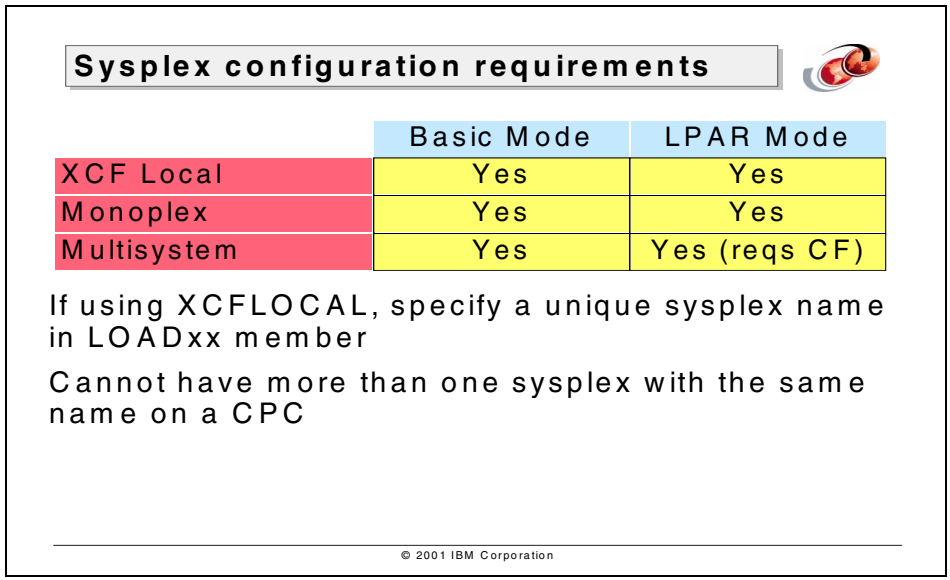
The contents of the IOCDS are not used by the operating system, so it is possible for an operating system with no DCM support to run on a CPC whose IOCDS contains managed channel and control unit definitions. Remember that only OS/390 V2R6 and later is supported on an IBM zSeries 900 CPC.

Similarly, it is possible to run releases of OS/390 as far back as V2R6 with an IODF that contains DCM definitions, however the Dynamic Channel-path Management-specific definitions are not visible to those operating systems.

Regarding software and hardware Activates of a new configuration using an IODF that contains DCM definitions, the software Activates can be done on OS/390 V2R6 or higher. However, the hardware Activate for an IBM zSeries 900 that is using managed channels can only be done on z/OS 1.1 or higher. This is summarized in the table on the previous page.

Dynamic I/O reconfiguration, and how it relates to DCM, is discussed in more detail in 12.4, “Dynamic I/O reconfiguration” on page 339.

10.2.4 Sysplex configuration requirements



Unlike WLM LPAR CPU Management, it is not actually necessary for a system to be part of a sysplex to take advantage of Dynamic Channel-path Management.

If the CPC is in Basic mode, the system can be in XCFLOCAL, Monoplex, or Multisystem sysplex mode. As there is no other LP to share the managed channels with, there is no need for any information about the channels to be shared with another system, so there is no need for a WLM cluster structure.

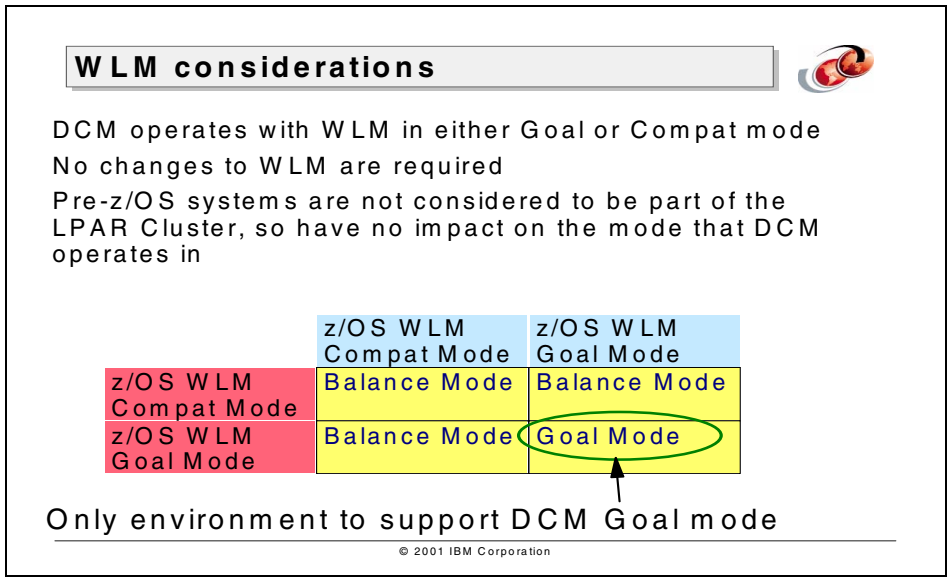
If the system is in LPAR mode, it is also possible to use Dynamic Channel-path Management in either XCFLOCAL, Monoplex, or Multisystem modes. It is only the system or systems that are in Multisystem mode that require a CF. Systems in XCFLOCAL or Monoplex mode can be using Dynamic Channel-path Management without requiring a CF.

Be aware that, on a given CPC, all the systems that have the same sysplex name *must* be in the same sysplex. This may sound obvious, but some customers actually run multiple sysplexes with the same name. Now, remember how PR/SM decides who can share a managed channel—it is based on the I/O Cluster specified on the channel definition in HCD, and the sysplex name passed to PR/SM by the operating system when it is IPLing. If you have two sysplexes with the same name, PR/SM will understandably think that all the systems with the same sysplex name are in the same sysplex, and give those systems access to all the managed channels that have been defined for use by that sysplex. In this case, an LP will be able to change the configuration of a channel, without being

able to communicate with all the LPs that are potentially using that channel. As you can imagine, this will lead to some confusion, at a minimum! Also, remember that you do not specify the scope of the LPAR Cluster anywhere: it is dynamically determined as systems IPL and pass their sysplex name to PR/SM.

A related consideration is if your system is in XCFLOCAL mode. If you do *not* specify a sysplex name, the system uses a default of LOCAL, and it is conceivable that two systems on the same CPC could both identify themselves to PR/SM as being in a sysplex called LOCAL, and both try to use the one set of managed channels. To avoid any chance of this happening, we *strongly* recommend explicitly specifying a unique sysplex name (other than LOCAL!) in the LOADxx member for *all* systems, even if they are not in an LPAR Cluster at this time.

10.2.5 WLM considerations



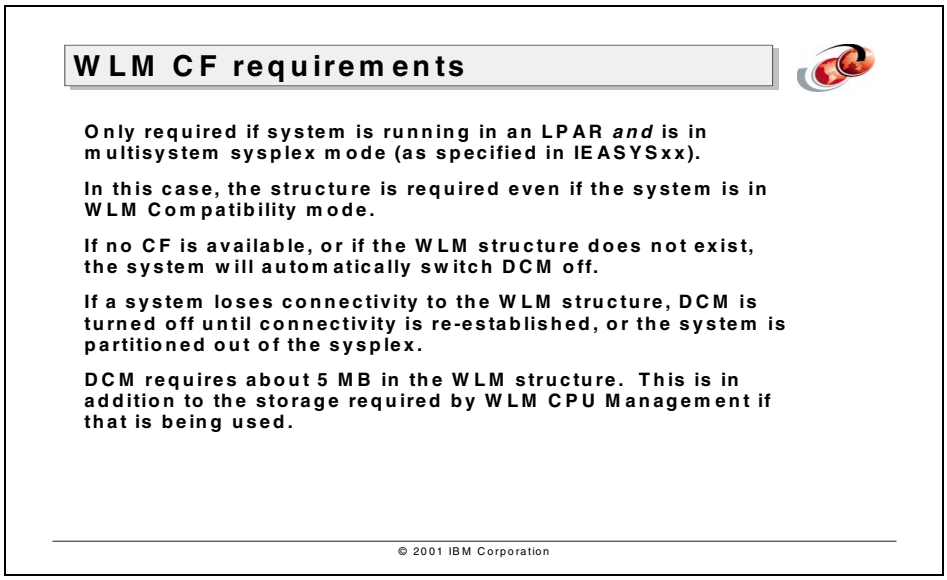
Dynamic Channel-path Management is supported in both WLM Compatibility and Goal modes. In either case, there are no changes required to WLM to enable Dynamic Channel-path Management.

If *any* of the systems in the LPAR Cluster are in WLM Compatibility mode, then Dynamic Channel-path Management *in that LPAR Cluster* operates in Balance mode. It is important to note here that the scope is LPAR Cluster-level, *not* sysplex-level. From the perspective of DCM, it is irrelevant what mode WLM is operating in on any of the systems outside the LPAR Cluster.

To operate in DCM Goal mode, *all* the systems in the LPAR Cluster must be running in WLM Goal mode. The response to the D IOS,DCM command tells you if Dynamic Channel-path Management is operating in Goal or Balance mode.

If all the systems in the LPAR Cluster are in WLM Goal mode, DCM will also operate in Goal mode. As a result, some control units may have explicit velocity targets set because they are impacting high-importance workloads. If you switch to WLM Compatibility mode, all those explicit targets will be removed, and the control units will revert to the default velocity target. As a result, you may find that some channels will be moved away from the control units that have explicit targets, potentially impacting the performance of the workloads using those control units.

10.3 DCM Coupling Facility requirements



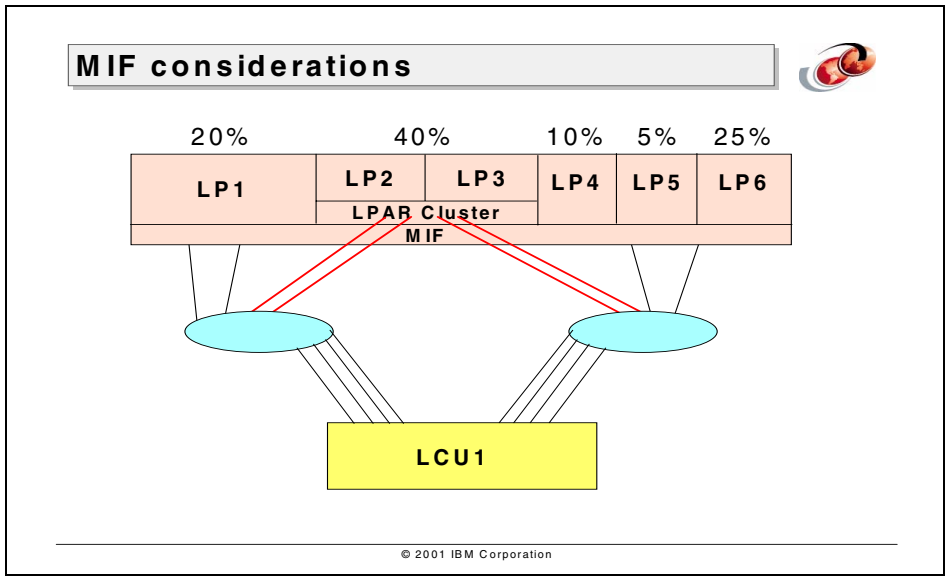
Dynamic Channel-path Management only requires a CF structure for WLM if the system using DCM is in multisystem sysplex mode (as defined in the IEASYSxx member), and is running in an LP. In this case, a structure is required regardless of whether WLM is in Compatibility or Goal mode. DCM uses the same structure as WLM LPAR CPU Management, as discussed in 3.12, “Use of CF structures” on page 96. If the system is running on a CPC in Basic mode, no CF is required.

If a WLM structure is required but does not exist, or there is no CF, DCM is automatically turned off in that LPAR Cluster. Similarly, if connectivity to the WLM structure is lost from any system in the LPAR Cluster, Dynamic Channel-path Management is turned off until the system in question is partitioned out of the sysplex or it reestablishes connectivity to the WLM structure. This is because all the systems in the LPAR Cluster share the managed paths, so DCM cannot make any change to the configuration if it can’t see what the impact is going to be on every system in the LPAR Cluster. This is different from WLM LPAR CPU Management, where the remaining systems in the LPAR Cluster can continue swapping weights even if one of the systems in the LPAR Cluster loses connectivity to the CF.

Rather than repeating the information here, the setup and prerequisites for the WLM CF structure can be found in 4.7, “Coupling Facility prerequisites” on page 115.

The amount of space required for DCM in the WLM structure depends on the number of LPs in the LPAR Cluster, the number of control units, and a number of other factors. Rather than trying to calculate a precise structure size, it is recommended to use an INITSIZE of 6144 KB, and enable Auto Alter for this structure. This size should be sufficient to handle the structures for both DCM *and* WLM LPAR Weight Management, if you are using that function.

10.4 MIF considerations



Before we move on to discussing which control units are good candidates for Dynamic Channel-path Management, it is important to consider the difference between managed channels, and shared channels as they existed prior to the introduction of DCM. This could have an impact on your plans for DCM, so we strongly recommend that you read this section.

Prior to DCM, any shared channel could be used by any LP that is in the candidate list for that channel. So, if there are 6 LPs sharing a set of channels to a given control unit, at any one point in time, any one of those LPs could be using between zero and 100% of the capacity of all the shared channels.

In a DCM environment, only the non-managed channels can be shared by all LPs. Managed channels can only be shared by LPs that are in the same LPAR Cluster. For example, in a configuration like the one above, the systems in the LPAR Cluster (LP2 and LP3) can use *all* the defined paths to the LCU—both the managed and non-managed—so those systems should have plenty of bandwidth available to them. However, the LPs *outside* the LPAR Cluster cannot share the managed paths with the LPs *inside* the LPAR Cluster, so those LPs will see a reduction in their bandwidth to the shared controllers unless you take some action to address this.

Remember that the scope of the LPs that can share managed channels is the LPAR Cluster, *not* the sysplex. This point is especially important when you are in the process of migrating from OS/390 to z/OS. At this time, you may have multiple systems from the same sysplex on the same CPC; however, they cannot all share the managed channels because the OS/390 systems are not in the LPAR Cluster. It is important to remind yourself of this point when reviewing which LPs can and cannot share the managed channels.

There are a variety of approaches you can take if you particularly wish to use DCM with a control unit that is used by systems inside and outside the LPAR Cluster:

- The first is to move some of the data to a different control unit, or at least a different logical control unit within the same physical DASD subsystem. As a generalization, you should not share volumes between systems that are not in the same sysplex. So, you may have some of the volumes being used by one sysplex and other volumes being used by a different sysplex. In this case, you might be able to move the volumes so that you end up with logical control units where most of the work comes from just one sysplex.
- Another approach is to specify enough non-managed channels to give all the systems outside the LPAR Cluster sufficient channel bandwidth. However if more than half the load on the controller comes from outside the LPAR Cluster, you would probably end up only being able to use a small number of managed channels for this control unit, and therefore would lose some of the benefits of using managed channels.
- A third approach, if the bulk of the load comes from LPs in just two sysplexes, is to set up your HCD so that each LPAR Cluster would have access to two non-managed channels, and up to six managed channels. However, you can only have a total of six managed channels to this LCU from *all* LPs on the CPC. You would achieve this by specifying:

  ```
  CNTLUNIT CUNUMBR=1000,PATH=(10,20,**,**,**,**,**,**)
  ```

  In this case, the managed paths are handed out on a first come, first served basis. If LPAR Cluster 1 were to quickly add 6 managed channels to the configuration for this LCU, LPAR Cluster 2 would not be able to add any managed channels to this LCU until LPAR Cluster 1 released one of its paths.

  It is important to realize that the number of managed paths that you specify for the control unit is the number of managed paths that can be used by *each LPAR Cluster*, not the total number of managed paths for all LPs on the CPC. So, if you specified two non-managed paths and 3 managed paths, and had two LPAR Clusters, you could potentially have 8 paths to the device - the 2 non-managed ones, 3 from the first LPAR Cluster and 3 from the second LPAR Cluster.

- The final alternative is to utilize the duplicate device number support. This allows you to have more than 8 paths from one CPC to a control unit. You achieve this by defining the control unit twice in HCD, specifying the same device numbers, but making sure that there are no paths in common between the two definitions.

  For example, assume you currently have two control units: LCU1000 and LCU2000. LCU1000 has 8 paths that are currently shared between LP1 and LP2 using MIF. Similarly, LCU2000 has a different set of 8 paths that are shared between LP1 and LP2 using MIF. These are defined as:

```
CHPID PATH=(00,08,10,18,20,28,30,38,40,48,50,58,60,68,70,78),
      SHARED,PARTITION=(A1,A2)
CNTLUNIT CUNUMBR=1000,PATH=(08,18,28,38,48,58,68,78)
CNTLUNIT CUNUMBR=2000,PATH=(00,10,20,30,40,50,60,70)
```

  Now, assume that you want to use these control units with Dynamic Channel-path Management, and each LP requires up to 6 paths at one time. Using Duplicate Device support, you would add a second control unit definition for each logical control unit, as follows:

```
CHPID PATH=(00,10),SHARED,PARTITION=(A1)
CHPID PATH=(08,18),SHARED,PARTITION=(A2)
CHPID PATH=(20,30,40,50,60,70,80,90),OS=01,IOCLUSTER=LP1
CHPID PATH=(28,38,48,58,68,78,88,98),OS=01,IOCLUSTER=LP2
CNTLUNIT CUNUMBR=1000,PATH=(08,18,**,**,**,**)
CNTLUNIT CUNUMBR=1001,PATH=(00,10,**,**,**,**)
CNTLUNIT CUNUMBR=2000,PATH=(00,10,**,**,**,**)
CNTLUNIT CUNUMBR=2001,PATH=(08,18,**,**,**,**)
```

  What had been LCU1000 is now defined twice, as LCU1000 and LCU1001. Because none of the paths used to access the control unit are shared between the LPs, it is valid to define the control unit twice as we have done above. The total number of channels being used to connect the two control units is now 20, instead of 16 as in the original configuration; however, remember that the managed channels can be moved to other control units when they are not needed on this one.

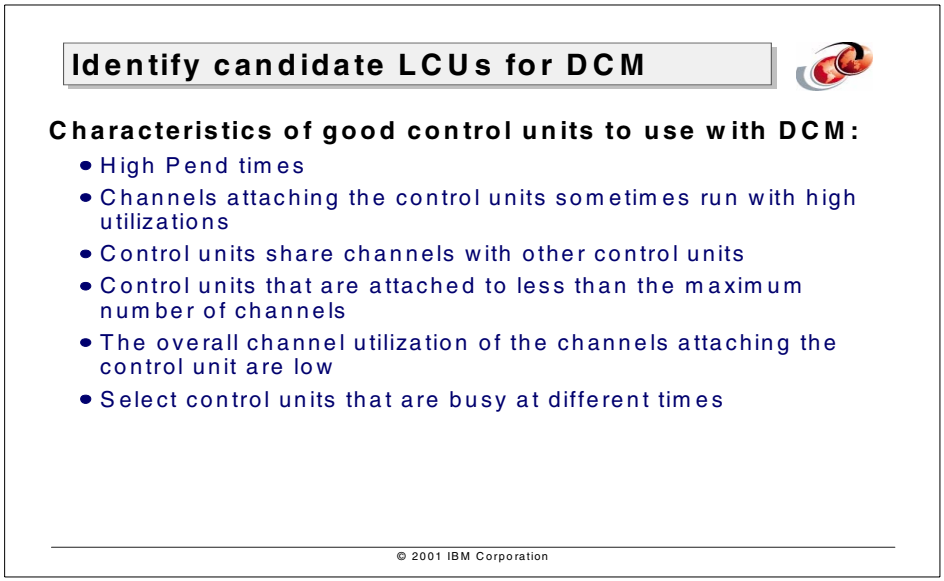
  While this configuration is a bit more tedious to set up than the original one, it does allow you to use Dynamic Channel-path Management with control units that are heavily used by more than one sysplex.

  If you *do* decide to implement a configuration like this, you should be aware that you are now tied to IPLing these systems in these specific logical partitions. Normally when you are using Dynamic Channel-path Management, you are free to IPL in any logical partition, because it is the I/O Cluster name, rather than the logical partition name, that is used to determine

who can use the managed channels, and the non-managed channels are shared by all logical partitions. Here the non-managed channels cannot be shared across LPAR Clusters because of the restrictions imposed by duplicate device support.

Which of these approaches you select will depend on your environment and the usage pattern for each control unit.

10.5 Identifying candidate control units



Now that we have covered all the hardware and software requirements and pre-requisites, the next task is to identify control units that are good candidates for Dynamic Channel-path Management. Of all the tasks involved in planning for DCM, this is the most time consuming, and has the biggest impact on how much benefit you see from using DCM.

Just to clarify, when we use the term *control unit* here, we mean *logical control unit*. The level at which you specify that you wish to use Dynamic Channel-path Management is the logical control unit, that is, the control unit as you defined it in HCD. Even though modern control units typically contain multiple logical control units within a single physical control unit, each logical control unit is defined separately in HCD, and could potentially be accessed by a separate set of channels. As a result, it is possible to have a mixture of managed and non-managed logical control units within the one physical control unit.

The section 8.6, “Environments most likely to benefit” on page 187 summarizes the characteristics that you are looking for to identify good candidate control units.

Most control units get some level of benefit from the use of managed paths, assuming that you have configured the control unit with the appropriate number of managed and non-managed paths. It is easier to identify the exceptions where the control unit might get less benefit from DCM, so we list these here, and then provide some guidance about how to identify them.

- You are in the fortunate position of already having 8 channels to every DASD control unit and there is only one control unit on each of those channels. From a performance point of view, there is nothing that DCM can do to help since you cannot have more than 8 channels per control unit.

  It *is* possible, even in this environment, that DCM may provide improved availability compared to your current configuration, depending on how much effort you put into this when you set up your current configuration.

  What DCM *can* do for you is give you the ability to add additional control units without having to add more channels at that time.

- There are a number of LPs on the CPC, that are not in the same sysplex, and they are all significant users of the control unit. This is referring to the configuration we discussed in 10.4, “MIF considerations” on page 283.

  To identify this situation, you need reports that show the load placed on each control unit by each system on the CPC. While RMF will give you the raw information for these reports, it does not itself provide this information in a single report.

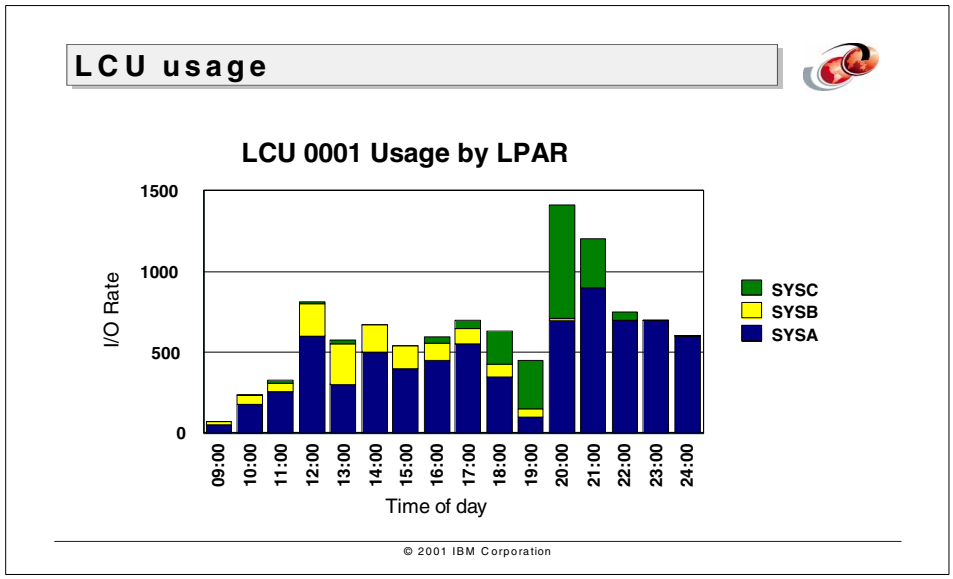
- Some of the control units are not supported by DCM. Older DASD control units may not fall within the list of supported IBM control units provided in 10.1.2, “Supported control units” on page 261.

- The non-managed channels on a managed control unit should not also be attached to any control unit that is not supported by DCM. The reason for this is that synchronous requests on those other control units will result in misleading utilization figures for the non-managed channels, possibly resulting in DCM making less-than-optimal projections of the number of channel paths required for the managed control unit.

- The load on the control unit changes dramatically over extremely short periods of time. An example would be a system used for share dealing, where the load goes from nearly zero just before the stock market opens, to thousands of I/Os per second just seconds after the market opens.

  Because the load on the control unit changes so quickly, it is possible that DCM could not react quickly enough to provide the required bandwidth in the short timeframe required. DCM is specifically designed not to make too many changes to a configuration in too short a time, to avoid causing dramatic

changes in system performance. In the case where a control unit would require many additional paths in a very short time, it may be more appropriate to use predominantly non-managed paths, with maybe one or two managed paths to provide additional bandwidth at times of abnormally high loads.



To identify the use of the control unit, you need to analyze RMF data for a number of representative intervals (different times of day, online vs. batch window, end of month processing, and so on) for each LP on the CPC, and for each control unit in the system. A given control unit has the same LCU number on every system on a given CPC, so this should make it a little easier to collect the data you require.

The information you want is in the RMF DASD Activity report in the LCU summary line and in the RMF I/O Queueing Activity report:

- I/O rate (number of I/Os per second)
- Average connect time
- Average pending time (CUB delay, DB delay, DPB delay)

If you can produce a graphical report, similar to the one in the figure above, it makes it easier to see the load placed on the control unit by each LP.

The average Pend time minus the control unit busy and device busy delays (CUB, DB) indicates the degree to which DCM can potentially help this LCU. Another important figure to monitor is the %ALL CHN PATH BUSY in the I/O Queueing Activity report, which reports the percent of times an I/O had to be queued because all the channels attaching the device were busy. Any value above 5% makes the LCU a good candidate for DCM.

However, just because a control unit does not have very high utilization, or high Pend times, does not mean that you should not consider it as a candidate for managed paths. While the control unit itself may not directly benefit from DCM, converting some of its channels into managed channels gives DCM access to channel bandwidth that may be used to help a different control unit.

You should also specifically investigate control units that are one of a number of control units on a given channel. It is likely, in configurations with multiple control units per channel, that there will be times when more than one control unit is busy, and other times when all the control units are idle. Such configurations are excellent candidates for DCM since the system will now automatically rebalance the managed channels across the managed control units, and thus avoid or at least reduce the channel contention that is likely in such an environment.

Finally, in relation to selecting “good” candidates, you should select control units that are busy at different times of the day. If all the control units you select are busy at the same time, then you lose the benefits provided by DCM moving channel bandwidth from idle to busy control units; if all the control units are busy at the same time, there is no one to take the channels from.

Ideally, you should build a table, similar to the following, showing the maximum I/O rate multiplied by the average connect time during the measurement period for each control unit from each LP on the CPC.

	SYSA	SYSB	SYSC
LCU 0010	300	1000	1
LCU 0011	1200	0	0
LCU 0012	0	300	700

In addition, for each control unit, you should create a graph similar to the one in the previous figure. This graph shows the usage of the control unit by each LP over the day. This helps you identify the maximum bandwidth requirement in total as well as by LP. It also highlights the LP that is placing the most load on the control unit. If one LP is consistently the biggest user of the control unit, this may be a good candidate for DCM. Conversely, if the load is distributed fairly evenly across all LPs throughout the day, this may not be a good candidate. Don’t forget to take into account which LPs are in the same LPAR Cluster. If the usage is

spread evenly over two LPs, but those LPs both are in the same LPAR Cluster, those LPs should be treated as one from a DCM point of view.

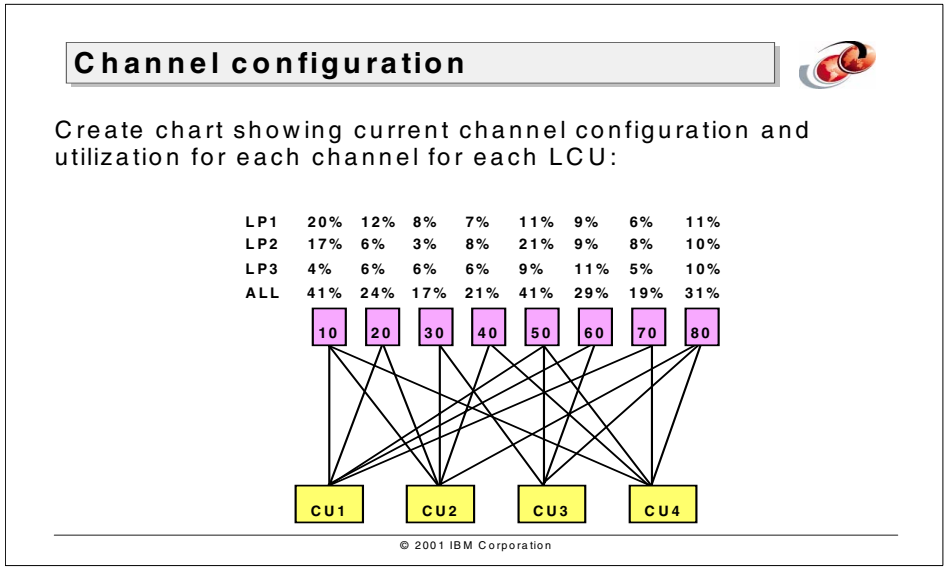
To help you create the graphs, you can use the RMF Spreadsheet reporter, which converts RMF data to spreadsheet format and which provides a practical example of how to use spreadsheet macros for converted reports and Overview records. The spreadsheet macros contained in the Spreadsheet Reporter are samples to demonstrate how you can use spreadsheets to process RMF data.

Using this information, you should be able to calculate the number of non-managed paths required for each control unit. The absolute minimum is one, although for availability we recommend a minimum of two paths for every control unit. Above this, the number of non-managed paths depends on the use of the control unit by systems *outside* the LPAR Cluster and the number of paths required to provide acceptable response times for the control unit's average load. To determine the minimum number of non-managed paths, you need to identify the maximum bandwidth requirement of systems outside the LPAR Cluster.

To help you identify whether a given control unit is supported by DCM or not, we have provided the following checklist. It summarizes all the known restrictions for DCM and should help you quickly identify whether a given control unit can be managed by DCM or not.

Restriction	**Violated by CU?**
Must be d/t 2105 or 9393 or supported non-IBM device	
LCU must be only connected via ESCON or FICON Bridge (FCV)	
LCU must be attached to managed channels via a switch	
The non-managed ESCON or FICON Bridge channels should only be attached to LCUs that are supported by DCM	
Cannot mix FICON Native channels with ESCON or FICON Bridge on the same managed LCU	
No non-managed channel connecting this control unit can be attached to any other control unit that has a mix of ESCON or FICON Bridge (FCV) and FICON Native. See 10.1.2, "Supported control units" on page 261.	
The load from LPs outside the LPAR Cluster should be small enough to be handled by the non-managed channels	
The load on the LCU should not increase dramatically over a short period of time (a few minutes)	

10.5.1 Understanding your configuration

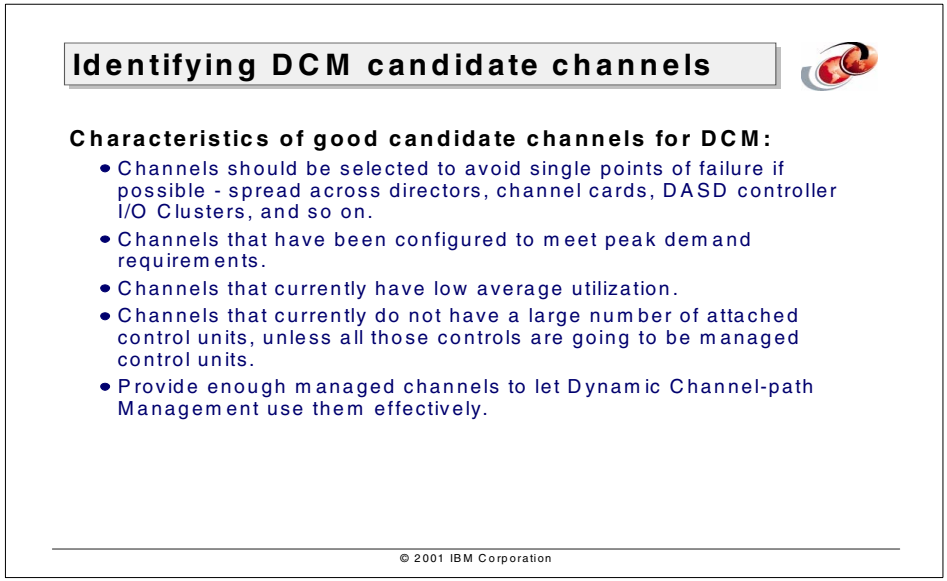


Having identified the control units that appear to be good candidates for DCM, you should create a chart similar to the one above. It is not necessary to show the ESCON Directors since we are only interested in channel utilizations at this time. This chart shows the LCUs on each channel, and which channels each LCU is attached to. It also shows the utilization for each channel from each LP on the CPC, as well as the total channel utilization for the whole CPC.

Remember that a managed channel cannot have any control units defined on it. So for each managed control unit, you have to decide which channels will be the managed ones (this is discussed in 10.5.2, “Identifying channels for DCM” on page 293). The chart above will help you identify the impact of changing any of the channels from non-managed to managed. For example, if you changed just CU1 to have one managed path, and changed CHP 10 to being a managed channel, CUs 2 and 4 and all the other channels that connect to them would be impacted because they could no longer use CHP 10 (until they were also defined as having one or more managed paths).

It is possible to create a graphical representation of the configuration using either HCD or HCM. Option 4 from the HCD main menu provides the ability to create a configuration diagram using either DCF/SCRIPT or GDDM. Using HCM, you can use the more powerful PC graphics capabilities to view and print the configuration.

10.5.2 Identifying channels for DCM



The other decisions that must be made are how many managed channels to have, and which ones should be managed. Each channel obviously has to meet the requirements of being an ESCON or FICON Bridge (FCV) channel that is attached via an ESCON Director.

You should provide Dynamic Channel-path Management with channels that are spread across ESCON Directors, CPC channel cards, DASD controllers, and so forth. This will give Dynamic Channel-path Management the greatest flexibility to add channels that avoid single points of failure, and thereby improve the availability of the control units they attach to. This is discussed more thoroughly in 9.9, “RAS benefits” on page 254.

To arrive at the target number of managed channels, you total up the number of channels currently being used for the control units that will be managed, and subtract the number of non-managed paths you wish to retain on each control unit, allowing for situations where there are multiple control units on a channel. The result is the number of channels that are available to be used as managed channels.

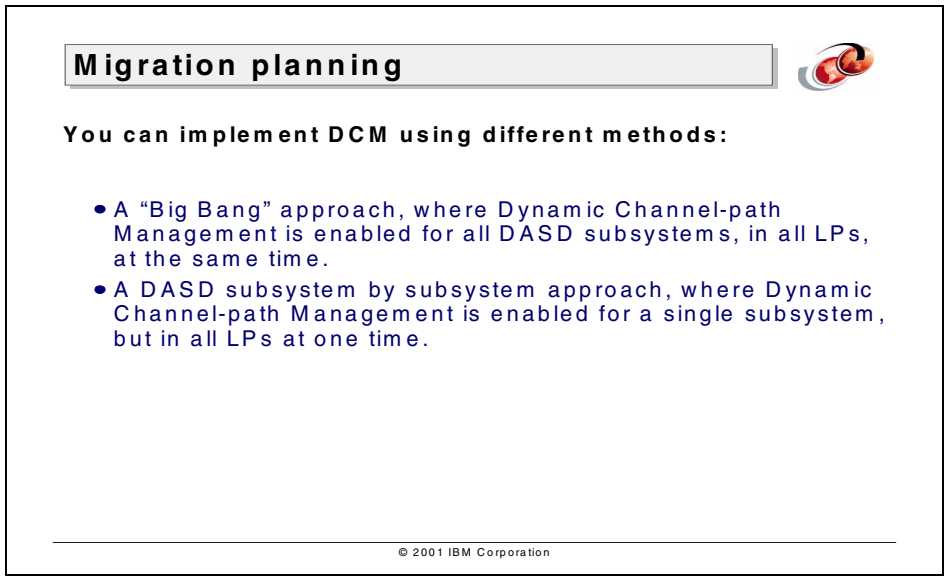
Channels that have been added to a control unit specifically to support peak load requirements are good candidates for conversion to managed channels. The bandwidth provided by these channels is not required most of the time, and can effectively be used elsewhere during these times.

Similarly, channels that currently have low average utilization are good candidates. If they are lightly used, removing them from the control units they are currently attached to should not have a significant impact, and because those control units are lightly loaded they can probably survive with a smaller number of non-managed channels.

If some of your channels have just one control unit on them, it is relatively easy to identify some of those channels as candidates for being managed channels.

If most of your channels are attached to multiple LCUs, it is a bit more difficult to identify how many, and which ones, should be converted to managed mode. A good place to start is with a configuration diagram, like the one in 10.5.1, “Understanding your configuration” on page 292.

10.6 Migration planning



There are a number of approaches that can be taken to the task of implementing Dynamic Channel-path Management. In this section, we review the options available to you, and provide some guidance about which may be the most appropriate for your installation.

You can implement DCM by using one of two approaches:

- A “Big Bang” approach, where DCM is enabled for all DASD subsystems, in all LPs, at the same time.
- A DASD subsystem by subsystem approach, where DCM is enabled for a single subsystem, but in all LPs at one time.

Another approach would be to migrate LP by LP, however DCM is enabled or disabled at the LPAR Cluster level. Therefore, the only way to implement this LP by LP approach would be to migrate one LP from OS/390 to z/OS and enable DCM in that LP, then migrate another LP to z/OS, at which point it would start using DCM, and so on. However, because the systems that are not in the LPAR Cluster cannot use the managed channels, this is unlikely to be an acceptable approach.

While the Big Bang approach has certain appealing qualities (it is much easier to plan for than a phased approach), we think that most installations will wish to implement DCM in a phased manner. The ‘Big Bang’ might be acceptable for a small test LPAR Cluster.

In the case of a system that is running on an IBM zSeries 900 in Basic mode, the considerations are similar to the Big Bang approach—the only difference being that there is effectively only one “LP” in the LPAR Cluster. Once again, however, we doubt that most installations will want to make that many changes to a production system at one time.

So, by a process of elimination, we arrive at the most likely migration approach: migrating one or a small number of control units to DCM at a time.

Ideally, you want to start with control units that are the only control unit on the channels they are attached to. This will permit you to convert some of those channels to being managed channels, without removing channel bandwidth from other control units.

At the same time, you may take busy control units that are attached to less than 8 channels and add some managed paths to those control units. This should provide a performance boost for those control units, without the complexity of trying to convert some of the existing paths to managed paths.

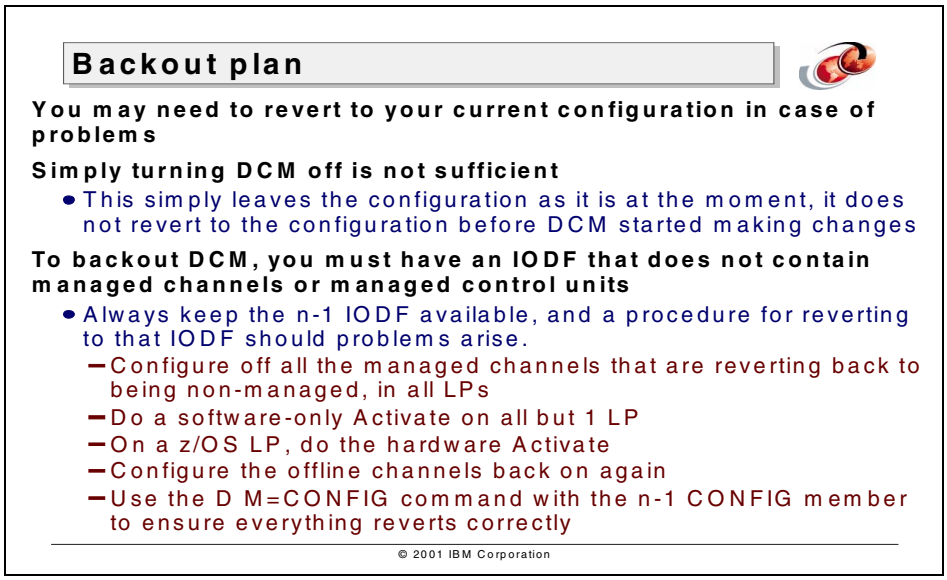
As with any migration, the amount of resources you require during the migration may exceed the amount you had at the beginning, or will have at the end. In the case of DCM, the additional resource that you may require is in the form of channels. You should plan on potentially needing a higher than normal number of channels during the period that you are in the process of implementing DCM. This ensures that you will continue to have acceptable performance should you need to stop using the managed channels for some reason.

For example, when you start, you may have two LCUs, each with six paths. There is only one LCU on each channel, and all the channels are shared by all the LPs. The final configuration may be two non-managed and up to six managed paths that connect to both LCUs. However, during the implementation phase, you may decide not to have fewer than four non-managed paths to each LCU, to ensure that you have sufficient bandwidth even if the managed paths are not available. As the target configuration is to have six managed paths, you would require a total of 14 channels during this phase of the project. In this case, you would need to “borrow” two additional channels until the migration to DCM for these control units is complete. Once you are happy that DCM is performing to your expectations, you can start removing channels, ending up with a total of just 8, rather than the 12 you started with. The channels that are freed up can be put towards the next set of control units that you migrate.

When you are planning your final configuration, and how you will get there, there are a few things you should bear in mind:

- DCM has no knowledge of activity outside its LPAR Cluster. Adding a path to a control unit from one LPAR Cluster might help that LPAR Cluster, but it could potentially impact work running in other systems on other CPCs (because adding a path may increase control unit busy). If this appears to cause a problem, define fewer managed paths from this LPAR Cluster. Remember that the number of managed paths defined for a control unit does not have to be the same in every CPC.
- Until you are comfortable with DCM, you might like to use it with less busy, less critical control units. The channels that you free up as a result could be moved, as non-managed channels, to other more critical control units until you are ready to implement DCM on those control units.

10.7 Backout plan



Any significant change that is being made to a production system requires a backout plan, regardless of how thoroughly it has been tested in advance. Hopefully, if your testing and planning have been thorough, it should not be necessary to invoke the backout plan, but the time that you don't have one is the time that you will need it.

You have two options if you have a problem with DCM that causes you to consider backing it out. The first is simply to turn off DCM completely in the LPAR Cluster by using the SETIOS DCM=OFF command. The effect of this command is to keep the configuration as it stands at the moment. It will *not* back out DCM, however it will stop it from making further changes to your environment. This provides you with an opportunity to investigate the perceived problem, without impacting the system by making a potentially large number of configuration changes, as you would if you completely back out DCM.

However, if you followed our recommendations and implement DCM in a phased manner, a few control units at a time, it is possible to back out the changes, hopefully without affecting too many control units. To implement the backout, you must have the IODF that was in use before the current one. This might be an IODF with no DCM definitions, or perhaps it just has fewer managed channels and control units than the current IODF. In any case, you must first configure offline all the channels that will revert back from managed channels to non-managed. Next, do a Software-only activate on all but one of the systems on

the CPC. Once the Software activates have completed successfully, do the Hardware activate on the remaining (z/OS) LP. This should revert you to the configuration that was in use prior to the latest DCM changes. At this point you must configure the offline channels back online again, in all LPs that will be using those channels.

Obviously, this approach depends on the fact that you have not made any other changes to the IODF since you updated the DCM configuration. If you have made other changes, those changes will be lost if you back out to an older IODF. However, it is likely that if there are problems with Dynamic Channel-path Management, they will show up shortly after you activate the new IODF, and you can then back out straight away should you wish to do so.

In addition to keeping the old IODF, you should also save the corresponding CONFIGxx member. Once the Hardware Activate completes and you have configured the channels back online again, you can use the D M=CONFIG(xx) command to ensure that everything is as it should be.

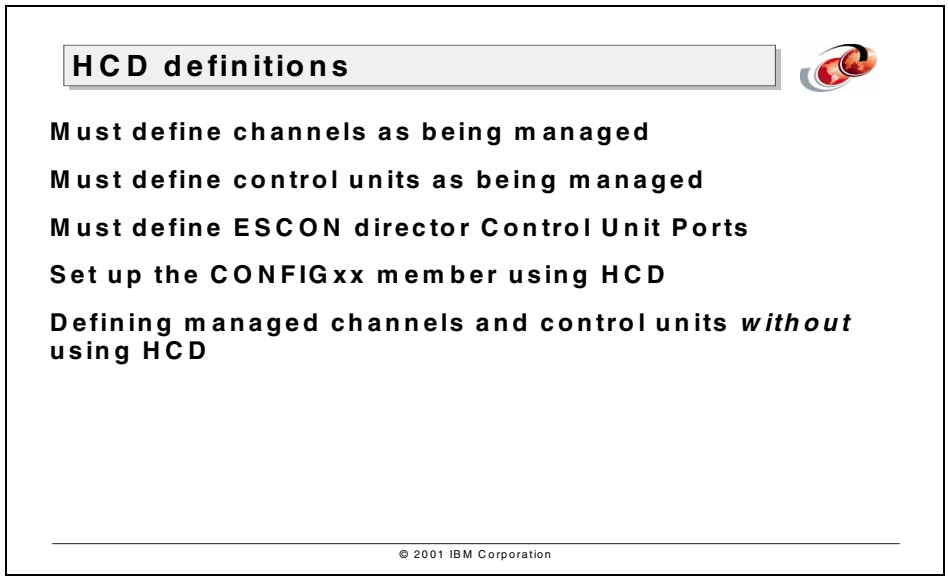
11

Implementing Dynamic Channel-path Management

If you have gotten this far, the hard part is over. Actually enabling Dynamic Channel-path Management (DCM) itself is fairly straightforward. In this chapter, we provide a step-by-step process to lead you from the point where you know what channels and control units you want to manage with Dynamic Channel-path Management through to having DCM up and running and managing some of your DASD paths.

We assume that the starting point is a level of the operating system that is DCM-capable and that you are running on a IBM zSeries 900 or later processor. We also assume that you have already assembled all of the information that you need, and understand how DCM works. Therefore, this chapter is really just a checklist of the steps that have to be carried out to get DCM up and running.

11.1 HCD definitions

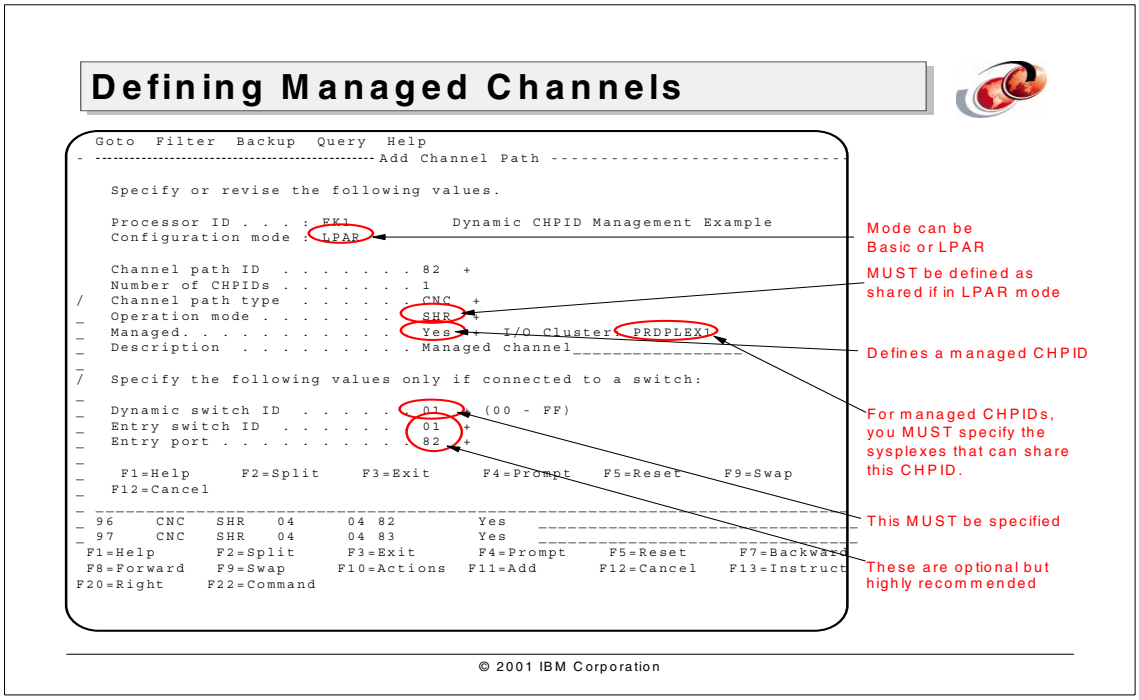


As discussed in 9.3, “Configuration definition for DCM” on page 219, there are two (possibly three) sets of changes that need to be made in HCD:

- You must define some channels as being managed.
- You must define some control units as having at least one managed path.
- You have to define the Control Unit Port on the ESCON Directors. This might have been done already in your existing configuration. If not, a very good description of how to do this is available in Chapter 10, *Add an ESCON Director* in *z/OS Hardware Configuration Definition Scenarios*.

In addition, we strongly recommend that you use HCD to create a CONFIGxx member every time you update your configuration, and then use the D M=CONFIG(xx) command after every IPL to ensure that the current configuration matches the planned one.

11.1.1 Managed Channel definitions



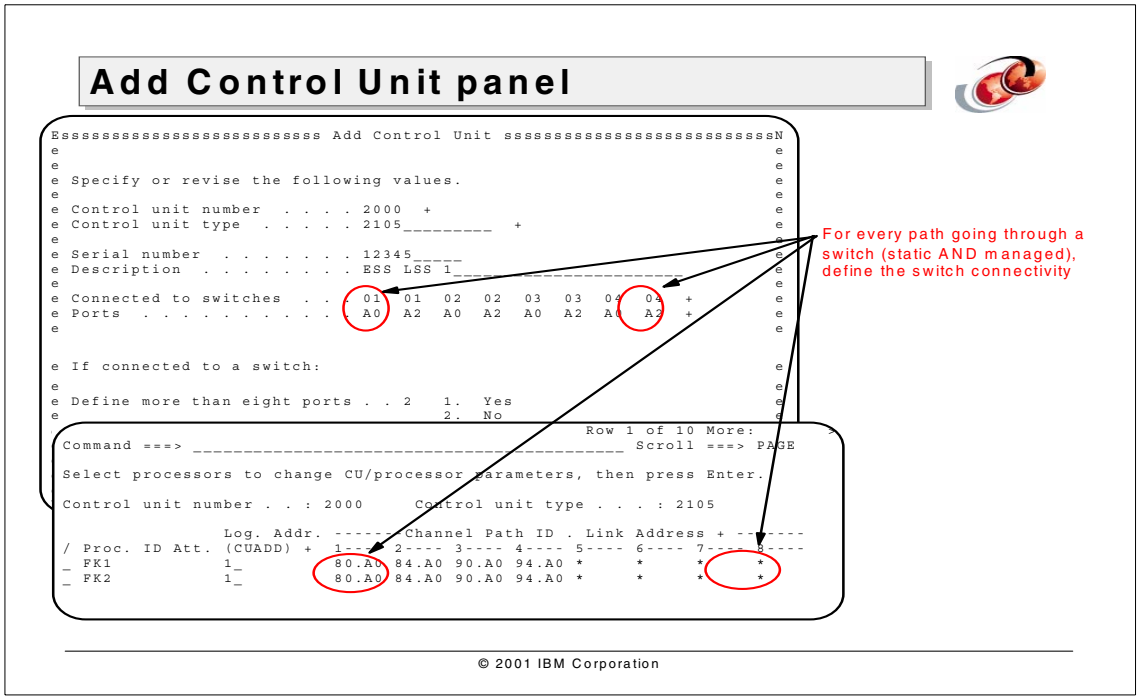
Managed channels *must* have the following characteristics in HCD:

- They cannot be in the path list for any control units. If you are converting an existing channel to become a managed channel, you must first remove it from the definition of any control units that it is currently attached to.
- If the CPC is in LPAR mode, the channel must be defined as Shared.
- You must specify YES in the Managed field.
- You must specify the sysplex name of the LPs that can share this managed channel. This is specified in the I/O Cluster field.
- You must specify the Switch ID of the dynamic director in the path to the control unit.

In addition, we recommend that you complete the Entry Switch ID and Entry Port fields.

Complete this panel for each channel that is to be a managed channel.

11.1.2 CU definitions



Having defined the managed channels, you now have to tell HCD about the control units that it can use with those channels.

A managed control unit must have the following HCD characteristics:

- The control unit must be attached to a Switch that is in turn attached to managed channels.
- There must be at least one path that is specified with an asterisk(*) rather than a CHPID and Link Address.
- The control unit must be one of the types supported by Dynamic Channel-path Management. As many different types of control units are defined to HCD as a 3990, it is not possible to use HCD to enforce that you only define managed paths for the correct types of control units.

  However, when Dynamic Channel-path Management is initialized, it knows which control units are not supported, and it knows the actual device type of every attached control unit, so it will not add managed paths to an unsupported control unit—even if you have (incorrectly) defined that control unit to HCD as having managed paths.

- Remember the restriction that all the non-managed channels that are connected to a managed control unit must only be attached to control units that are supported by DCM. That is, if a managed 2105 is attached to non-managed CHPID 20 (for example), CHPID 20 should not also be attached to a 3990. HCD is unable to enforce this restriction because the control unit type defined in HCD does not necessarily match the actual control unit make and model.

  Similarly, any control unit that has a static ESCON or FICON Bridge channel in common with a managed control unit, even if it itself is not managed, should not have any channels in common with control units that are not DCM capable.

In addition, we strongly recommend that you provide HCD with accurate information about the Switch ID and Switch port that *every* Host Adaptor Port is attached to, as shown in the figure above. If you *do* include this information, then HCD requires that you provide the equivalent information for every non-managed channel this control unit is attached to. The reasons why we recommend that you do this are:

- So that HCD can check to ensure that you have not mistakenly defined two control units connected to the same Switch port.
- So that HCD can create a complete, and correct, configuration report.
- To enable HCD to build a comprehensive matrix of the Switch ports that can be used by Dynamic Channel-path Management. This information is used when you build the CONFIGxx member using HCD. If you do not provide this information, HCD is not aware of the Switch ports that are only used by the managed paths.
- To make it easier to complete the configuration information when you import the configuration definitions into HCM.

To convert an existing control unit into a managed control unit, you must update the control unit definition in HCD, either by converting existing paths to be managed paths, by replacing the CHPID.Link Address with an asterisk(*), or by adding additional paths and making those managed paths.

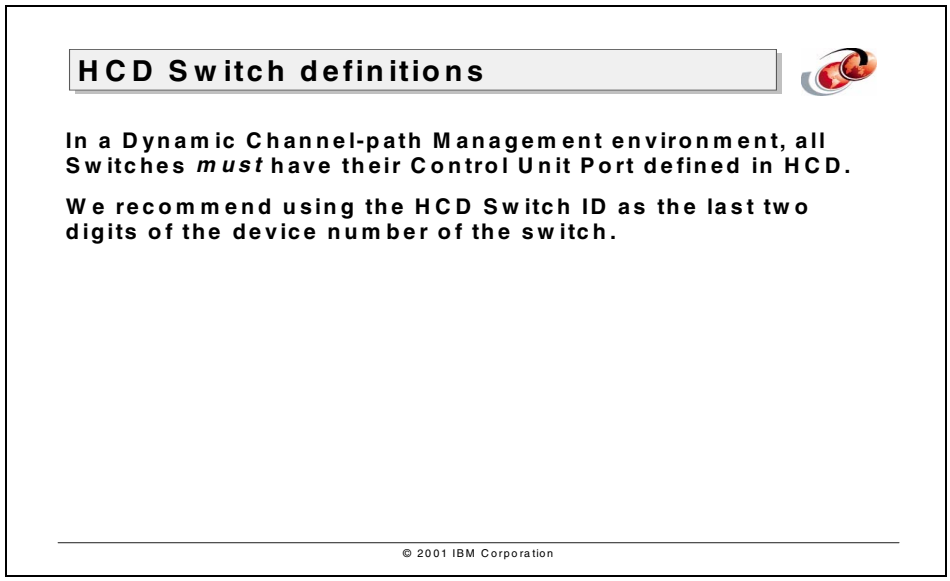
Don’t forget about logical path restrictions. When you identified the number of paths to have on each control unit, the logical path capabilities of the control unit should have been taken into account at that time.

While it is not strictly related to DCM, this is a good place to discuss how you define your control units in HCD. The introduction of Dynamic Channel-path Management once again raises the question of how many HCD control unit definitions to have per LCU. Most installations have traditionally used two definitions per LCU. However, some have used four, while others have used just

one. To give DCM the maximum flexibility to dynamically change paths, you should have as few control unit definitions per LCU *as recommended by the control unit vendor*. For example, IBM 3990s should have two control unit definitions per LCU, whereas IBM 2105s should only have one.

Tip: If you only define a single control unit on a given switch as being managed, HCD will issue a warning message (CBDG514I). The reason for this is that, in terms of getting DCM to actively manage channel paths, there is no point in defining just one managed control unit. When the system IPLs, DCM will assign its paths to that control unit and have nothing further to do. You must have more than one managed control unit attached to a given switch for DCM to be able to move paths between them as required.

11.1.3 Switch definitions

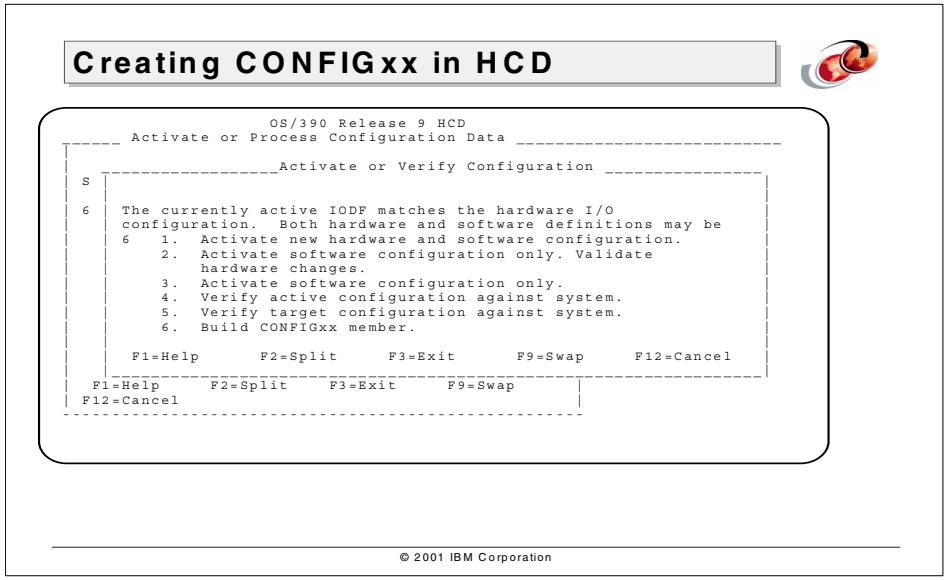


Because Dynamic Channel-path Management uses the Control Unit Port on the Switch to get information about the configuration of the Switch, and all the devices attached to it, it is required that you define the Control Unit Port in HCD. Prior to DCM it was recommended that you do this, but it was not required.

To define the Control Unit Port, you must specify a control unit number and device number for the Switch in the Add Switch panel in HCD. The control unit that you specify is automatically attached to the Control Unit Port (FE) and the device is connected to the control unit. You must ensure that the CUP is attached to all the operating system configurations that will be used by any of the systems in the LPAR Cluster.

We recommend using the Switch ID as the last two digits of the device number that you assign. This makes it easier later to associate the Switch with the Switch definition in HCD.

11.1.4 Creating a CONFIGxx member



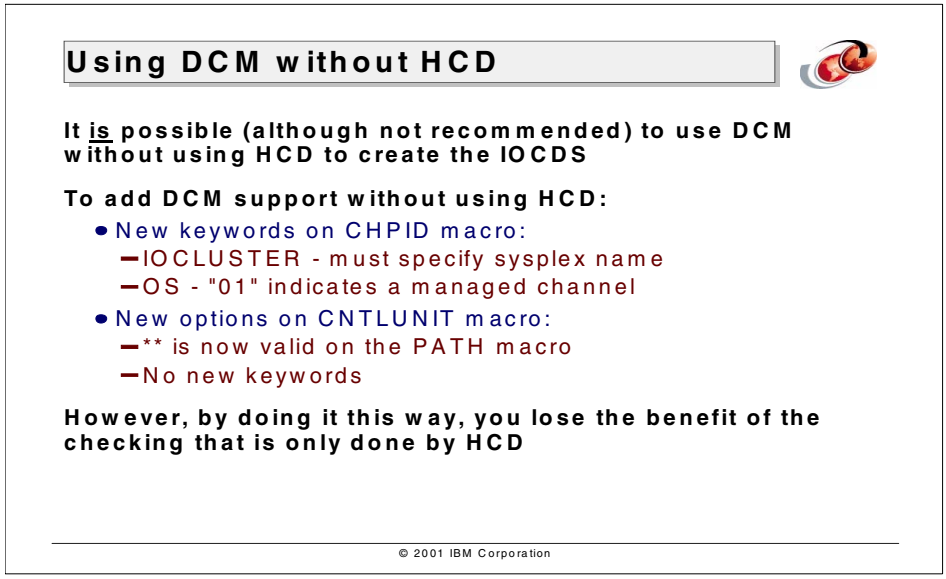
In OS/390 2.6, HCD added the ability to build a CONFIGxx member for you, based on the information contained in the IODF. This is a very useful facility which saves you considerable time building and maintaining your CONFIGxx member. We have always recommended using a CONFIGxx member, which provides a powerful and easy-to-use way of checking that your current configuration matches your planned one.

This is even more important in a DCM environment, because it is easier for an inoperative component to go unnoticed. For this reason, we strongly recommend using the HCD capability to build your CONFIGxx member.

As well as specifying which components should be on and offline, and which channels should be managed, this facility also provides an easy way of indicating which switch ports should be disabled for Dynamic Channel-path Management. When you select the option to build the CONFIGxx member, HCD presents you with a matrix of all the Switch ports, indicating the ones that it knows are attached to managed control units. For the managed paths, it relies on you having entered the Switch ID and port number in the CBDPCU10 panel (the first panel when you add a new control unit). If you do not enter this information, HCD only knows about the ports that are used for the non-managed paths, and therefore will only provide a subset of the ports that can potentially be used by DCM.

If you wish to stop DCM from using some of these ports, you can indicate this by simply overtyping the “Y” with an “N” to indicate that you do not want DCM to be able to use this port. You should understand that doing this will *not* stop DCM from using the port: all you are doing is providing information so that the CONFIGxx member will indicate that that port is not to be used by DCM. However, you can then use the CONFIGxx member with the CONFIG MEMBER(xx) command to change the current configuration to match the definitions you have in CONFIGxx. Also, when you change the status of a port from “Y” to “N”, that information is not saved anywhere other than in the CONFIGxx member, so the next time you use this function, you will have to update the matrix again to indicate if there are ports that you do not wish DCM in this LPAR Cluster to use. Before you use the CONFIG MEMBER(xx) command, however, refer to the discussion and warning about its use in 10.1.4, “Switch considerations” on page 267.

11.1.5 Setting up DCM without HCD



Hopefully you are not reading this section! At this point, we would hope that all customers are using HCD, rather than coding your IOCP decks manually.

However, if for some reason you are still building IOCP decks without using HCD, it *is* still possible to use managed channels and control units.

One thing to be aware of, however, is that there is some validity checking that is only done by HCD. If you are not using HCD, IOCP will not carry out this checking for you.

Also, if you do not use HCD to create the IOCDS, it will not be possible to implement dynamic I/O configuration changes from z/OS. It is, however, possible to implement the changes from a VM LP (we are assuming that the reason for not using HCD is that you maintain your IOCP statements in VM) as long as you do the Software Activate in MVS before you do the Hardware Activate from VM.

If you use VM to update the IOCDS, and you want to be able to use Dynamic Channel-path Management in the z/OS LPs, you must specify the `DYN` keyword when you run the IOCP program in VM. This causes the creation of a token in the IOCDS. This token *must* be present if you wish to use Dynamic Channel-path Management. When you use HCD to update the IOCDS, HCD takes responsibility for the creation of the token.

There are two IOCP macros that are changed as a result of Dynamic Channel-path Management. The first of these is the CHPID macro, to allow you to define a channel as being managed. Two new keywords are added:

IOCLUSTER This keyword must contain the name of the sysplex that an operating system must be a member of in order to use the channel.

OS This keyword is used to identify a channel as being managed. It is a one-byte field, containing 00-FF. If bit 7 is on, this indicates that this is a managed channel. Therefore, you should use a value of 01 for managed channels. It is not necessary to add this keyword to non-managed channels.

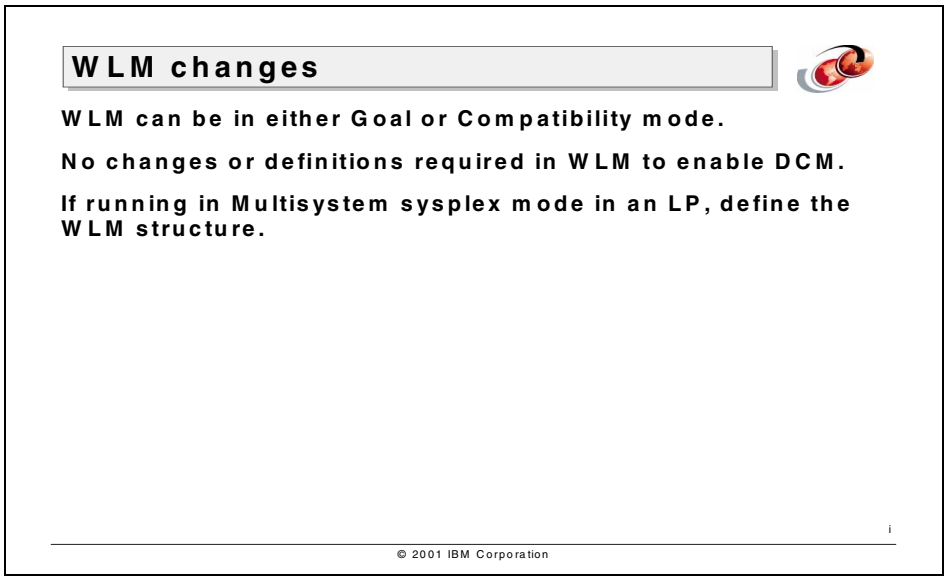
The other macro that has been updated is the CNTLUNIT macro, once again to let you indicate that this is a managed control:

PATH Keyword has been updated to support double asterisk (**) as a valid value, indicating that this is a managed path. Note that even though you use a single asterisk in HCD to indicate a managed path, you must specify two asterisks on the CNTLUNIT macro when you are creating an IOCP input deck.

LINK The LINK keyword has also been updated to support the specification of double asterisk (**) as a valid value, indicating that the Switch port will be determined dynamically.

If you maintain your IOCDS from VM, it is especially important in a Dynamic Channel-path Management environment that all changes are reflected in the z/OS IODF and a Software Activate is done in all z/OS and OS/390 LPs before the Hardware Activate is done from VM. This may require closer cooperation between the MVS and VM groups than may have existed previously.

11.2 WLM changes

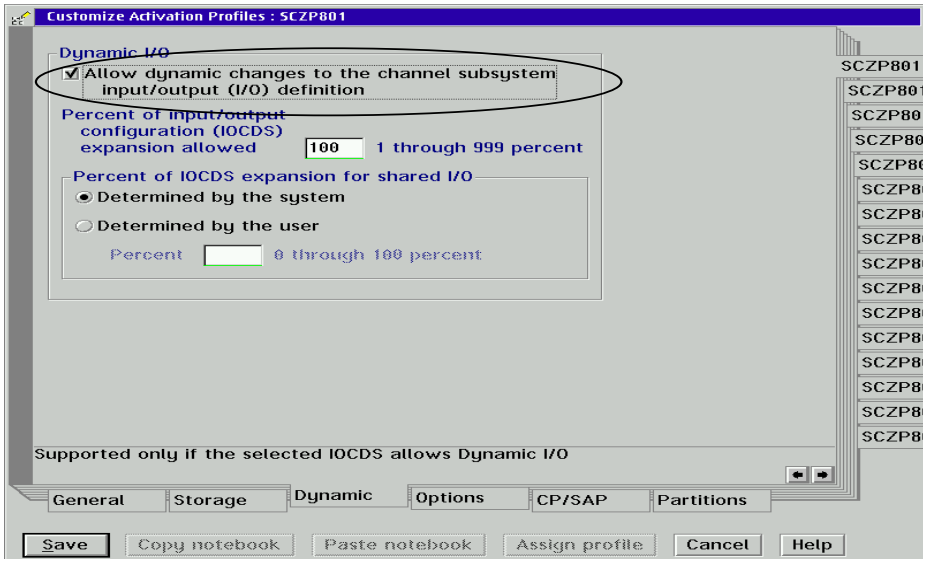


There is no “switch” as such in WLM to turn Dynamic Channel-path Management on or off. So if you are running in Basic mode, or the system that will be using DCM is in XCFLOCAL or Monoplex mode, there are *no* WLM-related changes required.

If the system that will be exploiting DCM will be running in an LP and is in Multisystem sysplex mode, you *must* have a WLM structure. If the structure is not defined or is not available, DCM will not function in this environment. Rather than repeating all the information about how to set up the structure here, we refer you to 5.3, “Defining WLM structures” on page 129 for information about defining the WLM CF structure. The structure size recommended in that section is large enough to support both DCM and WLM LPAR CPU Management.

Don’t forget to give the RACF userid associated with the WLM address space access to the LPAR Cluster structure. This is protected by the IXLSTR.structure_name profile in the SAF class CLASS(FACILITY). If you have not defined a userid for WLM, and you have a RACF profile that covers the LPAR Cluster structure, the RACF violations will not tell you which address space is failing to obtain access.

11.3 HMC changes

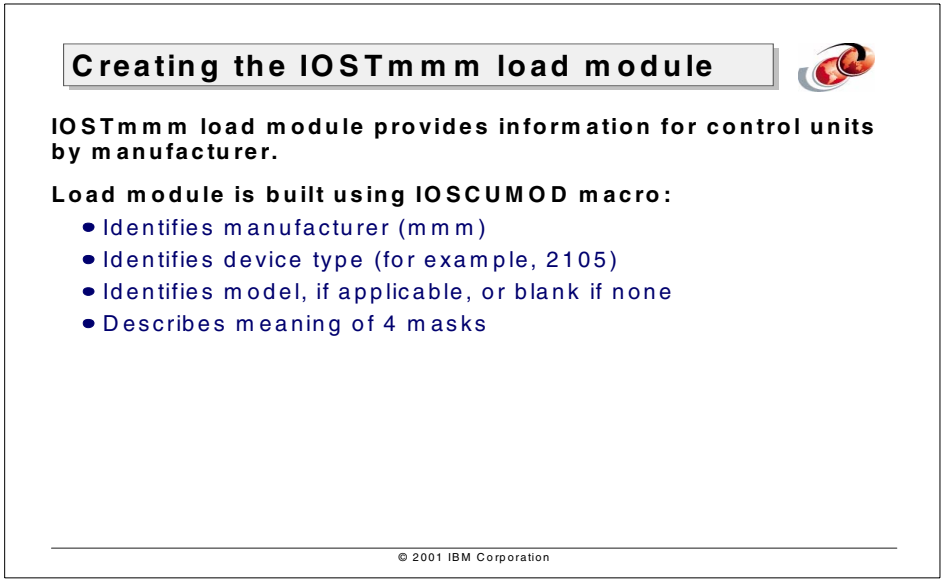


There are no changes to the HMC specifically for Dynamic Channel-path Management. However, in order for an LP to use the Dynamic I/O Reconfiguration capability, you must have enabled this function in the CPC Reset Profile, as shown in the figure above.

In addition, all LPs in the LPAR Cluster must be authorized to make I/O reconfiguration changes for the CPC. Otherwise, only a subset of the systems in the LPAR Cluster would be able to initiate DCM configuration changes. This is discussed in more detail in 10.1.1, “CPC requirements” on page 259.

Also, on the Options tab of the CPC Reset Profile, you must enable the “automatic input/output (I/O) interface reset” option. This is also discussed in 10.1.1, “CPC requirements” on page 259.

11.4 Building the IOSTmmm module

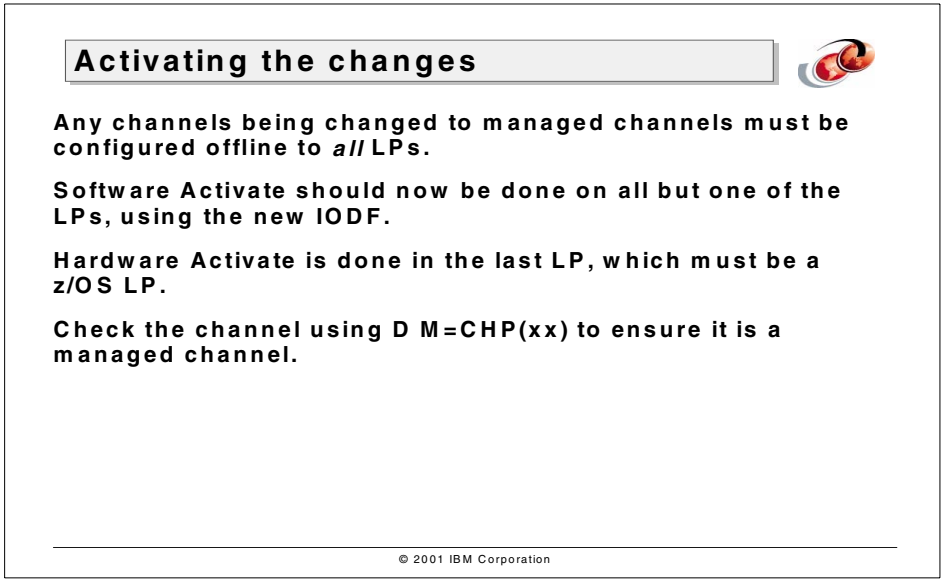


The IOSTmmm load module must reside in a LNKLST library and contains information used to determine the points of failure within a given control unit. The mmm in the load module name represents the control unit manufacturer name as provided in the manufacturer field of the Node Descriptor. The IOSTIBM module, the one that maps the IBM control units, is shipped in SYS1.LINKLIB as part of the system, and is updated as part of the changes provided with every new IBM control unit.

For non-IBM control units, you will need to obtain a copy of the IOSTmmm load module from each DASD vendor that you use.

Updates to the IOSTmmm load modules can be activated dynamically using the SETIOS DCM,REFRESH command.

11.5 Activating the changes



Having completed all these changes, you are now ready to activate the changes. First, configure offline any channels that you are converting from non-managed to managed channels. This must be done in *every* LP that is currently using the channel.

Once the channel is offline everywhere, do a Software Activate using the new IODF on all but one of the systems on the CPC. Don't forget that there could be systems in other sysplexes that might be using this channel, so *all* affected systems should be updated. When this has completed successfully, do a Hardware Activate on the final system. Obviously, this last system has to be a z/OS system.

When the Hardware Activate has completed, the managed channel will be offline, and must be configured online in every LP in the LPAR Cluster. To ensure that the change was successful, issue a D M=CHP(xx) command for every managed channel to ensure that it is in fact now a managed channel, and that it is online and attached to a Switch. Also, issue a D M=DEV(xxxx) for a device behind a managed control unit to ensure the control unit has the expected number of managed paths defined. The use of these commands is discussed in 12.1.1, "D M=CHP command" on page 319 and 12.1.3, "D M=DEV command" on page 322.

The managed channels are now online and available to DCM, and they should now start being assigned to the various managed control units. You can use the Operator commands to display information about the managed channels and control units, and the enhanced RMF reports to show performance information for these entities.

12

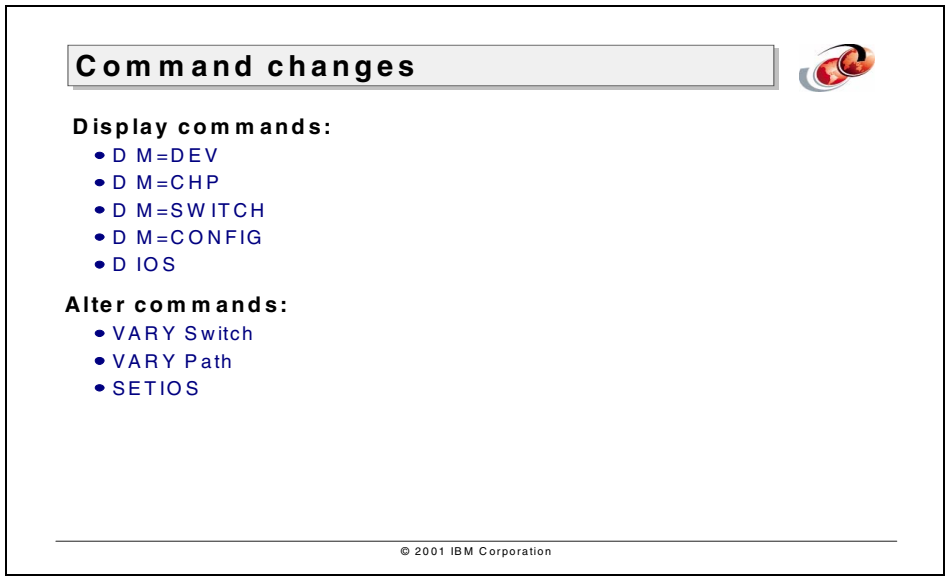
Operating Dynamic Channel-path Management

In normal operation, there should be no need to interact with Dynamic Channel-path Management (DCM). If the installation has defined managed channels and DASD subsystems with managed paths, DCM will automatically start managing the paths following an IPL. All actions that are taken by DCM are automatic and should be transparent, from an operator's point of view.

However, if you wish to display information about DCM or exert some control over the actions that it can take, then there are some new commands and enhancements to existing commands that you must be aware of.

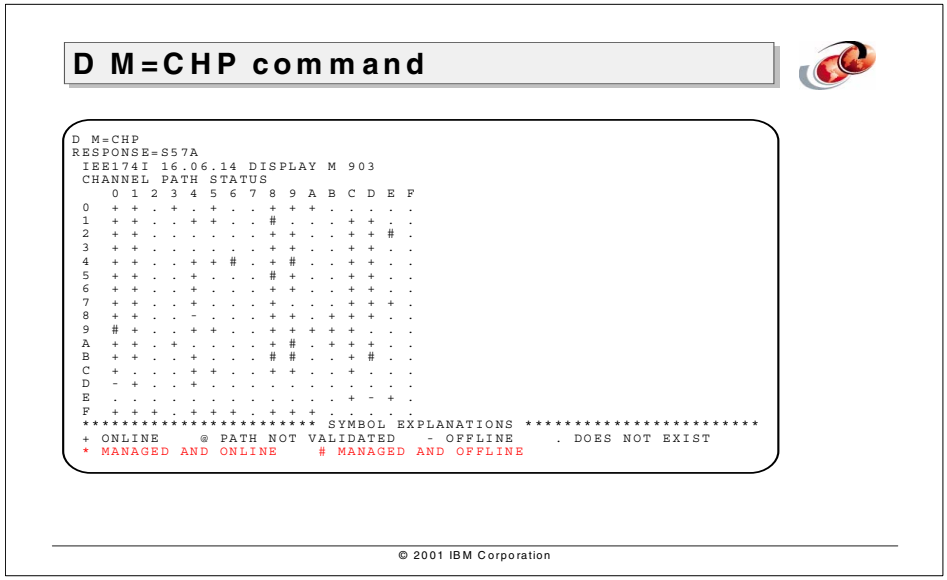
Before we discuss the actions that an operator or systems programmer might take, we describe the command changes and the new DCM-related messages. After that, we provide a number of typical scenarios you may encounter and show how you might deal with them.

12.1 New operator commands



In support of DCM, there are new commands and updates to existing commands. There are commands to display information about the configuration, and commands to change the configuration. They are discussed in detail in this section.

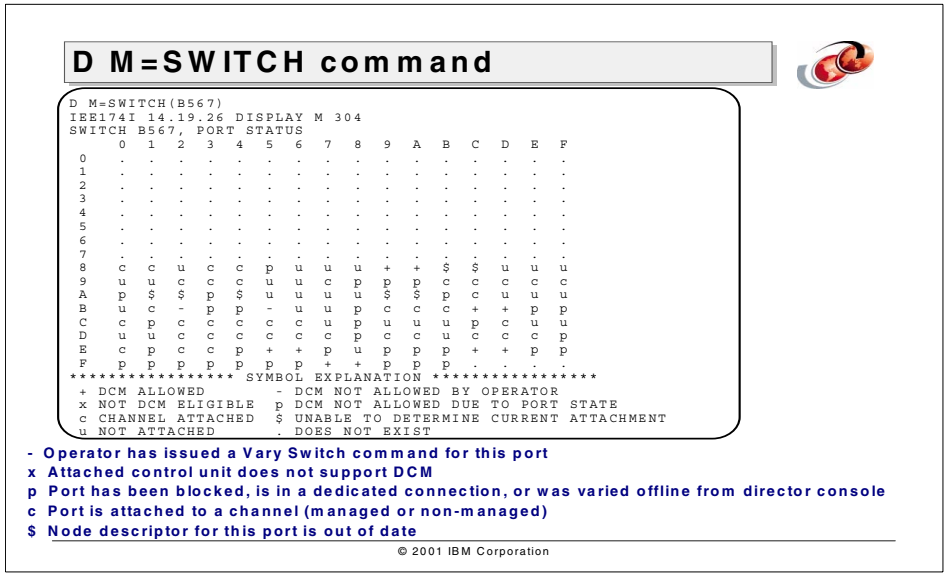
12.1.1 D M=CHP command



The DISPLAY M=CHP command has been updated to provide additional information related to Dynamic Channel-path Management. If the D M=CHP command is issued without specifying a CHPID number, a two-part report will be provided. The first part indicates the status of each channel (online, offline, managed, and so on), and the second part indicates the channel type (ESCON, CTC, FICON, and so on). Managed channels are indicated in the first part of the report by an asterisk (*) if the channel is managed and online, and by a “#” if the channel is managed and offline.

If you specify a CHPID number on the display command, the response will include the channel type, whether it is online or not, the status of any devices on the channel, and the device number of the Switch it is attached to (if it is attached to a switch). The provision of the Switch device number is new, and this information will be used to display information about the Switch configuration. If you specify a range of CHPID numbers, you will get the same information, but for every CHPID that you specified.

12.1.2 D M=SWITCH command



The SWITCH option of the DISPLAY M= command was introduced by OS/390 V2R10. Even though OS/390 V2R10 does not support DCM, it does do the topology gathering that is used by DCM in z/OS, and you can use this command in this release. The command allows you to display information about the selected switch (indicated by specifying the switch device number), and the components that are attached to it (CPCs, control units, or other switches).

The Switch device number is found in the output when you issue the D M=CHP command for a specific CHPID.

The information provided in the output from the D M=SWITCH command is a combination of actual hardware information about the Switch, discovered when DCM gets the Node Descriptors for all attached devices, and software information, which indicates whether DCM in this LPAR Cluster can use the related port. If you specify a port number on the display command (D M=SWITCH(ssss,pp)), the response includes the node descriptor of the device that is connected to the port, as well as the DCM status of the port.

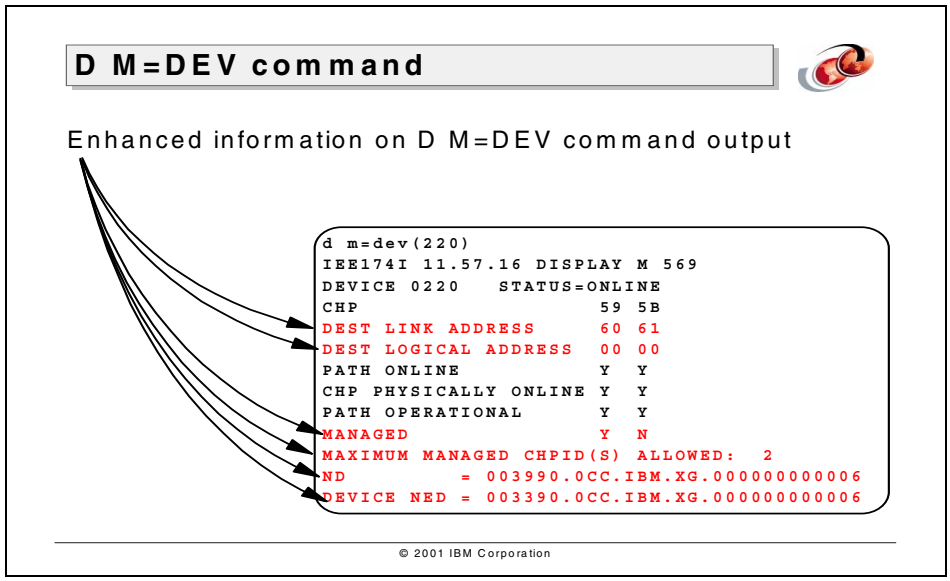
It is important to understand that the Switch itself has no knowledge of DCM, and is not involved in the decision as to whether DCM can use a specific port. The information about whether DCM can use a specific port or not is held entirely within the operating systems in the LPAR Cluster. It is entirely possible that one

LPAR Cluster will not be allowed to use a specific port, but a different LPAR Cluster will be able to use it. The information provided in the response to the D M=SWITCH command only reflects the status of the LPAR Cluster that it was issued in.

In OS/390 V2R10, this command shows a status of DCM NOT ALLOWED DUE TO PORT STATE for any port that has been blocked, is in a dedicated connection, or was varied offline from the director console or ESCON Manager. Because OS/390 V2R10 does not support DCM, it is not aware of the DCM status of the ports—for that information, you must issue the command on a z/OS system.

On z/OS systems at the time of writing, a port status of DCM ALLOWED simply indicates that the port is connected to a DASD control unit. It does not mean that the control unit is a managed control unit, or even that it is a control unit type that is capable of being managed by DCM (for example, it could be a 3990 control unit).

12.1.3 D M=DEV command

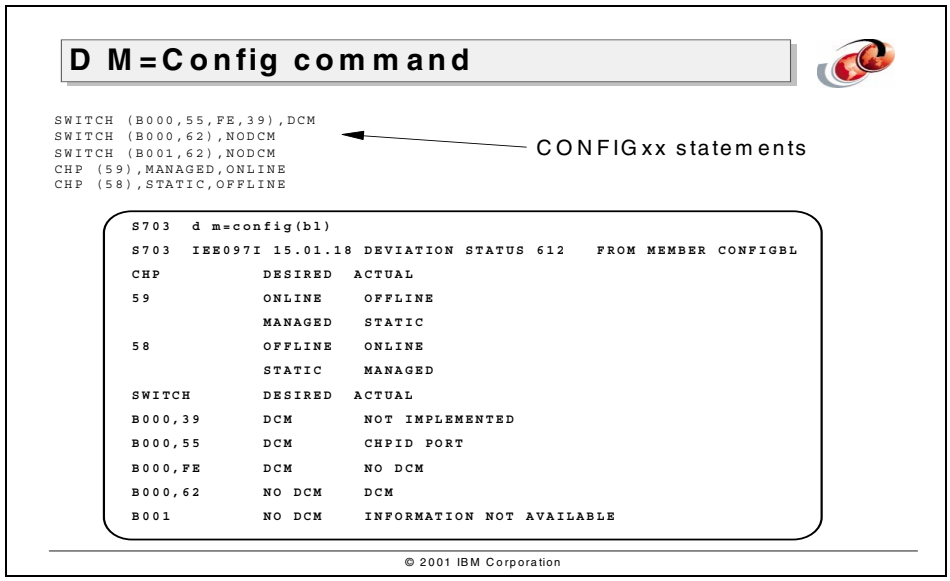


The DISPLAY M=DEV command has been expanded to include an indicator of whether a given path is managed or not. It also lists the maximum number of managed paths that can be used by the control unit the device is attached to.

Additional information provided for each path includes the CUADD of the LCU the device is attached to (in the DEST LOGICAL ADDRESS field) and the port on the switch that the associated control unit host adaptor port is attached to (in the DEST LINK ADDRESS field).

The final additional piece of information provided is the Node Descriptor information for the device itself and the control unit the device is attached to. The Node Descriptor of the control unit is displayed in the “ND” line, and the Node Element Descriptor of the device is contained in the “Device NED” field.

12.1.4 D M=CONFIG command



The CONFIGxx member of SYS1.PARMLIB can be used to compare the current configuration with a “model” configuration. Ever since configurations started getting more complex, IBM have recommended the use of this member to check that the current configuration matches a model configuration. This is done by defining the model configuration in the CONFIGxx member and using the D M=CONFIG command to compare the current active configuration with the model defined in the CONFIGxx member.

The problem with this process was that it was a time-consuming process to create the CONFIGxx member. To alleviate this particular restriction, HCD in OS/390 V2R5 added the ability to create a CONFIGxx member based on a specified IODF, thus giving you the benefits of this facility without the headache of having to manually keep the CONFIGxx member up to date.

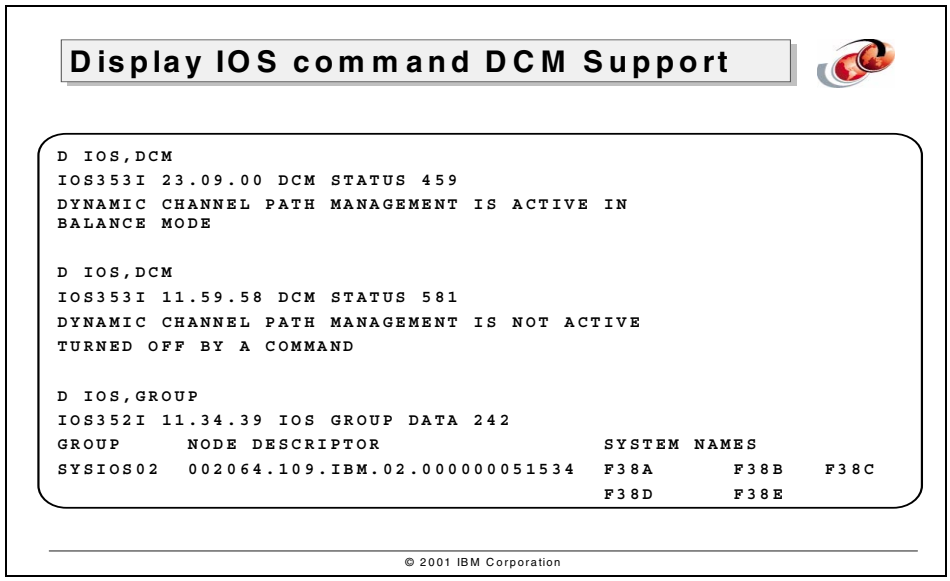
New support has been added to the CONFIGxx member to allow you to define which channels are to be managed, and whether DCM is allowed to use each port on the Switch.

The D M=CONFIG command lists any differences between the current active configuration and the configuration as it is defined in the CONFIGxx member. For example, in the figure above, there is a channel that is online, but according to the CONFIGxx member it should be offline, another that is offline that should be online, and some Switch ports that are not in the desired state.

In a DCM environment, variations from the desired configuration (such as an inoperative channel) may not be as apparent as they are in a non-DCM environment. For this reason, we especially recommend that you define and use this member to check your configuration, at least after every IPL.

If it turns out that there are differences between the current configuration and the model configuration, you can use the CF MEMBER(xx) command to issue the commands to bring the current configuration back in line with that described in the CONFIGxx member. Before you do this, however, you *must* to be sure that the CONFIGxx member is accurate and up-to-date and that it will not undo any changes (including temporary ones) that were made since the last IPL.

12.1.5 D IOS commands



The Display IOS command has been enhanced for Dynamic Channel-path Management, adding the ability to display the status of DCM in this LPAR Cluster, and the systems that currently make up the LPAR Cluster.

The D IOS,DCM command shows the status of DCM on this system, and therefore in the LPAR Cluster. This command is a bit different from most commands you will be familiar with. Most commands display information about either a single system, or about the whole sysplex. This command displays information about a subset of systems in the sysplex (the LPAR Cluster).

Remember that DCM has to be in the same state in all the systems in the LPAR Cluster. So, it is either *on* in all systems in the LPAR Cluster or it is *off* for all the systems in the LPAR Cluster. If DCM is not active, the output from this command will give the reason why. The status of DCM in one LPAR Cluster has no impact on the status of DCM in other LPAR Clusters.

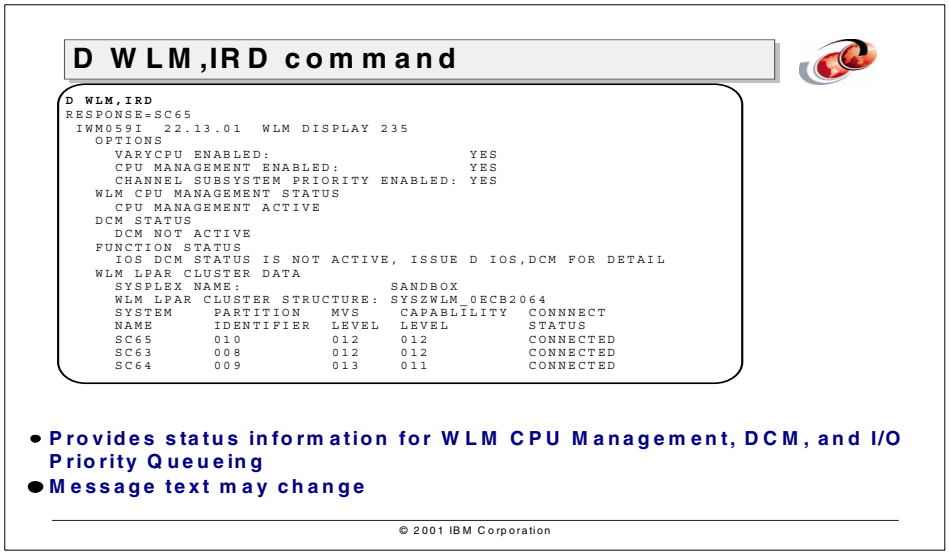
One thing to be aware of when using this command is that if you issue it while a system in the LPAR Cluster is in the middle of IPLing and has not yet joined the IOS XCF group. In this case, the other systems in the LPAR Cluster will be aware that a new member is about to join the group and will cease making any configuration changes until the new system successfully joins the group. During the interval, if you issue the D IOS,DCM command, you will get the following response:

```
IOS353I 16.07.29 DCM STATUS 145
DYNAMIC CHANNEL PATH MANAGEMENT IS NOT ACTIVE
CF CONNECTIVITY ERROR IN MULTISYSTEM CONFIGURATION
```

While this message may appear to indicate an error, if one of the members of the LPAR Cluster is IPLing, it is actually informing you that one of the members of the cluster is not yet connected to the LPAR Cluster CF structure, which is correct at this time.

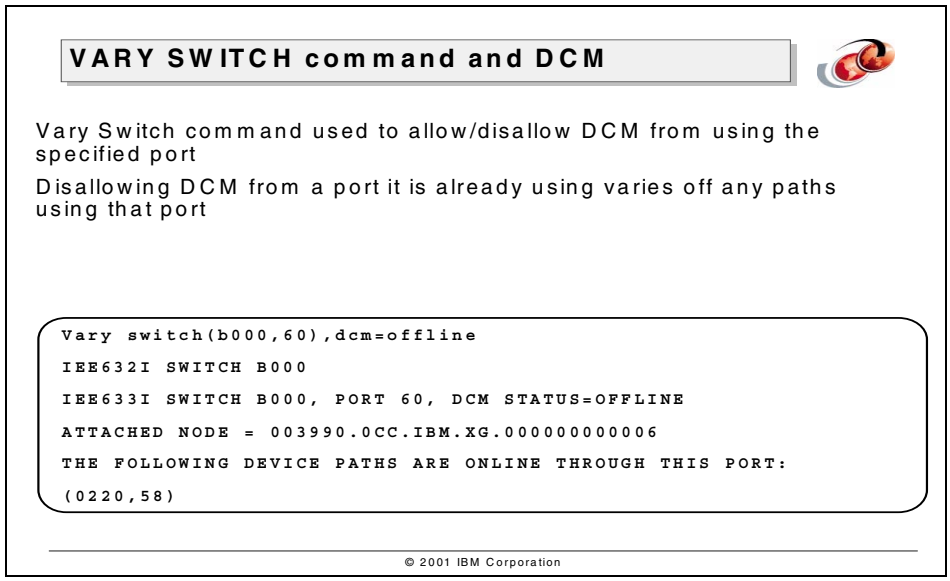
In addition, a list of all the systems in the LPAR Cluster can be displayed using the D IOS,GROUP command. The list of systems is based on the systems that join the IOS XCF group (SYSIOSxx) for this LPAR Cluster.

12.1.6 D WLM,IRD command



To provide more information about the IRD environment, APAR OW48601 provided a new IRD option on the D WLM command. This command is intended primarily to help diagnose perceived problems in IRD. It provides the status of each of the three IRD functions (WLM CPU Management, DCM, and Channel Subsystem I/O Priority Queueing) in the system the command is issued on. The example output shown in the figure above is from a development version of this support—the actual contents of the response may change.

12.1.7 VARY SWITCH command



The new VARY SWITCH command is used to tell Dynamic Channel-path Management on all systems in the LPAR Cluster whether it is allowed to use the specified port on the Switch or not. It is only necessary to issue the command on one system in the cluster—IOS looks after routing the command to all systems in that cluster.

Specifying DCM=ONLINE tells DCM that it can use that Switch port to set up a path between a managed channel and the attached control unit.

Specifying DCM=OFFLINE will stop DCM from setting up a managed path using this port. If it is already using this port for an existing managed path, the associated path will be varied offline and removed from the configuration for the affected LCU.

One thing to be aware of is that the VARY SWITCH command cannot be issued from the COMMNDxx Parmlib member. The reason for this is that commands in that member get issued before DCM processing has been started, so there is no DCM switch table to update at that point.

In addition, if the port is being used by any non-managed paths, the response to the Vary Switch command will provide the device numbers and CHPIDs for those paths on all the systems *in the LPAR Cluster*. If you wish to take the port physically offline, perhaps for an engineer to do some maintenance, then you can use this information to manually do a Vary Path Offline on each system for those

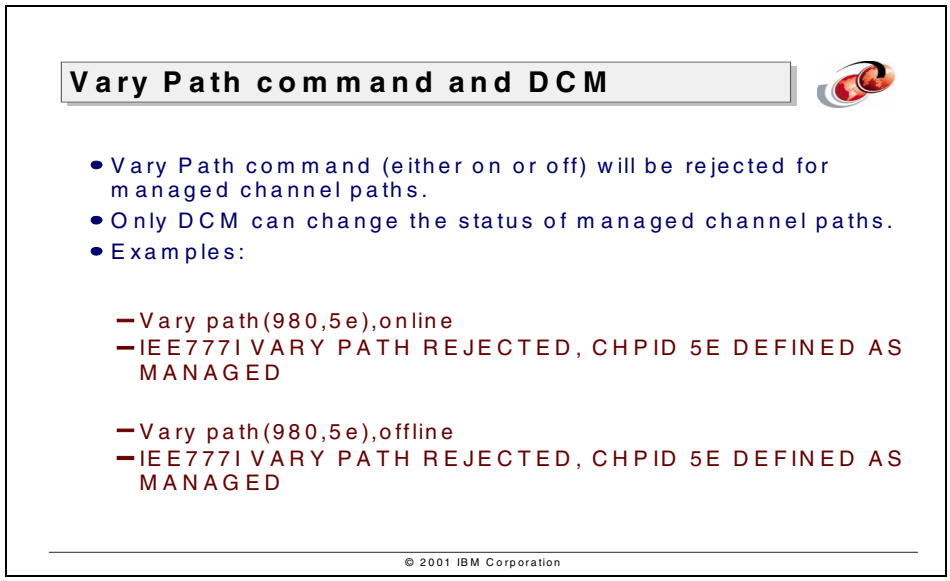
non-managed paths. However, remember that the scope of the Vary Switch command is only a single LPAR Cluster - if there are other systems outside the LPAR Cluster that are also using the port, it is your responsibility to identify them and act accordingly.

You should be careful when changing a port status to Blocked or Prohibited. If you Block or Prohibit a port that is connected to a managed channel, you should configure the CHPID offline before changing the port status. Once the channel is offline, DCM will automatically remove that channel from the configuration of any control units it is attached to. If you are Blocking or Prohibiting a port that is connected to a managed control unit, you should first use the Vary Switch DCM=OFFLINE command to get DCM to discontinue using that port. When ports are un-PROHIBIT'ed or un-BLOCK'ed, these operations need to be followed, as necessary, by VARY SWITCH commands to bring ports ONLINE to DCM. This only needs to be done for managed subsystem ports that are affected by the un-PROHIBIT or un-BLOCK functions.

See 10.2.2, “Other software requirements” on page 274 for more information about the use of System Automation for OS/390 (SAFOS) in a DCM environment.

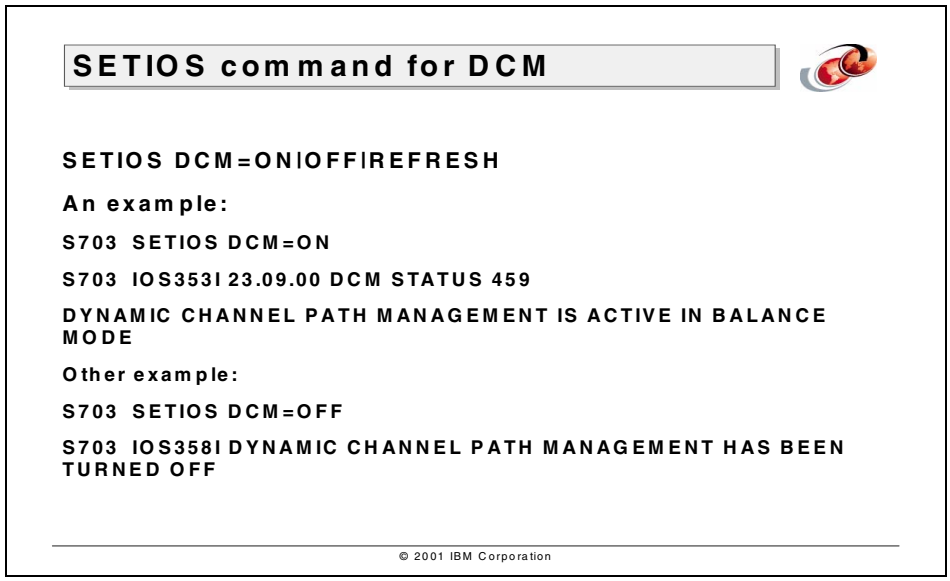
Note that the Vary Switch command does not make any change to the configuration of the Switch itself: all it does is update information maintained in DCM about whether it can use the port or not. As such, it does not replace the function provided by the I/O Operations component of System Automation for OS/390 or the ESCON Director console. This command only works for ports that are connected to a control unit. The ports that connect to managed channels cannot be taken offline to DCM using this command.

12.1.8 VARY PATH command



Because Dynamic Channel-path Management is solely responsible for the managed paths, you cannot use the Vary Path command to either vary on or vary off a managed path. If you try to use this command with a managed path, the command will be rejected as in the example above. If you have System Automation for OS/390 with the appropriate APAR (OW47198), we recommend that you use this to manage all your channel paths. It knows which paths are managed and which are non-managed and issues the appropriate commands in each case.

12.1.9 SETIOS command



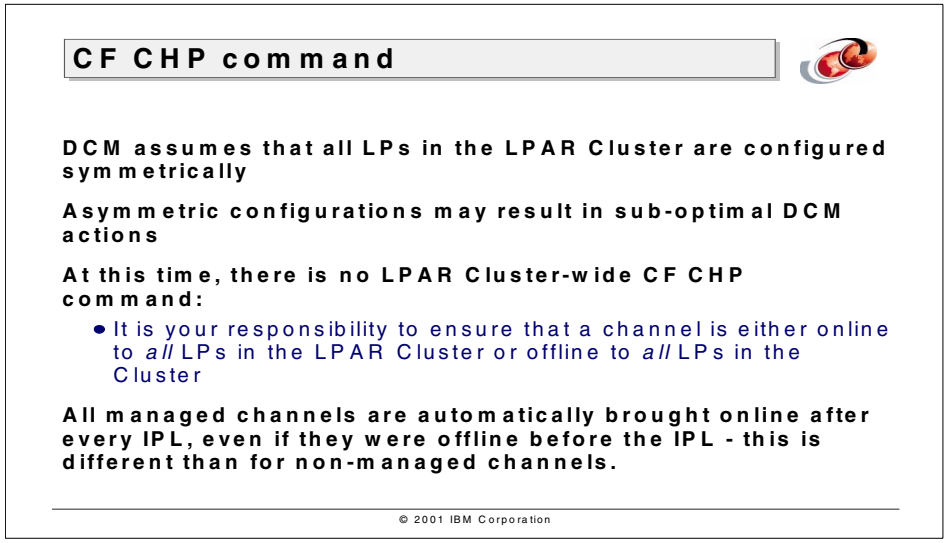
The SETIOS command with the DCM= option can be used to turn DCM on or off.

For the first system to be IPLed into the LPAR Cluster, Dynamic Channel-path Management will automatically be started after the IPL, assuming the required prerequisites are in place. If this is not the first system in the LPAR Cluster, then the status of DCM after the IPL depends on the status of DCM in the previous systems in the LPAR Cluster.

If you wish to change the status of DCM after the IPL, use the SETIOS command. Don't forget that this command has an LPAR Cluster-wide scope, so turning it on or off on one system will produce the same effect on all the other systems in the LPAR Cluster.

In addition to stopping and starting DCM, the SETIOS DCM command with the REFRESH option causes DCM to reload the IOSTmmm modules. This can be used if you wish to add a new control unit type non-disruptively, and want to update the table containing information for detecting points of failure for that manufacturer.

12.1.10 CF CHP command



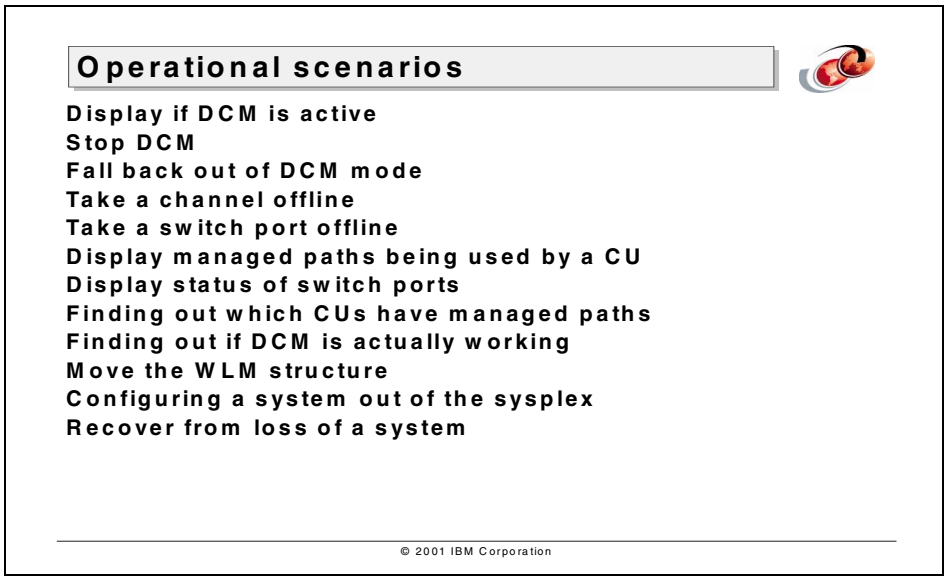
The access that each LP has to a given channel can be controlled by configuring the channel (via its CHPID number) on or offline to that LP. It is possible for a channel to be online to some LPs in a CPC, but offline to others. In fact, all managed channels that can be used by an LPAR Cluster will appear to be offline to all LPs that are not in that LPAR Cluster.

Prior to Dynamic Channel-path Management, it was possible to have a channel offline to one LP and online to another, with no performance consideration except that one LP would have access to more bandwidth to the attached devices. In a DCM environment, it is recommended that a channel should either be online to *all* LPs in the LPAR Cluster or offline to *all* those LPs. The reason for this is that the DCM algorithms assume that all LPs are configured symmetrically. So, if one LP has 6 paths to a device, the algorithms assume that all the LPs in the LPAR Cluster have 6 paths. The decisions that it makes, regarding whether more or fewer channels are required, are based on that assumption. If you create a configuration that is not symmetric (by configuring a channel offline to just a subset of systems in the LPAR Cluster), the decisions made by DCM may be sub-optimal.

There is no LPAR Cluster-wide CF CHP command, and taking a managed channel offline to one LP does not cause DCM to stop using that channel in other LPs in the LPAR Cluster.

There is one other consideration you should be aware of. For non-managed channels, if the channel was offline to a system prior to it being IPLed, it will still be offline after the IPL. However, all managed channels that can be used by an LP are automatically brought online after the IPL, regardless of whether they were online or offline before the IPL.

12.2 Operational scenarios



In this section, we go through the operator tasks listed in the figure above, and show how each should be carried out using the new operator commands and any other tools that may be required.

Display DCM status

Use the D IOS,DCM command to find out if DCM is started or stopped, and if it is operating in DCM Goal or Balance mode.

Stop DCM

Use the SETIOS DCM=OFF command to stop DCM. The effect of this command is to stop any further changes. It does *not* reset the configuration to its initial state.

Fallback out of DCM mode

The only way to turn managed channels and control units back into non-managed channels and control units is to revert to an IODF where those channels and control units are defined as non-managed. This is discussed in 10.7, “Backout plan” on page 298.

Take a channel offline

To take a channel offline, whether it is a managed or a non-managed channel, issue the CONFIG CHP(xx),OFFLINE command. It is not necessary to issue a V PATH or a V SWITCH command before taking the channel offline. Part of the CONFIG CHP processing is to take the path offline, regardless of whether it is a managed or a non-managed path. In addition, for a managed channel, that channel will be removed from the configuration for that device

Take a Switch port offline

To take a Switch port offline, perhaps for maintenance, you have to stop use of that port by both managed and non-managed paths. Remember, as we discussed in 10.1.4, “Switch considerations” on page 267, that a given port can be used concurrently by both managed and non-managed paths.

It is very easy to stop the managed paths from using the port: you use the Vary Switch(ssss,pp),DCM=OFFLINE command. If there are any existing managed paths using that port, they will be varied offline and removed from the configuration of any managed control unit attached through that ports. This command will also stop Dynamic Channel-path Management in that LPAR Cluster from selecting this port for any new paths it might be setting up.

To stop the non-managed paths, issue a Vary Path Offline command for each non-managed path that is currently using the port. To identify *all* those paths, you should use whatever mechanism you used prior to the introduction of DCM. Remember that the port is probably also used by systems outside the LPAR Cluster, so you will have to vary all those paths offline as well.

If you use the I/O Ops component of SAFOS, and the APAR for DCM support is applied, you can use it to control the whole process of varying paths and ports online and offline.

Display managed paths used by a control unit

To display the managed paths used by a control unit, issue the D M=DEV(dddd) command for one of the devices behind the control unit. This will provide a list of *all* the paths used to access the control unit. The output from the command shows which paths are managed and which are non-managed. It also shows the maximum number of managed paths.

Display the status of Switch ports

The ESCON Director console or the I/O Ops component of SAFOS can be used to display physical information about the ports on the Switch. However, this does *not* provide any information about the DCM status of the ports. It *does* tell you which ports are blocked, and which are prohibited from communicating with which other ones.

When the second phase of HCM support for Dynamic Channel-path Management is available, it will provide information about the Switch, both from a physical and a DCM point of view, from a single place.

Finally, the new D M=SWITCH command can be used to display some physical information about the switch configuration. However, it also displays information about the DCM status of each port: whether the port can be used by Dynamic Channel-path Management or not, and if not, why. The D M=SWITCH command, when issued for a specific port, also provides the node descriptor of whatever device is connected to that port.

Find out which control units have managed paths

There is no easy way to find out from the console which control units are managed. You can find out which channels are managed, and then find which devices are attached to each of those channels (using the D M=CHP(xx) command). There is no D M=CU command, or anything like that that will give you a summary of all the control units.

If you need to determine which control units are managed, the best option is to use HCD to view the active IODF, and determine from that which ones are managed.

Find out if DCM is actually doing something

As DCM runs, and adjusts your configuration, it does not issue any messages to show you what it is doing. As a result, you may find yourself wondering if it is actually doing anything. The easiest way to find out if you are using your managed channels is to use the RMF Monitor II Channel Path Activity Report. At the top of that report, there should be a line that has a Channel Path Type of CNCSM - this is a summary line for all the managed ESCON channels. If there is activity on that line, you know that DCM has added at least some of the managed channels to the configuration of some of the managed control units, and those channels are being used. For each of the managed channels, there is still a line in this report, and the Channel Path Type for those will also be shown as CNCSM.

You can also use the RMF Monitor II I/O Queueing Activity report. This gives you a report by LCU, and for each LCU it provides a list of the channels used to access that LCU, and whether each of those paths are managed or not. It also shows the minimum and maximum number of managed channels on this control unit during the interval, as well as the maximum number possible, as defined in HCD.

Move the WLM Cluster structure

There may be times when you wish to move the WLM Cluster structure from one CF to another. In case of a planned configuration change, you can simply use the SETXCF START,REBUILD command, with the LOC=OTHER keyword to move the structure from the CF it is in. This process should be completely non-disruptive.

If one system in the LPAR Cluster loses access to the WLM Cluster structure, WLM does not automatically rebuild the structure. In this case, DCM stops making any further changes to the configuration for this LPAR Cluster. You should manually rebuild the structure to the alternate CF. When the rebuild completes, DCM will automatically become active again.

If all systems in the LPAR Cluster lose connectivity to the structure, a new, empty structure will be automatically allocated, and all DCM actions will be suspended until the new structure has been populated with information from all the systems. In this case, there is no manual intervention required.

Configuring a system out of the sysplex

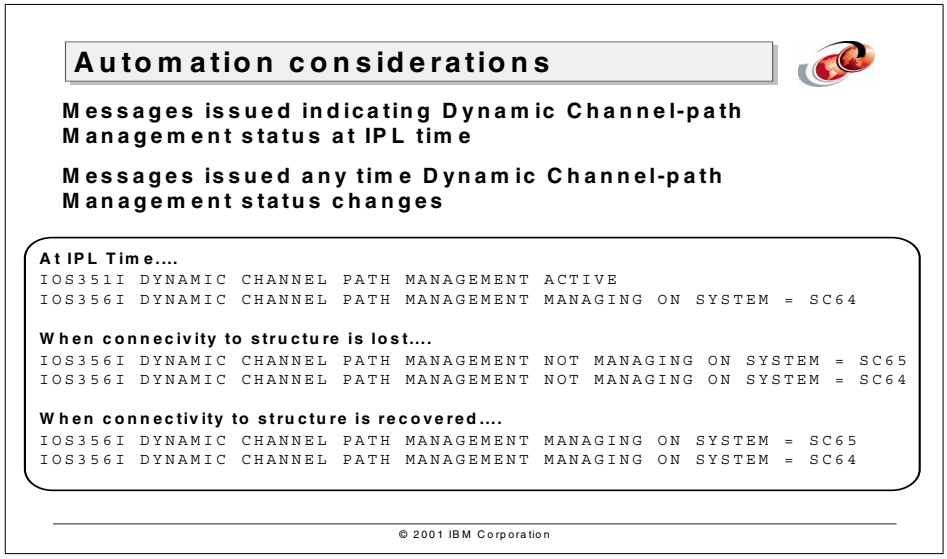
Care must be taken when using the VARY XCF, systemname, OFFLINE command to VARY a system out of a sysplex in the middle of a planned ACTIVATE of a new I/O configuration. If the system is VARY'ed out of a sysplex (or enters a non-restartable wait state) while in this state, the configuration will be left in an invalid state. This condition may go undetected until the D IOS, CONFIG command is issued and the response indicates ACTIVATE RECOVERY REQUIRED. In this state, if DCM is active, the function will be inoperative until ACTIVATE RECOVERY completes and again sets the configuration state valid.

Recovery from the loss of a system

There are no specific considerations relating to the loss of a system. If just one system in the LPAR Cluster fails, DCM actions in the remaining systems will continue. When the failed system is re-IPLed and brought back into the LPAR Cluster, it will detect what paths are being used by the other systems in the cluster, and start using those paths. No manual interaction is required to re-enable DCM.

This description assumes that the PTF for APAR OW48164 is applied—if the PTF is not applied, DCM does not automatically re-synchronize its Switch Table after IPLing into an existing LPAR Cluster. Rather than getting into a discussion of the considerations for such a situation, it is simpler just to ensure that fix is applied.

12.3 Automation considerations



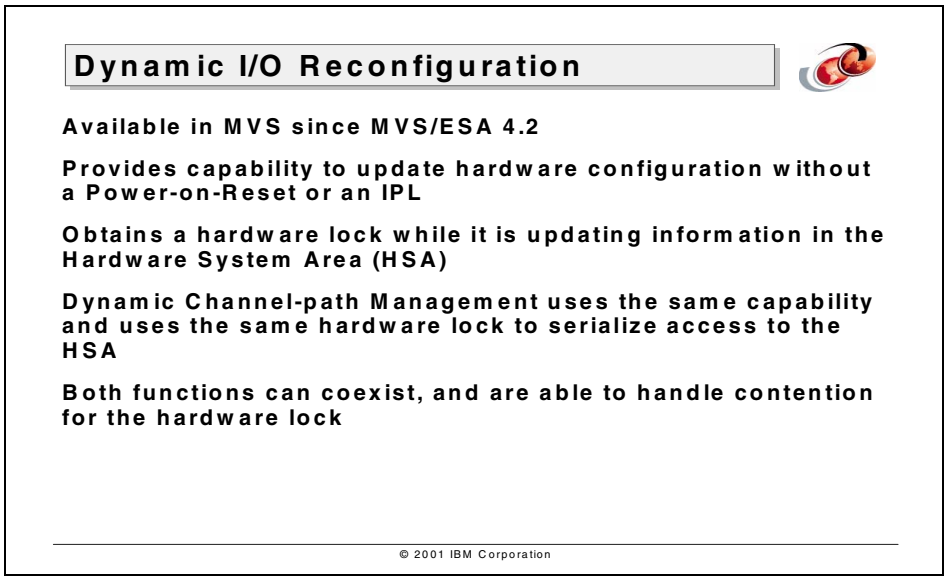
You will probably wish to add information to your Automated Operations product to enable it to monitor the status of Dynamic Channel-path Management. New messages are issued at IPL time and whenever the status of DCM changes. There are also messages issued whenever DCM adds or removes a managed channel to the configuration of a control unit. These messages are informational only and do not need to be monitored by the automation package.

When the system is IPLed, you will receive messages from WLM informing you that it has connected to the WLM LPAR Cluster structure. These messages will be followed by IOS351I and IOS356I messages indicating the status of DCM. The text of the IOS356I message is variable, so you will need to get the automation to check the message text to determine the precise status.

Should the system lose connectivity to the CF containing the WLM LPAR Cluster structure, you will receive some WLM messages indicating that connectivity to the structure has been lost. This will be followed by an IOS356I on each system in the LPAR Cluster indicating that DCM is no longer active. Remember that, if DCM is using the CF structure, all systems must have connectivity in order for DCM to be enabled. This is different to WLM LPAR CPU Management, where the remaining systems in the cluster can continue managing weights between them.

When all systems once again have connectivity, another IOS356I message will be issued on every system, informing you that DCM is active once again.

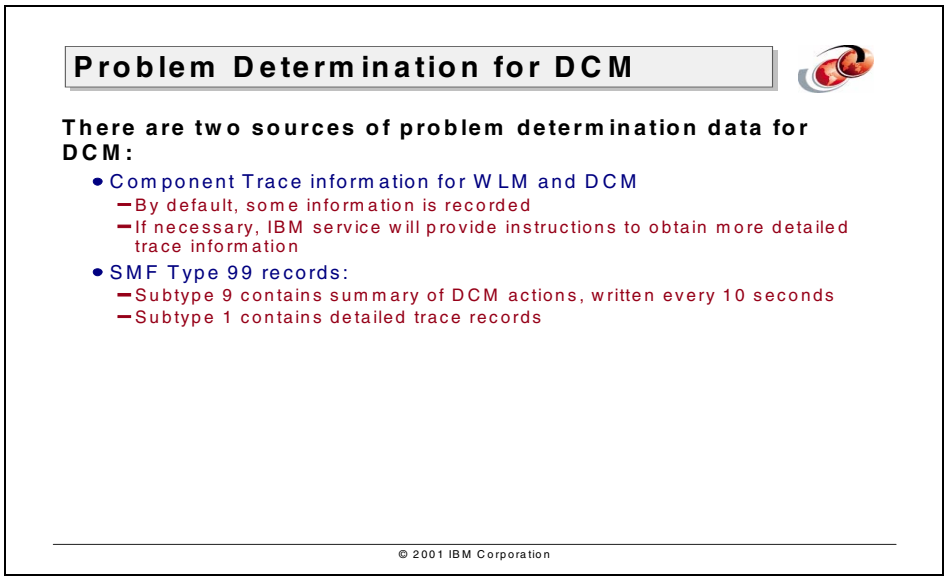
12.4 Dynamic I/O reconfiguration



Both Dynamic I/O Reconfiguration (using the ACTIVATE command) and Dynamic Channel-path Management update HSA, and the S/390 architecture dictates that only one task can update HSA at a time. This serialization is ensured by a task entering configuration mode—this obtains a hardware lock, and only one task may obtain that lock at a time on that CPC. DCM has been designed to allow for this, and will automatically recover from a situation where it cannot update HSA because another task is currently making an update.

The Activate command (when doing a Hardware Activate) also has support to wait if another task is currently updating HSA. In certain cases, it may be possible that the Activate command will time out, while waiting to get access to HSA. If this happens, the return code will indicate that the command failed because it could not get access to the resource, and you can simply issue the command again. It is *not* necessary to stop DCM while you are doing Hardware Activates.

12.5 Problem determination



In case of problems with Dynamic Channel-path Management, there are new types of diagnostic information available to help you work with IBM to diagnose and rectify the problem.

The first of these are enhancements to the information collected by the Component Trace (CTRACE) facility. By default, CTRACE will collect a certain amount of information about both IOS and WLM activity. In case that information is not sufficient, more detailed trace information can be obtained using parameters that will be provided by your IBM software service representative. Component Trace is described in *z/OS MVS Diagnosis: Tools and Service Aids*.

In addition, a new SMF record is produced that provides a summary of DCM actions. This new record, the SMF Type 99 subtype 9 record is produced once every 10 seconds. There is currently no IBM-provided tool to analyze this record.

Finally, the SMF Type 99 subtype 1 records contain WLM trace data that may be used to diagnose WLM actions. We do not advise that any installation should collect these records on an ongoing basis, due to the volume of data produced. Fifteen minutes of this information is always available in the WLM address space, even if you have the records turned off in SMFPRMxx. The trace codes are described in *z/OS MVS Planning: Workload Management Services.* A subtype 1 record is written every 10 seconds.

13

Performance and tuning for DCM

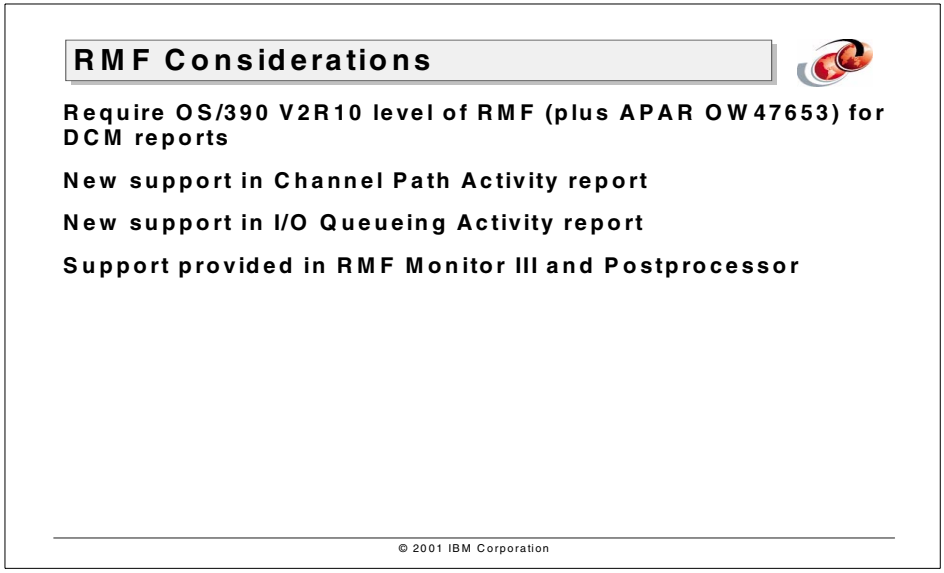
Dynamic Channel-path Management is specifically designed to remove a lot of the work involved in monitoring and tuning your I/O configuration. By dynamically changing your configuration to move less utilized channels to busy control units, it should lead to reduced channel contention and more even channel utilization.

However, there will still be a requirement to intermittently monitor the performance of DCM, and answer questions such as:

- What is the overall average utilization of the managed channels? If it is significantly higher than the average utilization across all non-managed channels, perhaps you should convert more non-managed channels to being managed.
- What is the channel utilization overall? If it is high across all channels, maybe you need to purchase additional channels.
- What is the Pend time by LCU? If one LCU has significant Pend time, can we assign more managed paths to that LCU?

The information to answer these and other questions is provided by RMF. We describe the enhancements to RMF in support of DCM in this chapter.

13.1 RMF considerations



In order to provide DCM performance data, the OS/390 V2R10 level of RMF (plus supporting APARs) provides enhancements to two existing reports that report channel and control unit utilization and performance. The same basic reports are available in RMF Monitors II and III, as well as the RMF Postprocessor. The enhancements are:

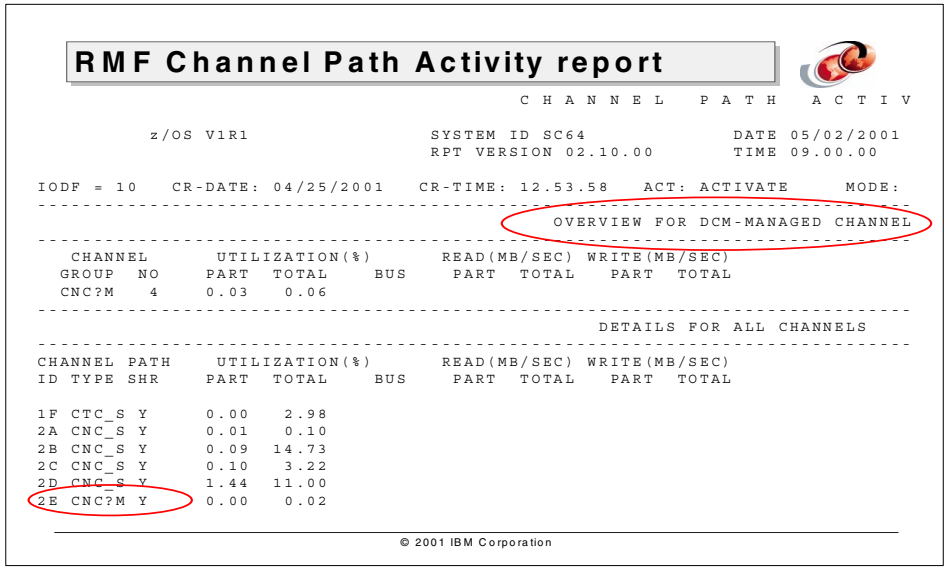
- A new summary section in the Channel Activity report
- New information related to system-managed CHPIDs in I/O Queuing Activity report

Also, new conditions have been added to the RMF Overview/Exception reporting capability.

Finally, new information has been added to the RMF SMF Type 73, 78, and 79 records in relation to managed CHPIDs.

All of these enhancements are described in the following pages.

13.1.1 Channel Path Activity report



RMF Channel Path Activity report

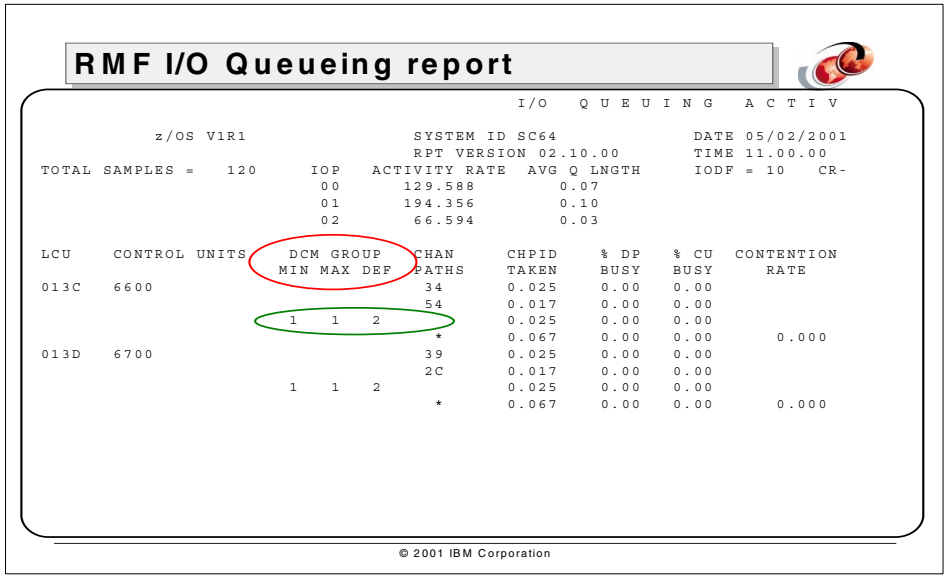
```
                                                C H A N N E L   P A T H   A C T I V

          z/OS V1R1                  SYSTEM ID SC64              DATE 05/02/2001
                                     RPT VERSION 02.10.00        TIME 09.00.00

IODF = 10   CR-DATE: 04/25/2001   CR-TIME: 12.53.58   ACT: ACTIVATE      MODE:
------------------------------------------------------------------------------
                                          OVERVIEW FOR DCM-MANAGED CHANNEL
------------------------------------------------------------------------------
   CHANNEL       UTILIZATION(%)       READ(MB/SEC) WRITE(MB/SEC)
  GROUP   NO     PART   TOTAL    BUS    PART   TOTAL   PART   TOTAL
  CNC?M    4     0.03    0.06
------------------------------------------------------------------------------
                                                DETAILS FOR ALL CHANNELS
------------------------------------------------------------------------------
CHANNEL PATH     UTILIZATION(%)       READ(MB/SEC) WRITE(MB/SEC)
ID TYPE SHR     PART   TOTAL    BUS    PART   TOTAL   PART   TOTAL

1F CTC_S Y      0.00    2.98
2A CNC_S Y      0.01    0.10
2B CNC_S Y      0.09   14.73
2C CNC_S Y      0.10    3.22
2D CNC_S Y      1.44   11.00
2E CNC?M Y      0.00    0.02
```


The Channel Path Activity report has been updated in support of Dynamic Channel-path Management. The main change is that you now get a summary section, at the top of the report, showing the total number of managed channels, and the average utilization across all managed channels. This helps you determine how intensively the managed channels are being used, and provides information to help you determine if you need more managed channels.

In addition to the summary section, information is provided about each of the managed channels separately, the same as for any other channel. Managed channels are indicated by a Channel Path Type of “CNC?M”.

Refer to z/*OS RMF Report Analysis*, SC33-7991, for information about the meaning of each of the fields in this report.

13.1.2 I/O Queueing Activity Report



The I/O Queueing Activity report has also been updated in support of Dynamic Channel-path Management.

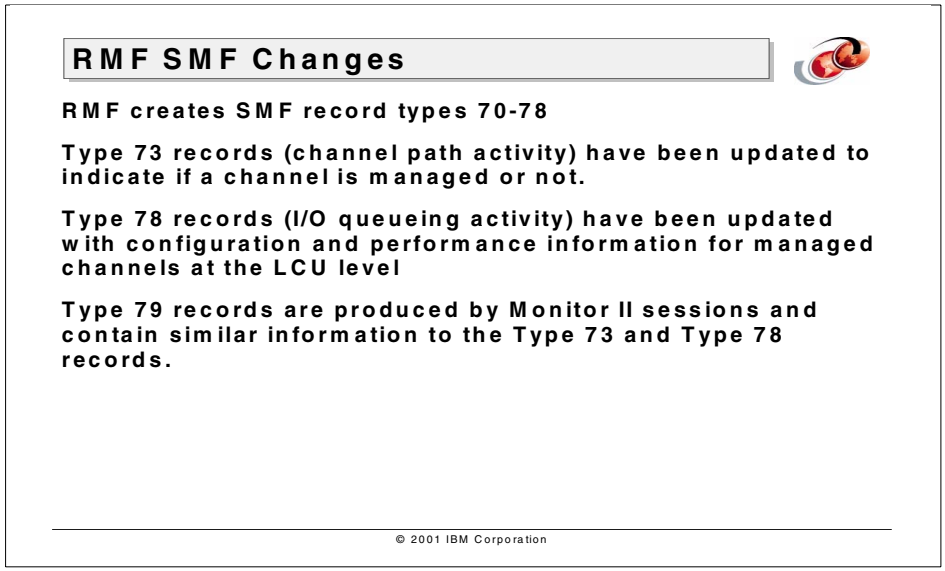
On the far right of the report (not shown in the excerpt above), there is a field entitled %ALL CH PATH BUSY. This field reports how long all the channels in an LCU were busy at the same time, indicating contention. This field should be monitored before and after DCM is implemented. Generally speaking, the values in this field should decrease after managed paths are added to the LCU (assuming the LCU did not have the maximum number of paths to start with).

Three new columns have been added that provide the minimum and maximum number of managed paths used by the LCU during the interval, as well as the maximum number of managed paths the LCU can possibly have, as defined in HCD.

The activity for all the managed channels is provided on the line that contains the Min, Max, and Defined number of managed channels. Information is not provided for each managed channel individually.

One thing to be aware of. If there is no activity against an LCU during an interval, that LCU will not be mentioned at all in the I/O Queueing Activity report. This is not an error—RMF deliberately skips LCUs that are idle.

13.2 SMF changes



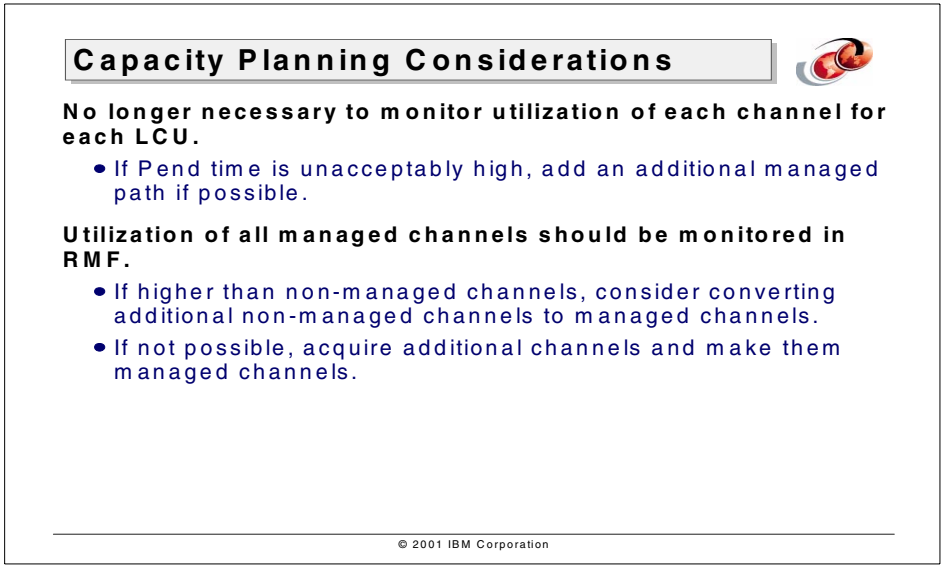
RMF creates SMF records in the range 70-79. These records are used by the RMF Postprocessor and other products. The records that have been updated in support of Dynamic Channel-path Management are:

Type 73 This record contains channel performance information. The record has been updated to add an indicator of whether a given channel is managed or not.

Type 78 The subtype 3 records contain I/O queueing activity information. They have been updated with information about whether each path is managed or not, and also the minimum, maximum, and defined number of managed paths for each LCU. They also contain accumulated performance information for all the managed paths on each LCU.

Type 79 These records are written during a Monitor II background session when feedback is requested as SMF records. There are many subtypes provided, but the subtype 12 contains channel path information, similar to that in the Type 73s, and the subtype 14 contains I/O queueing information similar to that in the Type 78s. The updates for DCM are the same as those made to the Type 73 and Type 78 records.

13.3 Capacity planning considerations



Because Dynamic Channel-path Management automatically moves channel bandwidth to the logical control units that are suffering from channel delays, you no longer have to spend as much time monitoring the channel utilization of each channel on each control unit.

Instead, you should now monitor the Pend time for each managed LCU, as reported in RMF. If the Pend time is unacceptably high, it may be possible to add more managed paths to that LCU, up to a maximum of 8 channels in total, and let DCM assign the capacity when it is required.

For channels, you should monitor the channel utilization of all managed channels as reported in the RMF Channel Path Activity report. If the average utilization of the managed channels is significantly higher than the non-managed channels, you may consider converting some more non-managed channels to being managed channels. If you do not have the ability to do this, you may need to purchase additional channels and make those managed.

IBM have a capacity planning product called Tivoli Decision Support (previously known as Performance Reporter for OS/390). However, at the time of writing, there is no specific support for DCM in this product.

Part 4

Channel Subsystem I/O Priority Queueing

14

Channel Subsystem I/O Priority Queueing

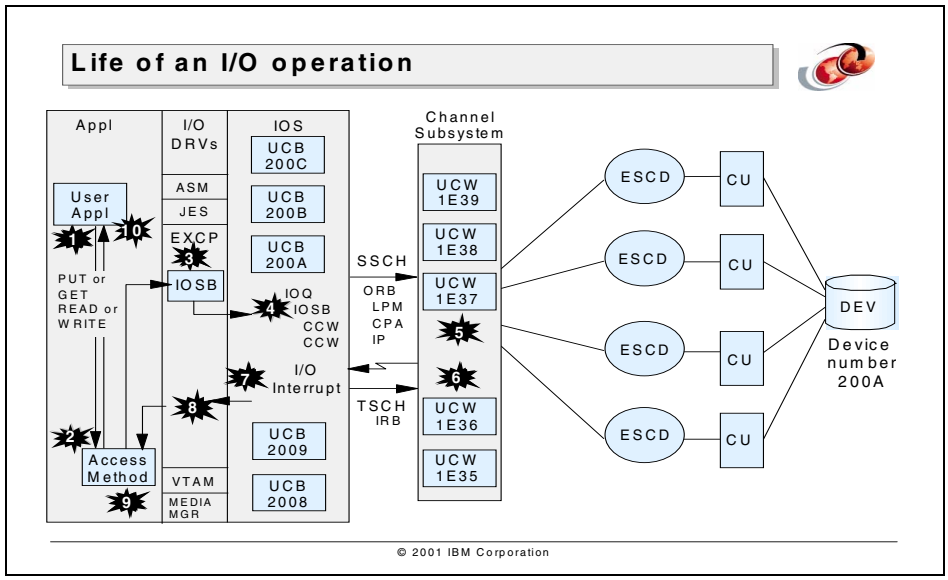
Channel Subsystem I/O Priority Queueing is a new capability delivered on IBM zSeries 900 and subsequent processors and exploited by z/OS. It is the third component of Intelligent Resource Director.

z/OS in WLM Goal mode uses this new function to dynamically manage the channel subsystem priority of I/O operations for given workloads based on the performance goals for these workloads as specified in the WLM policy. In addition, because Channel Subsystem I/O Priority Queueing works at the channel subsystem level, and therefore affects *every* I/O request (for *every* device, from *every* LP) on the CPC, you can also specify a single channel subsystem I/O priority that is to be used for all I/O requests from systems that do *not* actively exploit Channel Subsystem I/O Priority Queueing.

Of the three functions in Intelligent Resource Director, Channel Subsystem I/O Priority Queueing is the simplest and easiest to implement. Therefore, we have merged the “Introduction to” and the “How it works” chapters into a single chapter for this component, and we have merged the “Planning for”, “Implementing”, “Operating”, and “Performance and tuning” chapters into a single chapter, Chapter 15, “Planning & implementing CSS I/O Priority Management” on page 385.

Note: Even though most of the discussion in this chapter focuses on DASD I/O requests, Channel Subsystem I/O Priority Queueing actually applies to I/O requests to every type of device that is attached to the system, even for those attached via the old Parallel channels.

14.1 Life of an I/O operation



Before we get into explaining how Channel Subsystem I/O Priority Queueing works, it is helpful to review the basic concepts of the execution of an I/O operation. The same information is provided in 9.1.1, “Life of an I/O” on page 191, but for your convenience it is repeated here. The highlighted numbers in this graphic refer to the process steps listed below.

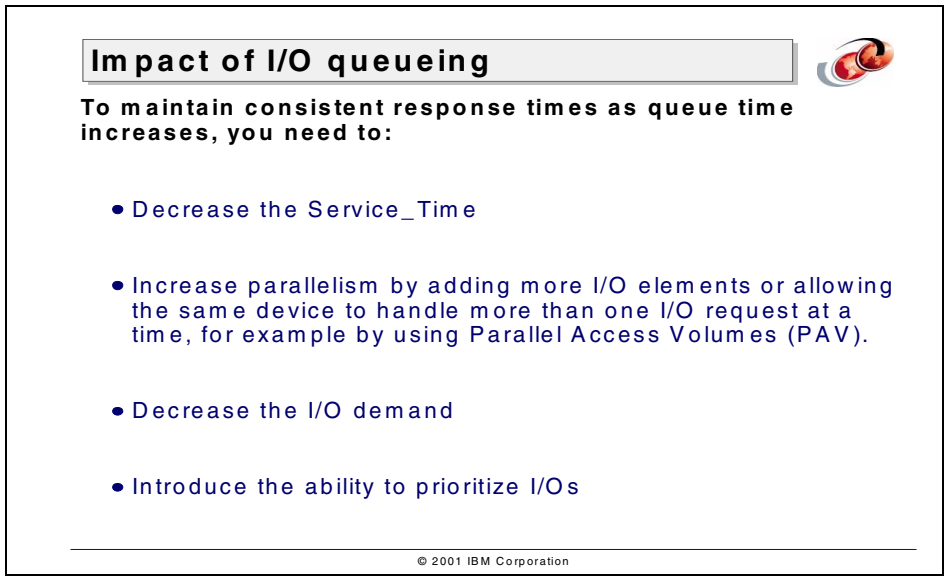
The main aim of any computing system is to process data obtained from I/O devices:

1. The user program begins an I/O operation by issuing an OPEN macro instruction and requesting either input or output of data using an I/O macro instruction like GET, PUT, READ, or WRITE, and specifying a target I/O device. An I/O macro instruction invokes an access method that interprets the I/O request and determines which system resources are needed to satisfy the request.

   The user program could bypass the access method, but it would then need to consider many details of the I/O operation, such as the physical characteristics of the device. The program would also have to create a channel program composed of instructions for the channel subsystem, and invoke the EXCP processor, an IOS driver, to handle the next phase of the I/O process.By using an access method, a user program maintains device independence.

2. There are several MVS access methods, each of which offers different functions to the user program. The selection of an access method depends on how the program plans to access the data (randomly, or sequentially, for example) and the data set organization (sequential, PDS, VSAM, and so on).
3. To request the movement of data, either the access method or the user program presents information about the operation to the EXCP processor by issuing the EXCP macro instruction. EXCP translates the information (CCW Command Chain Addresses and CCW Data Addresses) into a format acceptable to the channel subsystem, fixes the pages containing the CCWs and the data buffers, validity-checks the extents, and invokes the I/O Supervisor (IOS).
4. IOS places the request for I/O on the queue for the chosen I/O device in the UCB and issues the Start Subchannel (SSCH) instruction to send the request to the channel subsystem. At this point, the central processor can continue with other work until the channel subsystem indicates, with an I/O interrupt, that the I/O operation has completed.
5. The channel subsystem selects a channel path to initiate the I/O operation between the channel and control unit or device, and controls the movement of data between the channel and processor storage.
6. When the I/O operation is complete, the channel subsystem signals completion by generating an I/O interrupt.
7. IOS processes the interruption by determining the status of the I/O operation (successful or otherwise) from the channel subsystem using a Test Subchannel (TSCH) instruction.
8. EXCP or the user program indicates that I/O is complete by posting the access method and calling the dispatcher.
9. When appropriate, the dispatcher reactivates the access method.
10. The access method returns control to the user program, which can then continue its processing.

14.2 Impact of I/O queueing



Priorities only make a difference when there is a queue. This applies whether we are talking about queueing for access to the CPU, queueing to start an I/O, or queueing in the bank to get your money out! If you are the only one in the queue, it makes no difference whether you have the highest or lowest priority.

If there *is* a queue, then your priority makes a significant difference to the response time you will receive. Imagine if you were queueing in the bank over lunch time (the busiest time), and everyone that came in was able to get in the queue in front of you - your response would be somewhat different than when you were the only one in the queue!

Therefore, before we discuss I/O priorities, we will talk a little bit about I/O response times, the impact that queue time has on them, and the types of queues that an I/O request may encounter.

The response time for an I/O operation can be represented by the equation:

```
Response_Time = Service_Time + Queue_Time
```

If the Queue_Time starts to build up, affecting the Response_Time, there are several things you can do to try to bring the Response_Time back to the original value:

- You can decrease the Service_Time, offsetting the increased Queue_Time. This is achieved, for example, by adding more cache, or faster I/O elements such as channels, control units, and disks.
- You can increase parallelism by including more I/O elements (channels, control units, devices) or by allowing the same device to handle more than one I/O request at same time, using something like the IBM 2105 Parallel Access Volume (PAV) feature.
- You can try to decrease the I/O demand, but this infers that you will get less work done overall.
- Or you can introduce the ability to prioritize I/Os. This does not decrease the *average* Queue_Time but it does ensure that the more important I/O requests will experience reduced Queue_Time, resulting in better response times for *those* requests.

Therefore, it is clear that the effectiveness of prioritizing I/O requests depends on:

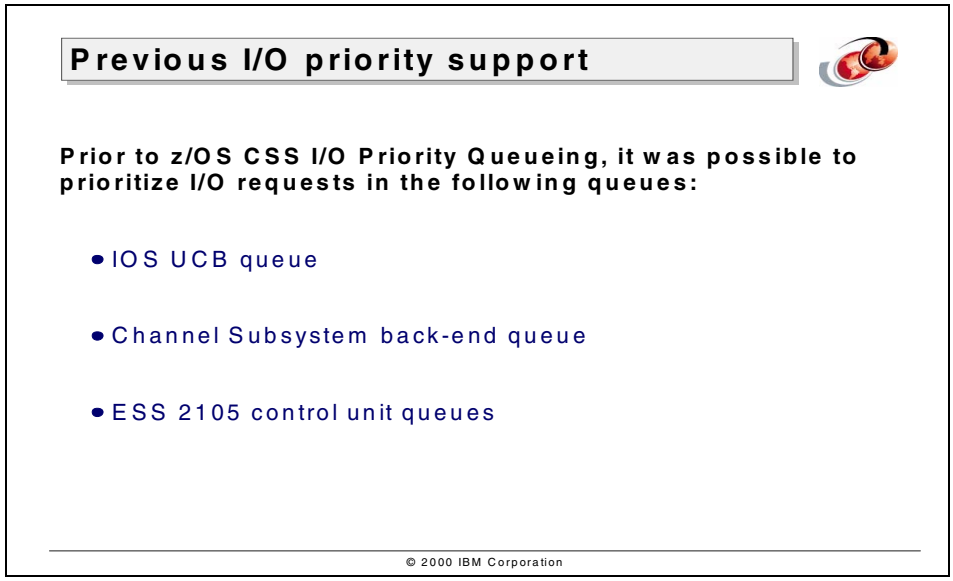
- Having a queue. Or, in other words, there must be regular occurrences where the queue depth is at least two.
- The I/O requests in the queue coming from transactions running in different SCPs, and having different importances.

Now, let us see where queues may exist during an I/O operation. There are a number of places during the execution of an I/O request where it can be queued because the resources necessary to execute the next phase of the request are not available. These queuing points include:

- The *IOS UCB queue*. This is a local queue, because it is per device and all requests come from the same operating system.
- Queues within the *Channel Subsystem (CSS)*. The CSS queues are global, because they are for all the devices connected to that CPC and, if the CPC is in LPAR mode, all the I/O requests coming from *all* the LP images contained in the CPC (this is a very important concept that we will come back to later). The CSS queues, and CSS processing, are discussed in more detail in 14.4, “Channel subsystem queueing” on page 366.
- The *control unit queue.* The control unit queue type is global because it applies to *all* I/O requests coming from *all* the LPs in *all* the connected CPCs. Here you may have two types of queues:
  - The one formed by the unavailability of control unit resources, such as: path to caches, disks and paths to disks. Here, the queues are organized by logical 3390/3380 device (for S/390 volumes). There are additional functions in the IBM 2105 that affect this processing, and these are discussed in 14.3.2, “ESS Multiple Allegiance” on page 361.

– The one formed by the unavailability of channels at reconnect time. When the I/O operation completes, the control unit must reconnect to a channel to present the status to the CPC. If no channels are available, a queue is formed of completed I/O requests that are waiting to present their status.

14.3 Previous I/O priority support



Some of the queues we discussed already support the ability to prioritize I/O requests:

- IOS UCB queue

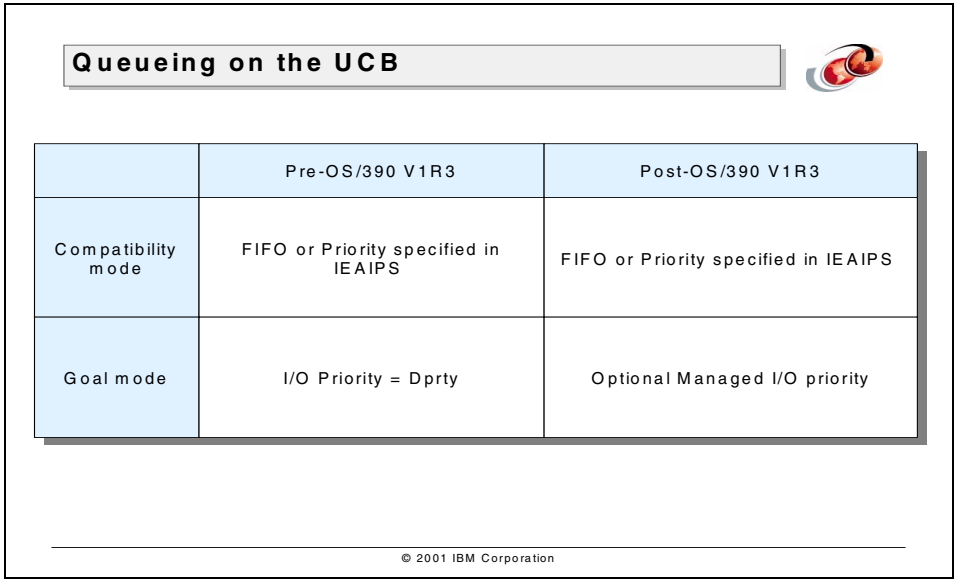
  MVS has supported priority queuing on the IOS UCB queue from the beginning. Based on a customer-specified option, I/O requests could be placed on the UCB queue in either:

  – First in-first out (FIFO) order.

  – According to the dispatching priority of the dispatchable unit (TCB or SRB) that issued the I/O request. This is controlled by specifying IOQ=PRTY in the IEAIPSxx member. This ensures that an I/O request from a high dispatching priority unit of work is queued on the UCB ahead of I/O requests from lower priority tasks.

  – According to a specific I/O priority. This is controlled by specifying IOP=xx (from 00 to FF) on the performance group period definition in the IEAIPSxx member.

  Prior to OS/390 1.3, these were the only algorithms available. OS/390 1.3 added the ability for WLM to manage the I/O priority. WLM management of I/O priorities is discussed in 14.7.1, “WLM management of I/O priority” on page 373. The available options are shown in the following figure.



- CSS queue (back end)

  The UCWs in the I/O interrupt queue are ordered by the numeric value of the Interrupt Sub Class (ISC). ISC is a field of the subchannel (UCW) and is stored there by the privileged instructions TEST SUBCHANNEL or CLEAR SUBCHANNEL, as requested by some z/OS components. ISC varies from zero to seven and has two functions:

  – It determines the priority of the UCW in the CSS I/O interrupt queue. This is used, for example, by the Auxiliary Storage Manager (ASM), a z/OS component responsible for doing paging I/O operations, to raise the priority of the I/O interrupts of any device containing a page data set.

  – It allows a selective filter in the CPU, when receiving an I/O interrupt. Each CPU has eight ISC bit masks in its control register (CR) 6. Only the I/O interrupts coming from I/O devices with an ISC value in UCW corresponding to an ON (enable state) bit in CR 6 are accepted. This function is not currently used by z/OS components.

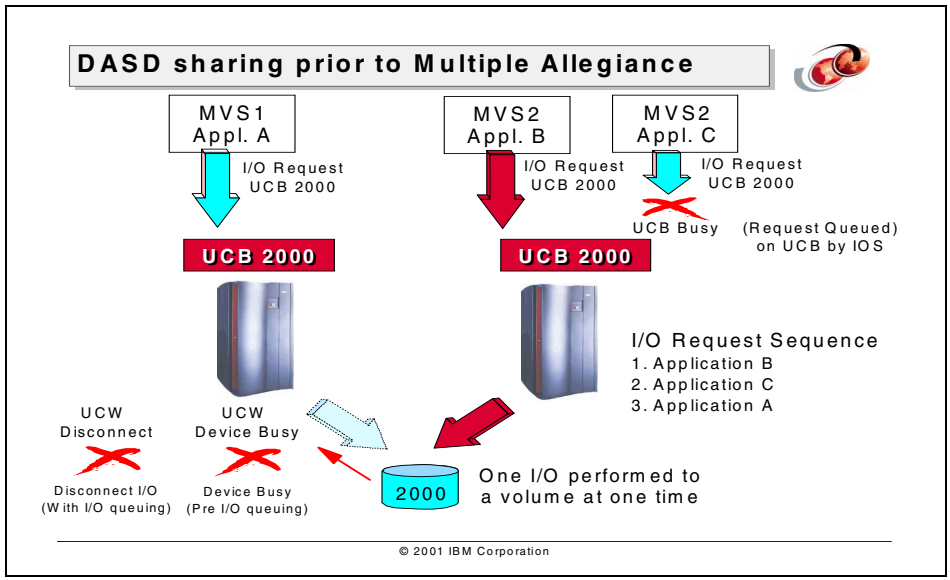
  The value of the ISC is set by the program responsible for initiating the request, and has nothing to do with any of the priorities set by WLM.

- IBM 2105 ESS Control unit

  This control unit is able to use the I/O priority passed by WLM (when in Goal mode) in the Define Extent CCW. However, the scope of this I/O priority is per volume, that is, it only applies to I/O requests accessing the same 3390/3380 device in parallel.

In the following six graphics, we review the IBM 2105 control unit. While you do not need to have a 2105 to use Channel Subsystem I/O Priority Queueing, the 2105 does have features and capabilities that make it relevant to a discussion about I/O processing and queueing.

14.3.1 DASD sharing prior to Multiple Allegiance

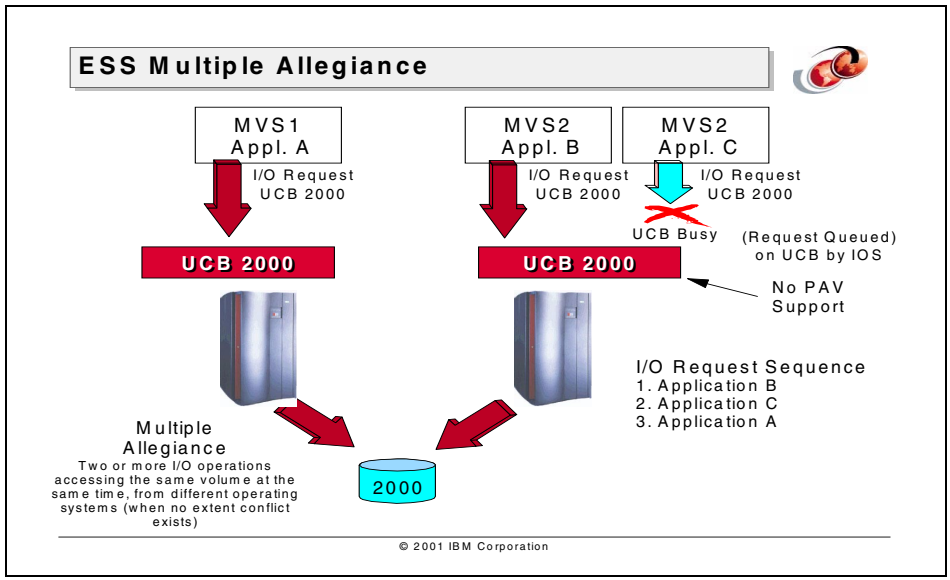


In this scenario, we have two MVS systems (MVS1 and MVS2) sharing a DASD device. Application B (in MVS2) makes the first request, and its I/O operation is started. When application C, which is also running in MVS2, requests data from the same device, the request is queued in IOS, because the UCB is marked busy (processing the request from Application B). IOS redrives this queued request when the I/O for Application B completes and the UCB becomes free.

Application A is running on MVS1, which has its own UCBs. When Application A requests data from the same device, the UCB is free (because no one else in this system is using that device), so the I/O request is passed to the CSS. When the channel attempts to connect to the device, the device is already busy, or it has allegiance to the channel path group (the set of channels accessing the control unit, coming from an MVS image) coming from MVS2. This also happens if there is an outstanding Reserve CCW from the channel path group coming from MVS2. On older control units, the control unit refuses the request, with “device busy” status, and the request is returned to the CSS where it remains pending. On recent control units, the request is accepted by the control unit and remains queued in the control unit after disconnecting from the channel. This queue time is reported in RMF as I/O Disconnect time.

However, even though the control unit accepted the request, only one I/O operation can be running on a device at a time. This restriction applies even if the data for both operations is in the control unit cache. The serialization takes place on the logical device, not the physical device.

14.3.2 ESS Multiple Allegiance



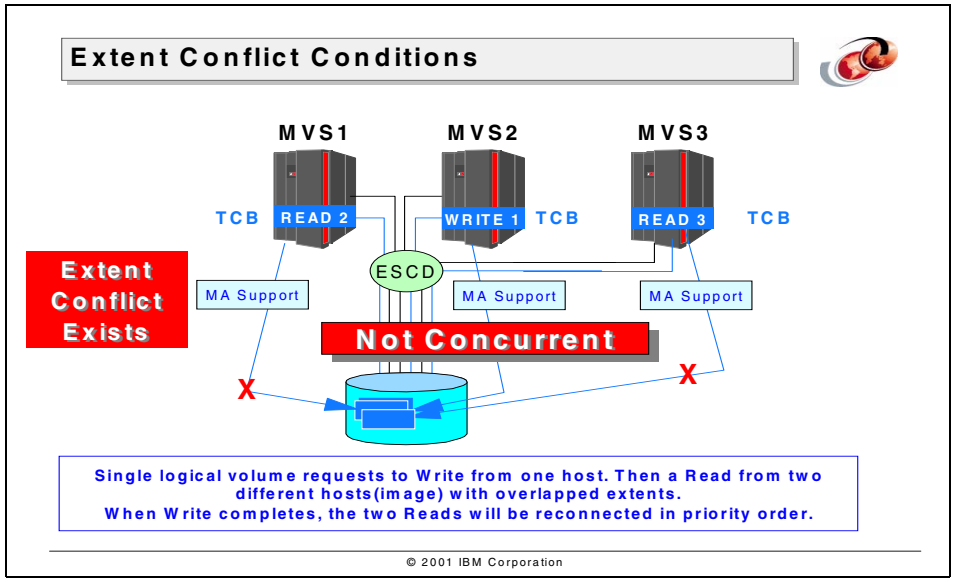
Now, if we take the same scenario we just described in 14.3.1, “DASD sharing prior to Multiple Allegiance” on page 359, but replace the old control unit with an IBM 2105 Enterprise Storage Server, there are two significant changes to the way the I/O operation is handled. The first of these is a feature known as Multiple Allegiance, and we will discuss this now. The other feature, Parallel Access Volumes, is discussed in 14.3.3, “ESS - Multiple Allegiance and Parallel Access Volumes” on page 364.

Multiple Allegiance provides the ability for the control unit to run multiple concurrent I/O operations to a device. Going back to our example, the I/O request from Application C is still forced to wait, because the UCB in MVS2 for the device is still marked busy. However, within the control unit, Multiple Allegiance allows many I/Os to be executed concurrently to the same DASD device. As a result, the I/O request from Application A is accepted by the control unit, and runs concurrently with the I/O request from Application B. This function is made possible because of the internal structure of a modern RAID control unit. In these control units, there is no longer a one-to-one correspondence between physical disk and logical 3390/3380 volumes. The tracks of the logical volumes are spread across a number of RAID disks, and each physical disk holds a portion of a number of logical volumes. Also, much more data is kept in the control unit cache than was the case with older control units. As a result, there is no physical limitation on doing multiple I/Os in parallel to the same 3390/3380 logical volume.

As part of the process of allowing multiple concurrent I/O requests, Multiple Allegiance guarantees that there are no extent conflicts with the other requests that are being processed at the same time. There are two cases of extent conflict:

- A channel program with write intent conflicts with all other channel programs, read or write, that are operating on the same set of extents. When an extent conflict exists, the conflicting I/O operation is queued within the control unit. This queue time shows in RMF as Disconnect time.

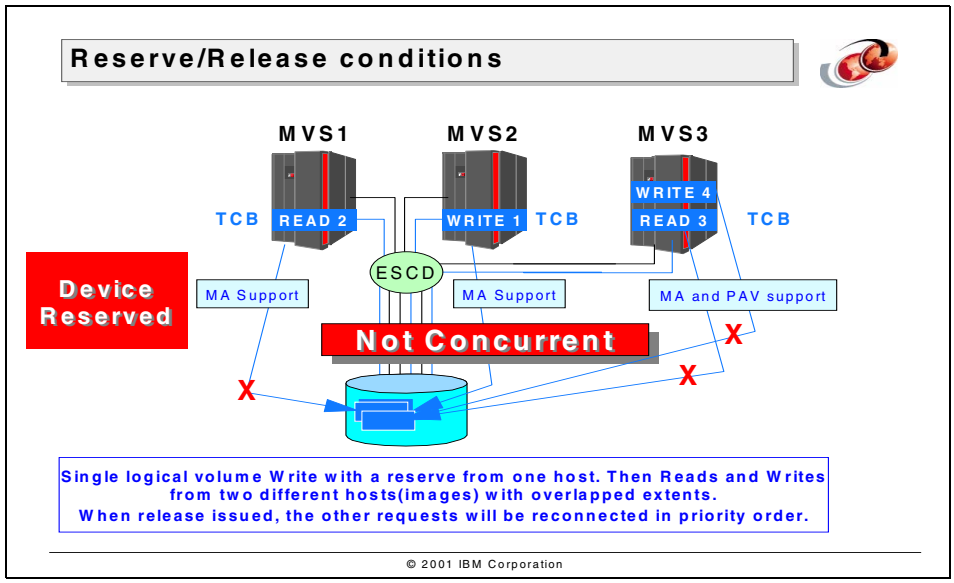
  When the extent conflict no longer exists, the I/O operations that were queued are now processed, and device end is presented to the CSS, indicating that the device can reconnect. The device end signal is presented by the ESS in an order that is based on the priority that is assigned by WLM.



- A Reserve conflicts with all other channel programs that access the volume from all other path groups, from different host images.

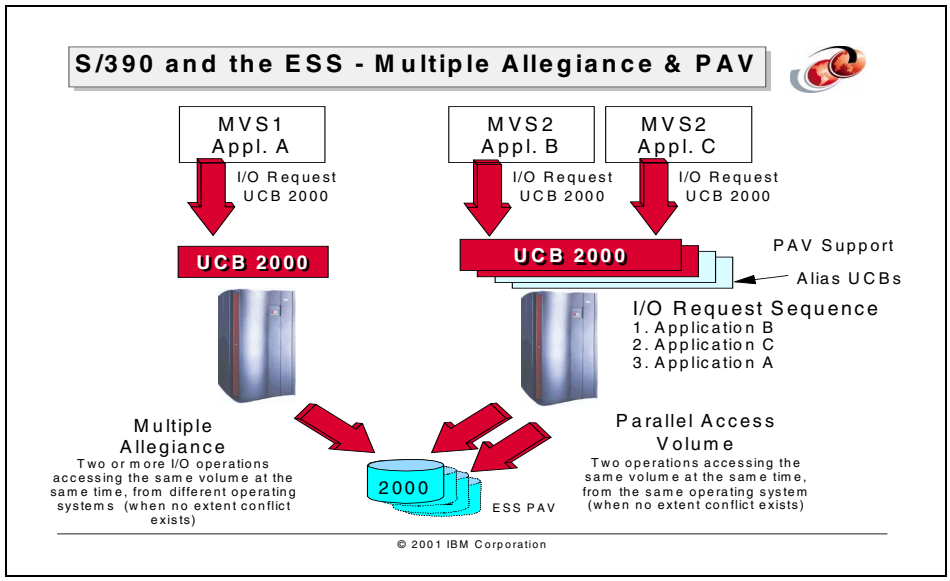
  When a Reserve exists on a device (logical volume) and any other normal I/O operation to the same device (logical volume) is received from a different path group (than the reserve request was received on), then the other operation is presented with a Device Busy status, and the I/O operation is queued in the CSS.

When the device is no longer reserved, the path groups for the device that were presented with device busy are now presented with a device end signal indicating (to the CSS) that the device is no longer busy. The device end signal is presented by the ESS in an order that is based on the priority that is assigned by WLM.



There is no specific support or anything that you have to define in the operating system to take advantage of this capability. In fact, the operating system is not even aware of what is happening.

14.3.3 ESS - Multiple Allegiance and Parallel Access Volumes



The other 2105 feature that we discuss here is called Parallel Access Volumes (PAV) support. PAV provides the ability for a single operating system image to drive multiple concurrent I/O requests to a single device.

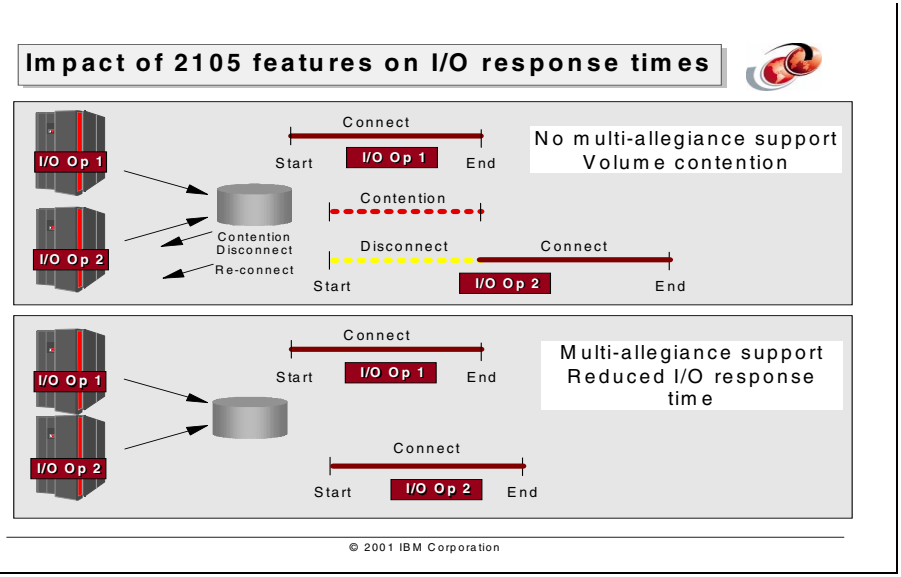
In our example, when we add PAV support, the I/O request from Application C is now processed and sent to the CSS as soon as it is received, rather than having to queue on the UCB waiting for the I/O from Application B to complete.

This capability is provided by adding Alias UCBs for a device. As a result, you can now run as many I/O operations in parallel to a device as you have Alias UCBs. The number of Aliases is initially defined in HCD. In addition, if you are in WLM Goal mode, you can optionally have WLM manage the movement of Aliases from one device to another, depending on the amount of queueing on the device at the time.

This can obviously have a significant benefit for the performance of busy devices. PAV can potentially nearly eliminate IOSQ time, so to see the benefit that it can provide, simply look in RMF at the IOSQ time of your busy devices.

Combining PAV with Multiple Allegiance support, it is now possible to drive many I/O requests from multiple systems, and drastically reduce the amount of queue time that was formerly associated with this processing.

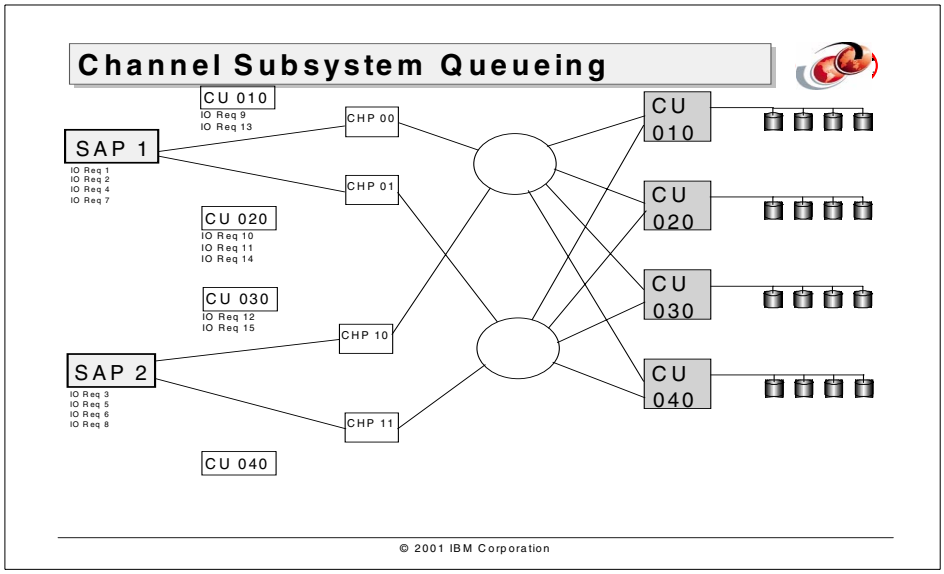
14.3.4 Impact of IBM 2105 features



The chart above shows the potential impact on response time that can be obtained by using the IBM 2105 Enterprise Storage Server. By allowing multiple operations to run concurrently, it nearly eliminates IOSQ time and queue time caused by a device being busy from another system.

We now come back to Channel Subsystem I/O Priority Queueing and how it addresses one of the other queueing points—within the channel subsystem.

14.4 Channel subsystem queueing



The S/390 channel subsystem is a highly tuned, very specialized part of the S/390 architecture. A single channel subsystem can efficiently handle the I/O requests generated by CPs capable of over 2 billion operations a second. I/O requests can be generated by any of up to 15 LPs, for communication to thousands of devices over up to 256 channels.

To handle this efficiently, the channel subsystem consists of a number of processors and queues. The I/O Processors (IOPs), also called System Assist Processors (SAPs), and the channels are both examples of processors. On the current technology, there is an affinity between a channel and a given SAP. Most devices will be connected to multiple channels, and the IBM configuration guidelines recommend configuring devices on channels that are spread across more than one SAP.

When an I/O request is sent from the operating system to the channel subsystem, the decision about which SAP to route the request to is based on information contained in the subchannel.

Each SAP has a work unit queue that contains all the Start Subchannel (SSCH) requests from all the LPs for a channel that is attached to that SAP. The requests in this queue are processed based upon the I/O Priority assigned either by WLM (if the LP is running z/OS with WLM in Goal mode) or by the hardware (if the LP does not exploit Channel Subsystem I/O Priority Queueing).

If there is an available channel to the device on this SAP, the request will be initiated. If there are no available channels on this SAP, the request is requeued to another SAP that also has connectivity to the target device. The request will be placed in the work queue for that SAP, and once again is processed based upon its I/O Priority.

If there is an available channel, and the request is initiated, the request can be successful, or it may fail because of Director Port (DP) Busy, Control Unit Busy, or Device Busy. In the case of DP or Control Unit Busy, the request is queued on another queue, and processed as soon as the control unit or switch is no longer busy.

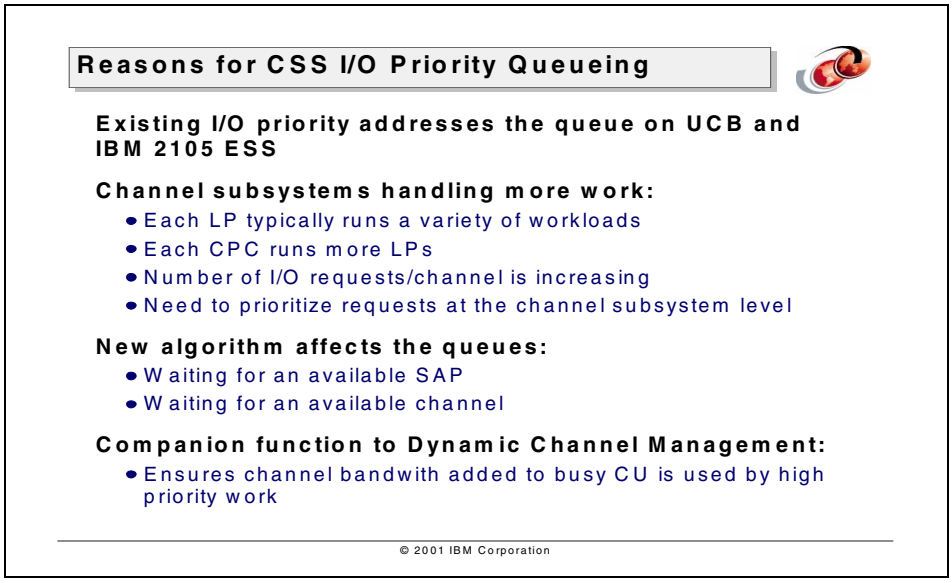
If the device is busy, the request is held until the control unit informs the channel that the device is no longer busy. The request is then resumed. At that point, the request either completes successfully, or it fails again and is placed on the appropriate queue.

Up until recently, the time spent waiting to be selected for the first time from the SAP work queue was not factored into RMF Pend time calculations. As a result, it was possible to see very high SAP utilization without this appearing to impact I/O response time. This has now been changed, and the Pend time reported by RMF includes all the time between when the SSCH instruction is issued and when the request is successfully started.

FICON Native and FICON Bridge channels have their own local queues. FICON Native channels select requests from the queue based on the I/O Priority of the request. FICON Bridge does not use the I/O Priority when selecting a request. More information can be found in the appropriate FICON documentation.

One other thing that may be of interest while we are discussing channels and SAPs, is the topic of balanced channel utilization. If a device has four channels to SAP 1 and four channels to SAP 2, and a given request is sent to SAP 1, all the channels attached to that SAP will be tried before the request is requeued to the other SAP. This provides the most efficient processing, and best response times; however, it may result in a skewed channel utilization. You should not be concerned about this skew as it generally means you are benefitting from more efficient SAP processing.

14.5 Reasons for Channel Subsystem I/O Priority Queueing



Channel Subsystem I/O Priority Queueing addresses the most significant remaining queueing points that up to now did not support the prioritization of I/O requests. Prior to Channel Subsystem I/O Priority Queueing, requests that were queued waiting for a SAP or waiting for an available channel were handled in a first-in, first-out (FIFO) manner. There was no discrimination between I/Os from discretionary workloads and production online workloads, or between I/Os coming from the Systems Programmer sandbox LP and a production LP.

In OS/390 1.3, WLM added the ability to manage the priority of I/O requests queued on the UCB based on the goals in the WLM policy. The same release added the ability to also pass this priority to the DASD control unit through the Define Extent CCW. The IBM 2105 Enterprise Storage Server is the first control unit that accepts such data and uses it as a priority.

The I/O priority used for the UCB queue and the control unit queue is calculated based on understanding which service class periods (SCPs) in the system are competing for the same *device*. When WLM changes an I/O priority, it assesses the impact of the change only on systems that are in the same sysplex and competing for the same resource.

However, neither of these functions has any effect on how I/O requests are handled when they arrive at the channel subsystem. If you consider that a large modern CPC is capable of driving many thousands of I/O requests per second, that results in a significant number of requests hitting the channel subsystem. Some of those requests will be from high importance workloads and some from lower importance workloads. Additionally, in an LPAR environment, some of the LPs may be running production workloads, while other LPs may be running test or development (lower importance) workloads.

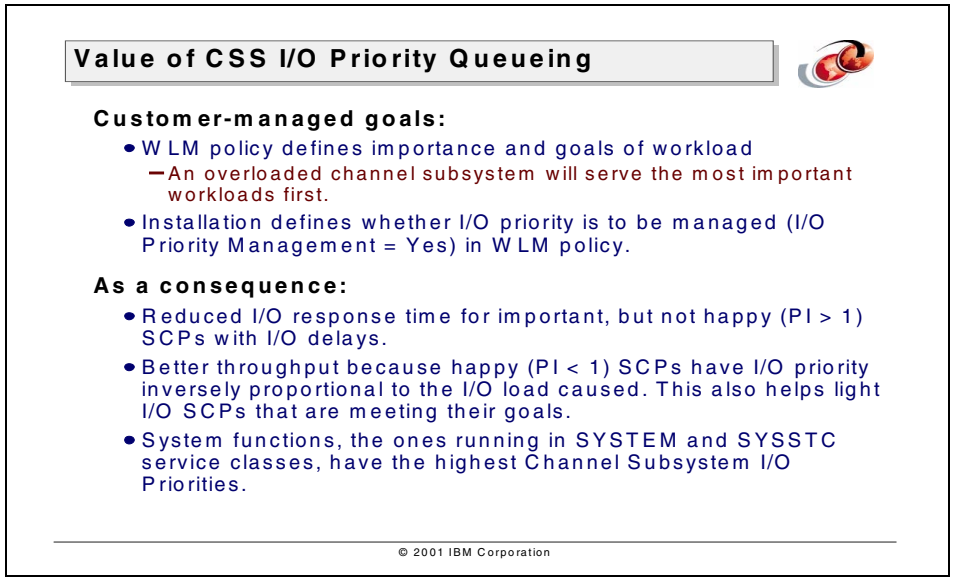
Prior to the IBM zSeries 900, there was no way, at the channel subsystem level, to differentiate between all these requests, meaning that high importance I/O requests may be delayed behind other, lower importance requests. And as time goes on, and the number of supported logical partitions per CPC and the processing power of the CPC increases, this situation would have become even more critical if it was not addressed. Channel Subsystem I/O Priority Queueing gives you the ability to differentiate between these requests at the channel subsystem level.

In addition to giving you more control over the order in which I/O requests are processed, Channel Subsystem I/O Priority Queueing is designed to be supportive of Dynamic Channel-path Management (DCM), discussed in Part 3, “Dynamic Channel-path Management” on page 167. DCM allows the channel bandwidth provided to control units to be managed based on the WLM goals of the SCPs using the control unit.

However, adding bandwidth to a given control unit provides more resources to *all* work using that control unit. If both high importance and low importance SCPs are using the same control unit, a large amount of bandwidth might need to be provided to the control unit to ensure the high importance work meets its goal, giving a free ride to the low importance work.

Channel Subsystem I/O Priority Queueing ensures that the added bandwidth benefits the higher importance workloads first, by prioritizing those I/O requests to be selected by the channel subsystem ahead of the requests from the less important SCP. Giving the high importance SCP a higher channel subsystem priority minimizes the bandwidth required by the high importance SCP to meet its goals, since the high importance SCP will not have to queue behind the requests from the low importance SCP.

14.6 Value of Channel Subsystem I/O Priority Queueing



The objectives of Channel Subsystem I/O Priority Queueing are the same as I/O Priority Queueing (in the IOS UCB and IBM 2105 controller) in general; that is, to cause a reduction in I/O response time for an important SCP that is experiencing I/O delays caused by other, less important, SCPs. This protects the customer's "loved ones" from degradation of response time, when other less important SCPs compete for I/O resources.

However, the CSS queues have some aspects that make them different from the IOS UCB queue: the CSS queues are global. That is, they apply to all the devices connected to the CPC and, if the CPC is in LPAR mode, *all* the requests from *all* the LPs in the CPC. If all the LPs in the CPC are in the same LPAR Cluster, and all of then are in WLM Goal mode, then all the requests will be prioritized relative to each other. However, the requests from any LPs that are *not* in the LPAR Cluster will have their priorities set independently of the other LPs.

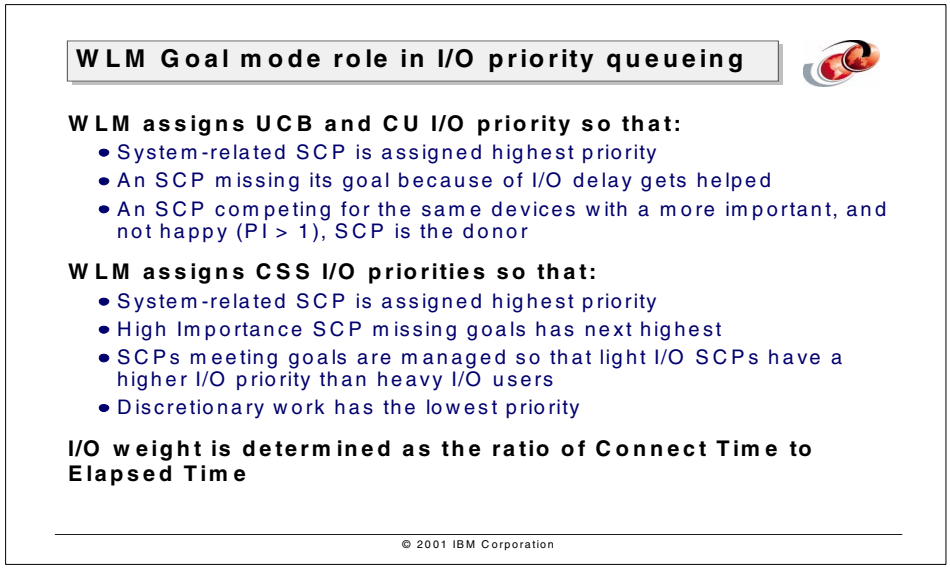
For this reason, you are given the option of specifying a range of priorities that can potentially be used for each LP. This gives you the ability to ensure that the I/O requests from a production LPAR Cluster will always have a higher channel subsystem I/O priority than the requests from a test or development LP running on the same CPC.

This capability is provided as follows:

- Each LP can have a range of CSS I/O Priorities that z/OS WLM is allowed to use. You should set the same range for all LPs in an LPAR Cluster. The available range of priorities is large enough so that LPAR Clusters on the same CPC can have different ranges of priorities. WLM manages the CSS I/O Priority within the range you specified. If the range has more than eight values, WLM uses the top eight.
- For non-z/OS LPs, a single priority will be used for all requests from that LP. Each LP can be assigned a different priority.
- For systems running in Basic mode, a default range of 0 to 15 is used. Whether a single priority or a range or priorities is used depends on the operating system running on the CPC. The only operating system that can use more than one priority is z/OS.

Refer to 14.9, “How to manage Channel Subsystem I/O Priority Queueing” on page 379 for more information.

14.7 WLM's role in I/O Priority Queueing



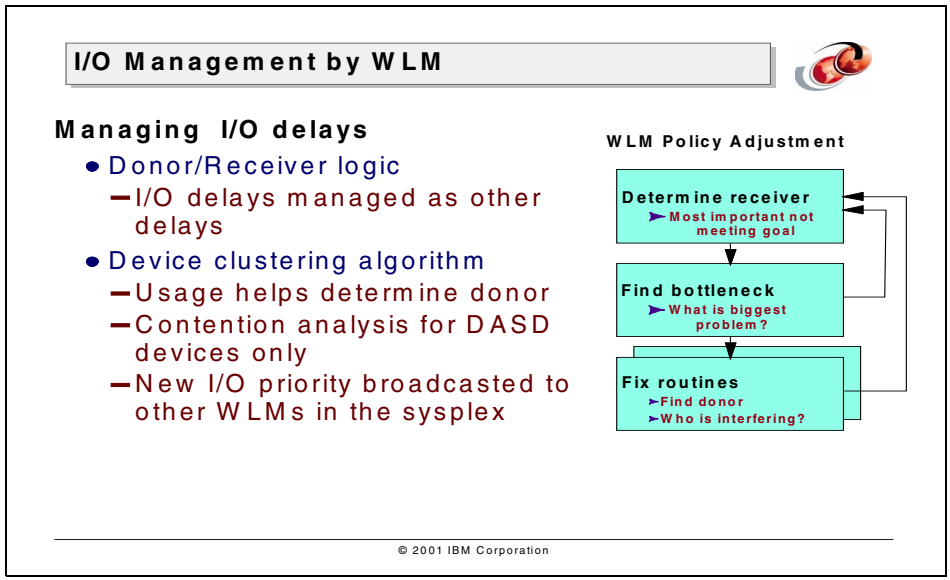
Since OS/390 1.3, WLM can manage the priories of I/O operations. It calculates a value for the I/O priority for each SCP. This value is used when queueing the requests on the UCB, and also when requests are queued within a DASD control unit. This value is based on managing the I/O delays for SCPs on a system running in WLM Goal mode. This is discussed in 14.7.2, "WLM-assigned I/O priority" on page 375.

In addition, z/OS WLM running on an IBM zSeries 900 also calculates a CSS I/O priority to be used by the CSS. This priority is distinct from the I/O priority, and is calculated in a different manner, as described in 14.7.3, "WLM-assigned CSS I/O priorities" on page 376.

Generally speaking, WLM assigns CSS I/O priorities so that:

- A system-related SCP is assigned the highest priority.
- A high importance SCP missing its goals has the next highest priority.
- SCPs meeting their goals are managed so that SCPs that are light I/O users are assigned a higher I/O higher than SCPs that are heavy I/O users.
- Discretionary work has the lowest priority.

14.7.1 WLM management of I/O priority



In order to use any of the WLM I/O priority management functions, you must have set up WLM with the I/O Priority Management option set to Yes. This is set in Option 8 from the WLM primary menu.

Enabling WLM I/O Priority Management has the following impacts:

- UCB and DASD control unit I/O Priority Management is enabled.
- CSS I/O Priority Management is enabled.
- The “I/O Using” and “I/O Delay” values are used in calculating the execution velocity of a SCP.

WLM Goal mode (when I/O Management is active) manages the I/O priority for SCPs in the same way it manages other resources. The policy adjustment algorithm runs periodically. It locates the most important SCP that is not meeting its goal. This is the *receiver* SCP.

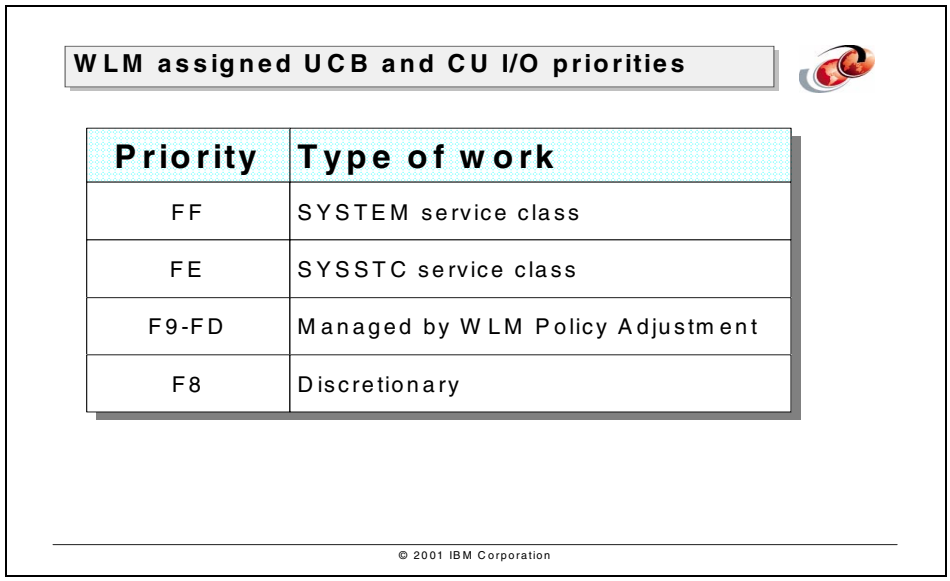
It then determines the bottleneck, that is, what resource is causing the SCP to not meet its goal. The resources managed by WLM Goal mode are CPU, storage, and I/O priorities.

Once the bottleneck is determined, then a *donor* of this resource is located. In the case of I/O delays, a donor candidate is an SCP that is interfering with the receiver for access to DASD devices. This contention is determined by tracking the devices used by each SCP, and the service time and delays for the devices. The SCPs are organized for this analysis into *device clusters*, which are sets of service classes competing for the same or a subset of the same devices. Contention caused by control unit or channel delay is not analyzed here.

WLM then projects the effect of moving resources from the donor to the receiver. If the projected net benefit in terms of a change in the PI of the affected SCPs is positive, then the donor has its I/O priority reduced and the receiver has its I/O priority increased.

Because I/O resources may be shared among different z/OS images in a sysplex, these new I/O priorities are broadcast to the other instances of WLM running in the sysplex so that the I/O priority assigned to the SCP is consistent sysplex-wide. Just for comparison purposes, the dispatching priority and the items associated with storage protection are local resources and consequently are not propagated to other WLMs.

14.7.2 WLM-assigned I/O priority



The WLM policy contains the specification of whether I/O management is enabled or not. See 15.1, “Enabling I/O priority management in WLM” on page 386 for a description of the ISPF panels where you define this information.

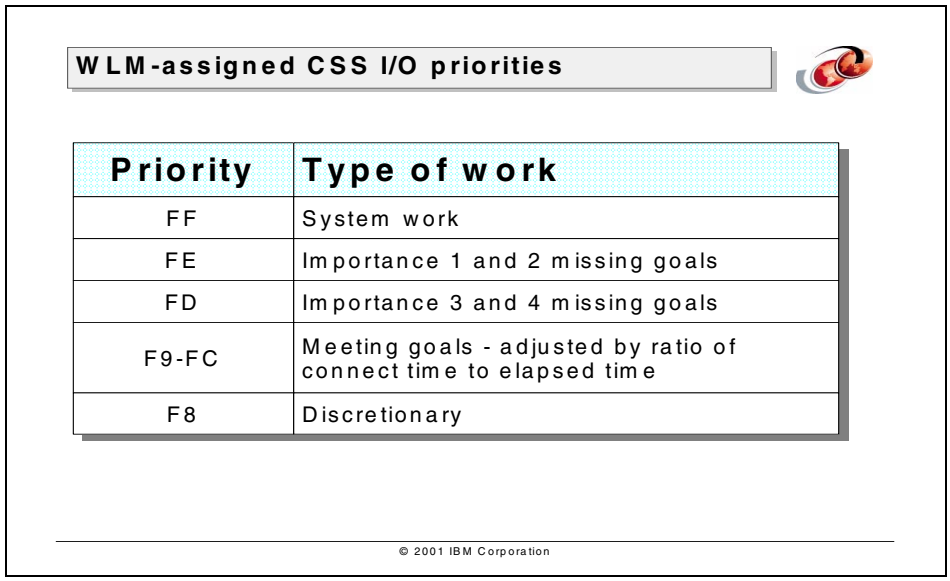
I/O Priority Management in z/OS controls both the I/O priority (UCB and DASD CU) and the CSS I/O priority (channel subsystem) for an SCP. Because the method of arriving at the priority value is different for each of these, we describe them separately. We start with I/O priority, which has been available since OS/390 1.3.

The I/O priority is used to establish the priority of I/O operations queued on the UCB and it is also made available for IOS to send to the control unit if the CU is capable of understanding priority.

Service class periods have their I/O priority modified up or down between X’F9’ and X’FD’ as part of the policy adjustment algorithm that runs every ten seconds. If an SCP is missing its goal because of I/O delays and another SCP is competing for the same DASD devices, then the priority of the receiver SCP has its I/O priority increased and the donor service class has its I/O priority decreased.

It is important to understand that, when we say that the SCP has an I/O priority, what we really mean is that the I/O requests generated by the address spaces and enclaves running in that SCP are the ones with an I/O priority value.

14.7.3 WLM-assigned CSS I/O priorities



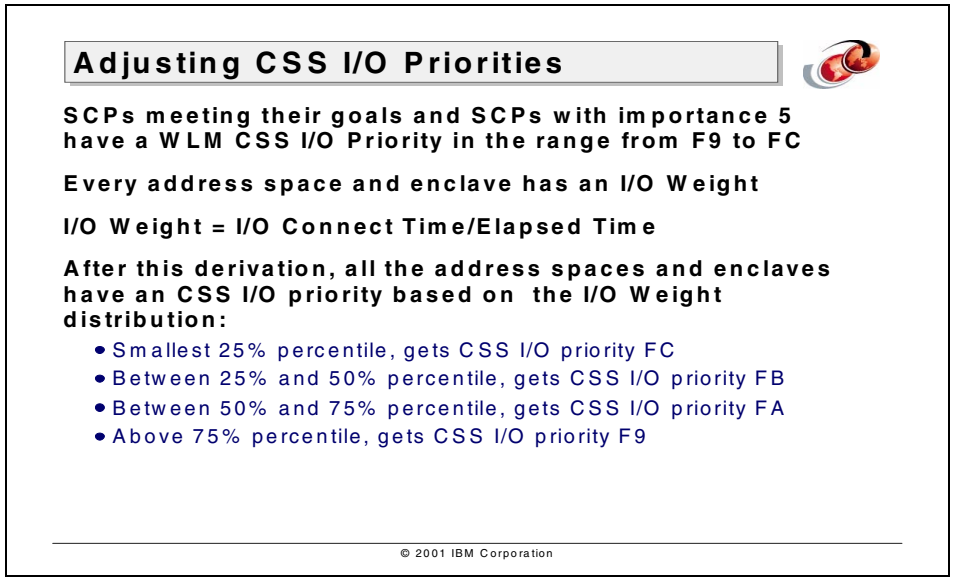
When running on an IBM zSeries 900, z/OS WLM calculates a channel subsystem priority for all I/O requests (assuming WLM I/O Priority Management is enabled). This priority is distinct from the I/O priority discussed in 14.7.2, “WLM-assigned I/O priority” on page 375, which is used for the UCB and control unit queues.

System SCPs (SYSSTC and SYSTEM) are assigned the highest CSS I/O priority (FF). Importance 1 and 2 SCPs missing their goal are next (FE), followed by Importance 3 and 4 SCPs that are missing their goal (FD). Priorities between F9 and FC are used for Importance 5 SCPs and for SCPs that are meeting their goal.

In this case, the CSS I/O priority is adjusted upwards (if the SCP is a light user of I/O) or downwards (if it is a heavy I/O user). This is a similar approach to the old SRM CPU mean-time-to-wait algorithm, where the heavy users stay in the bottom and the light users move to the top. I/O requests for Discretionary SCPs are assigned the lowest value (F8).

The values of CSS I/O Priority described in this figure (from F8 to FF) are mapped by WLM into the range you defined on the HMC for this LP. All the LPs in an LPAR Cluster should have the same range of possible priorities assigned on the HMC. If the range is greater than eight, WLM selects the top eight; if less than eight, it compresses some of its values.

14.8 Adjusting priorities based on Connect time ratio



SCPs meeting their goal (PI < 1) and SCPs with Importance 5 have a WLM CSS I/O Priority in the range from F9 to FC. The I/O intensiveness, or I/O weight, of an SCP determines which priority from within this range will be assigned. The I/O weight of an SCP is calculated as follows:

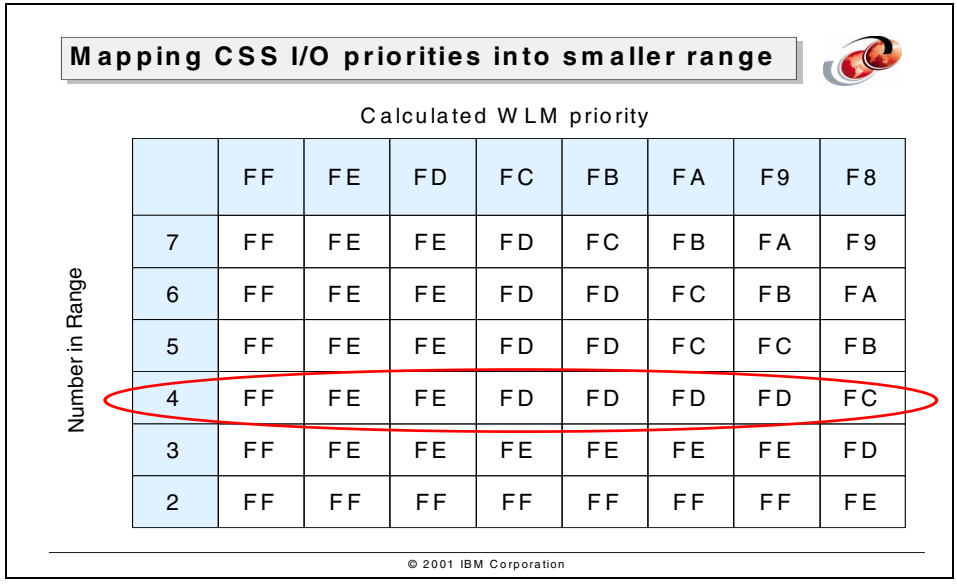
I/O Weight = I/O Connect Time / Elapsed Time

Based on this calculation, all the address spaces and enclaves where the I/O Weight is located:

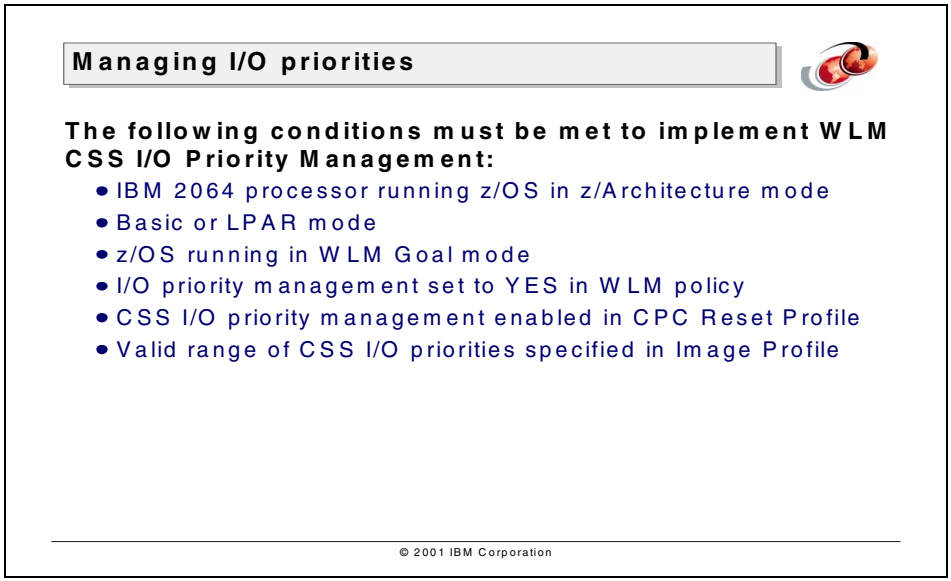
- In the smallest 25th percentile, get CSS I/O priority FC
- Between the 25th and 50th percentile, get CSS I/O priority FB
- Between the 50th and 75th percentile, get CSS I/O priority FA
- Above the 75th percentile, get CSS I/O priority F9

The major intent of this algorithm is to favor throughput by giving more priority to light users.

The CSS I/O priorities are finally mapped to a value within the range specified on the HMC for this LP, as shown in the next figure.



14.9 How to manage Channel Subsystem I/O Priority Queueing



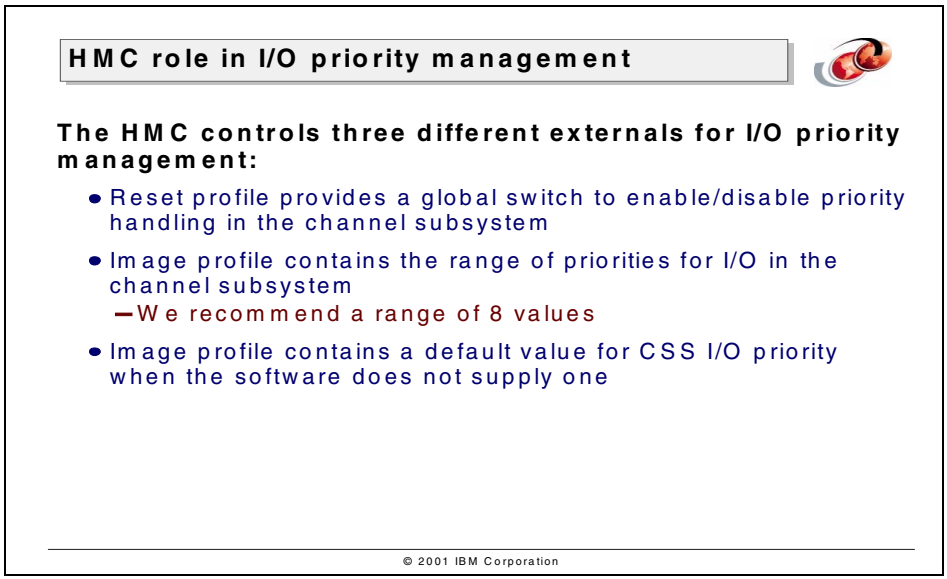
There is really very little involved in managing Channel Subsystem I/O Priority Queueing. Once you enable it on the CPC, and switch it on in WLM, there is nothing else to do. The D WLM,IRD operator command displays whether it is enabled for that LP or not, however RMF does not contain any information specifically relating to it.

The CPC Reset Profile has a switch that enables/disables Channel Subsystem I/O Priority Queueing for the whole IBM zSeries 900 processor. If this switch is set to disable the function, then the channel subsystem operates as before, treating I/O requests in a FIFO fashion. The I/O priority is still used for the queue on the UCB, and in the IBM 2105 Enterprise Storage Server control unit.

The Image Profile for the LP lets you specify the range of priorities that is to be used for this LP. We recommend you specify a range of eight values (for example, 0 to 7), as that is the range that WLM manages. Specifying a smaller range forces WLM to map the calculated priorities into the smaller range allowed for the logical partition, losing some granularity as a result. If the range has more than eight values, WLM will use the top eight priorities in the range. See 15.2, “Enabling CSS I/O priority management on the HMC” on page 387 for a discussion of this parameter.

The Image Profile also allows you to specify the channel subsystem I/O priority to be used if the image is running a release of an operating system that does not support Channel Subsystem I/O Priority Queueing. See 15.2, “Enabling CSS I/O priority management on the HMC” on page 387 for a description of this specification. The customer can prioritize all the CSS requests coming from this image against the other images by specifying a value for the default priority. If you take the default, all I/O requests from the LP will have a priority of 0.

14.10 HMC role



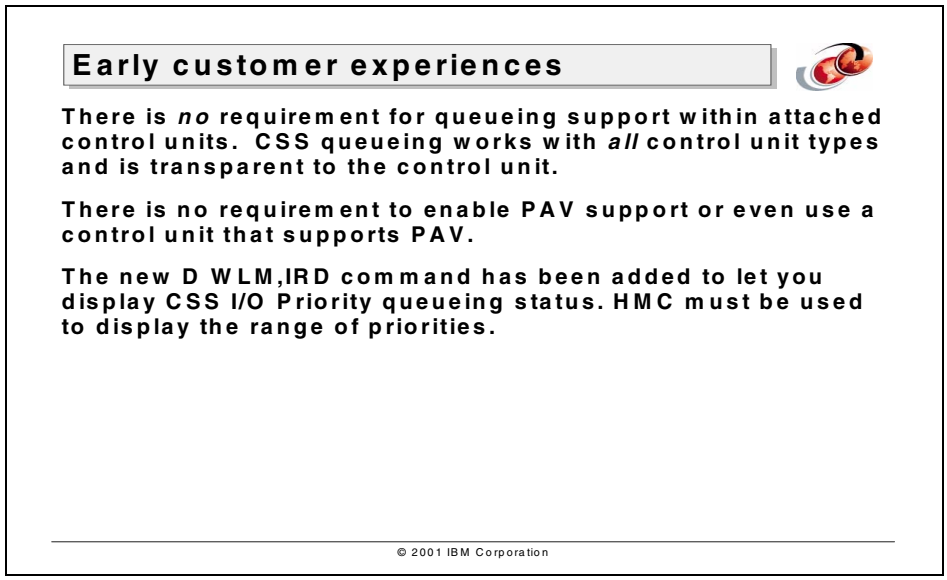
It is important to remember that Channel Subsystem I/O Priority Queueing is enabled or disabled at the CPC level. If the CPC is running in partitioned mode, you cannot use CSS I/O priority queueing for one LP, but not for another. You can specify the initial CSS I/O Priority Queueing status in the Reset Profile for the CPC. You can also switch CSS I/O priority queueing on and off dynamically using the “Enable I/O Priority Queueing” icon in the CPC Operational Customization work area on the HMC. This change is non-disruptive.

As well as turning CSS I/O priority queueing on and off, you also use the HMC to specify the range of priorities that can be used for each LP. This information is kept in the Image profile for each LP, and this controls the initial range of priorities when the LP is activated. In addition, you can change the range dynamically using the “Change I/O Priority Queueing” icon in the CPC Operational Customization work area on the HMC.

Non-z/OS LPs can only use a single I/O priority. This priority is taken from the Minimum I/O Priority field in the Image profile. There is not a separate field for non-z/OS LPs. This gives you the flexibility to IPL a z/OS operating system at one time, and have it use the full range of priorities, and then IPL a different operating system, which will use the minimum priority, without having to make any changes at the HMC.

The HMC panels used to control this are shown in 15.2, “Enabling CSS I/O priority management on the HMC” on page 387.

14.11 Early implementation experiences



The following points are based on customer experiences during the Early Support Program for Intelligent Resource Director:

- There is no interdependency between Channel Subsystem I/O Priority Queueing and I/O Priorities for UCB and/or control unit queueing. All are enabled or disable via WLM, but the control unit does not require support for I/O Priority queueing.
- There is no requirement to have Parallel Access Volumes enabled to use Channel Subsystem I/O Priority Queueing nor is it necessary for PAV support on the control units.
- The D WLM,IRD command has been added to provide a way to find out in z/OS whether Channel Subsystem I/O Priority Queueing is enabled. The only way to find the range of priorities is to check the HMC.

15

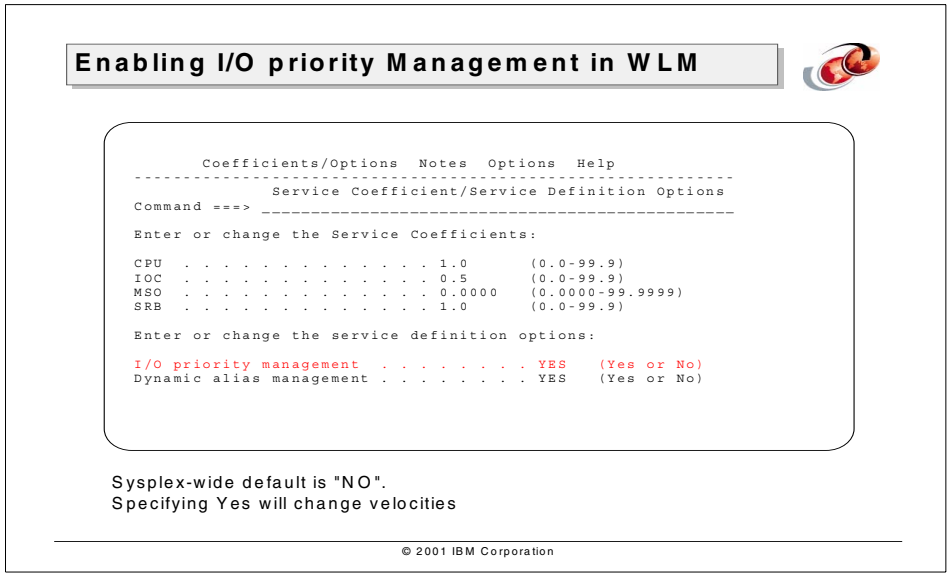
Planning & implementing CSS I/O Priority Management

The channel subsystem is a shared resource across *all* the LPs on a CPC. For all the z/OS partitions that are part of the same LPAR Cluster, and running in WLM Goal mode, WLM can consistently set channel subsystem priorities across those partitions. However, you must remember that the channel subsystem requests from any other LP may be impacted by the priorities that are assigned to the z/OS I/O requests. Implementing Channel Subsystem I/O Priority Queueing in an LPAR Cluster is relatively simple. What is a little more complex is implementing it in a manner that delivers the results you expect from *all* the LPs on the CPC.

Remember that the channel subsystem (and its queues) are shared by all the LPs in the CPC. If you don't assign priorities to any LP, all channel subsystem requests will have the default priority of 0, and everything will behave as it did prior to Channel Subsystem I/O Priority Queueing. However, as soon as you change the range of priorities for one LP, you indirectly impact the requests from every other LP.

In this chapter, we talk about how to enable Channel Subsystem I/O Priority Queueing, and the things to consider so that you receive the results you expect, in *all* LPs.

15.1 Enabling I/O priority management in WLM



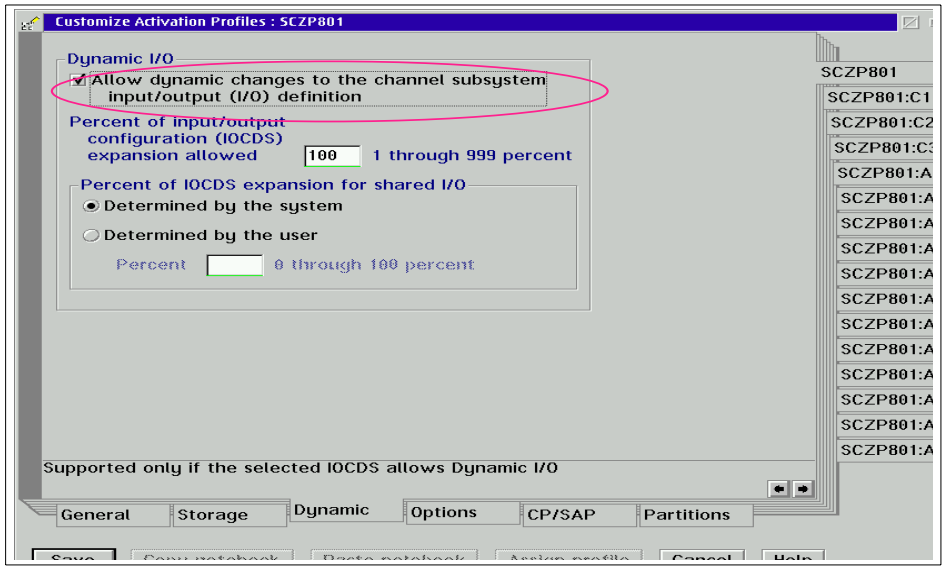
As discussed in 14.7.1, “WLM management of I/O priority” on page 373, you must indicate in the WLM policy whether you wish WLM to manage I/O priorities in the sysplex. This is specified in Option 8 from the WLM primary panel, as shown in the figure above.

The default for I/O priority management is NO, which results in I/O priorities being equal to dispatching priorities, and CSS priorities being set to the lowest value assigned for that LP in the Image Profile. For the I/O priorities, this is identical to how they were handled prior to OS/390 1.3. If you specify YES, WLM controls the I/O priorities in the sysplex, setting values that are based on the goals and goal achievements of the various workloads.

Additionally, when I/O priority management is set to YES, the I/O component of a workload is used in determining the velocity of the associated SCP. The new DASD I/O Using and DASD I/O Delay samples are reported by RMF even when WLM I/O priority management is turned off. This allows you to calculate the new velocity values in advance of enabling this function.

If WLM is not in Goal mode, the system will not assign a CSS I/O priority to its I/O requests, and all requests will be assigned the minimum I/O priority as defined on the HMC, just as if the operating system was at a level that did not support Channel Subsystem I/O Priority Queueing.

15.2 Enabling CSS I/O priority management on the HMC



The figure above shows the CPC Reset Profile panel for an IBM zSeries 900 CPC. The new field that controls whether Channel Subsystem I/O Priority Queueing is initially enabled after a reset of the CPC is highlighted. Note that this field is on the “Options” tab of the profile.

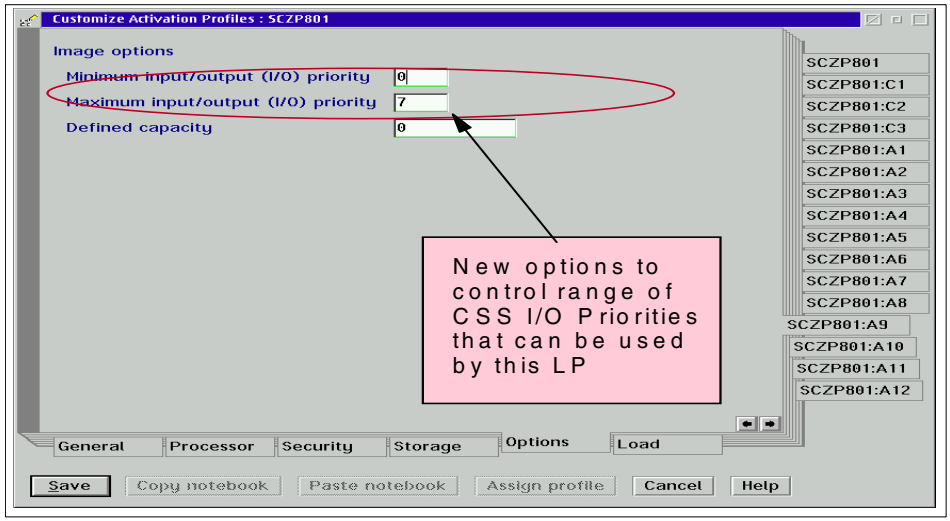
To dynamically enable or disable CSS I/O Priority Queueing, use the Change I/O Priority Queueing icon on the CPC Operational Customization panel on the HMC.

The Image Profile for an LP (shown in the figure on the next page) defines the range of priorities that can be used by an operating system in this LP. These values apply when the LP is activated. They can subsequently be changed dynamically if you wish.

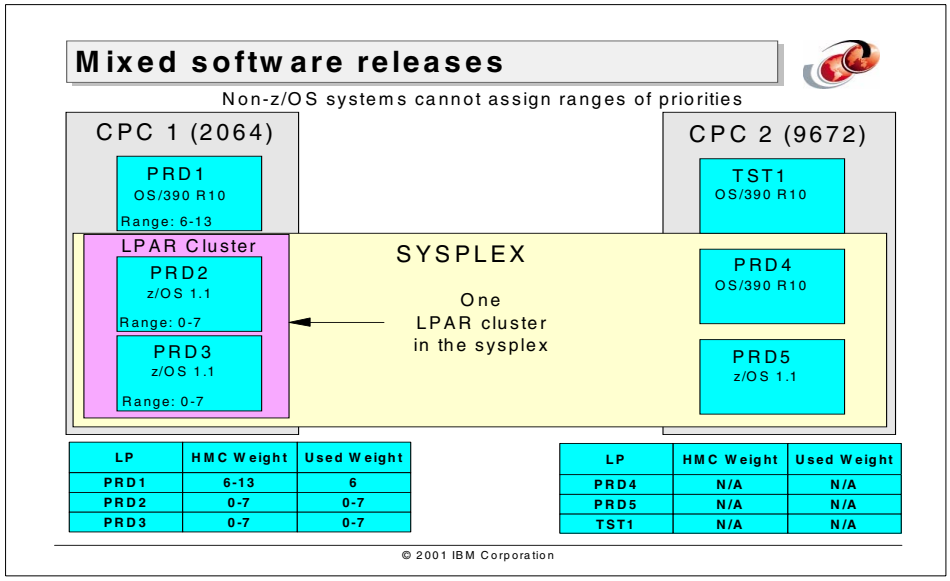
If z/OS is IPLed in the LP and WLM is in Goal mode, WLM will be presented with the range of priorities, and it will map its internally-calculated priorities onto the range you specify on the HMC.

If the LP is not running z/OS, all I/O requests originating in that LP will be assigned a priority equal to the minimum I/O priority you specify on the HMC.

We recommend that you specify a range of eight priorities, as WLM's internal algorithms assign one of eight CSS I/O priorities to each SCP. See 14.7.3, "WLM-assigned CSS I/O priorities" on page 376 for a description of the values. Note the range is from X'F8' to X'FF'. If you specified a range of less than eight priorities for this image, then WLM remaps the value it has calculated into the specified range. See 14.8, "Adjusting priorities based on Connect time ratio" on page 377 for a description of this process.



15.3 Planning for mixed software levels



In the figure above, we show two CPCs: one is a 9672 G6 processor, and the other is an IBM zSeries 900. The customer is running a production sysplex consisting of four MVS images, two on CPC 1 (the IBM zSeries 900), and two on CPC 2 (the 9672 G6). This sysplex obviously shares DASD devices and is controlled by a single WLM policy. There is also one other LP on each CPC that is not a part of the sysplex. Let's look at the CSS priority of I/O requests issued by each of the operating systems in the various LPs.

LPs PRD2 and PRD3 are both running z/OS in z/Architecture mode, are in the same sysplex, and are on an IBM zSeries 900. Therefore, they are in the same LPAR Cluster. Each of these LPs has an LPAR Image Profile which contains a definition of the range of CSS priorities that can be used by these LPs. Because these systems are in WLM Goal mode, the channel subsystem I/O priority will be managed; that is, a priority within the range specified on the HMC will be assigned to each I/O request by WLM.

LP PRD1 is also running on the IBM zSeries 900. It also has an Image Profile that contains a range of CSS I/O priorities that may be assigned to I/O requests from that LP. However, because PRD1 is running OS/390, it cannot manage its CSS priorities. As a result, the hardware will assign a CSS priority of 6 to all I/O requests issued by that system (the range of priorities defined for this LP is 6-13, so this LP will use the minimum of that range: 6).

Because CPC1 has LPs containing operating systems that do not support Channel Subsystem I/O Priority Queueing, you must give some thought to the range of priorities defined for each LP:

- All LPs that are in the same LPAR Cluster should have the same range of priorities defined to them.
- In the example above, both of the LPs that exploit Channel Subsystem I/O Priority Queueing will issue I/O requests with a CSS priority of 0 to 7 (as shown in the figure on the previous page).
- All LPs that do *not* support Channel Subsystem I/O Priority Queueing will have a CSS priority assigned to them that is equal to the minimum specified on the HMC.
- In our example, the PRD2 and PRD3 LPs will issue requests with a CSS priority of somewhere between 0 and 7. You have to set the range of priorities for all the non-supporting LPs based upon this range.

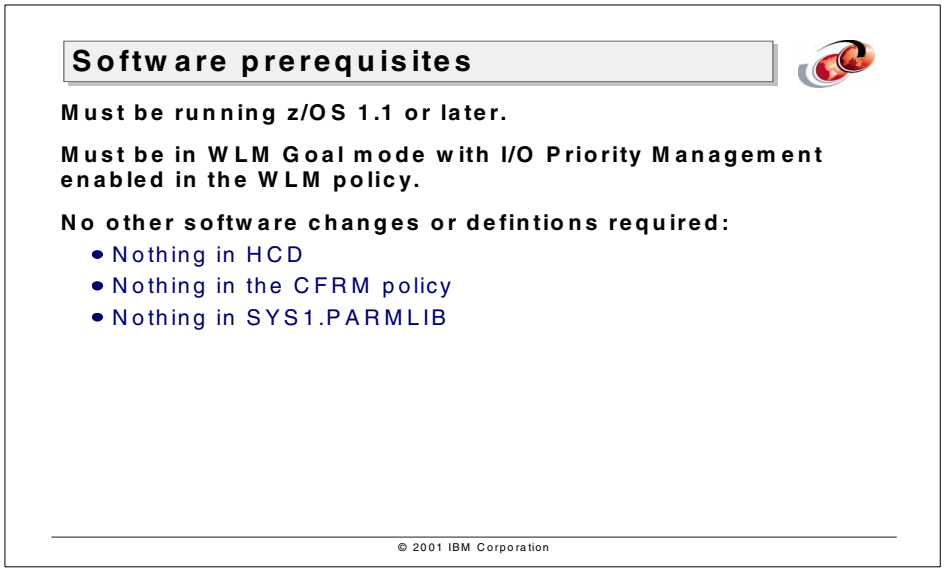
  If PRD1 contained the most important production LP, you might specify a range of 8 to 15 on the HMC. This would ensure that *all* I/O requests from that LP would have a higher CSS priority than any of the requests from the PRD2 and PRD3 LPs (because all requests from PRD1 would be assigned a CSS priority of 8).

  If PRD1 contained a test system, you might define a range from 0 to 7 for that LP. In this case, *all* the I/O requests from PRD1 would have the same priority (0) as the I/O requests from Discretionary SCPs in PRD2 and PRD3.

  We have assigned a range of 6 to 13 for PRD1. This is because PRD1 actually contains an important production workload. In this case, all the I/O requests from PRD1 would have a CSS priority of 6, which is equal to the Importance 1 and 2 workloads that are missing their goals in PRD2 and PRD3, but less than the I/O requests from any of the system SCPs in PRD2 and PRD3.

LPs PRD4 and PRD5, and TST1 are running on CPC 2 (a 9672-G6), which does not support Channel Subsystem I/O Priority Queueing—even though one of the LPs is running z/OS. All I/O requests for these LPs are handled by the channel subsystem in FIFO order.

15.4 Software prerequisites



In order to use Channel Subsystem I/O Priority Queueing, you must be running z/OS 1.1 or later, in z/Architecture mode, and on an IBM zSeries 900 or later CPC.

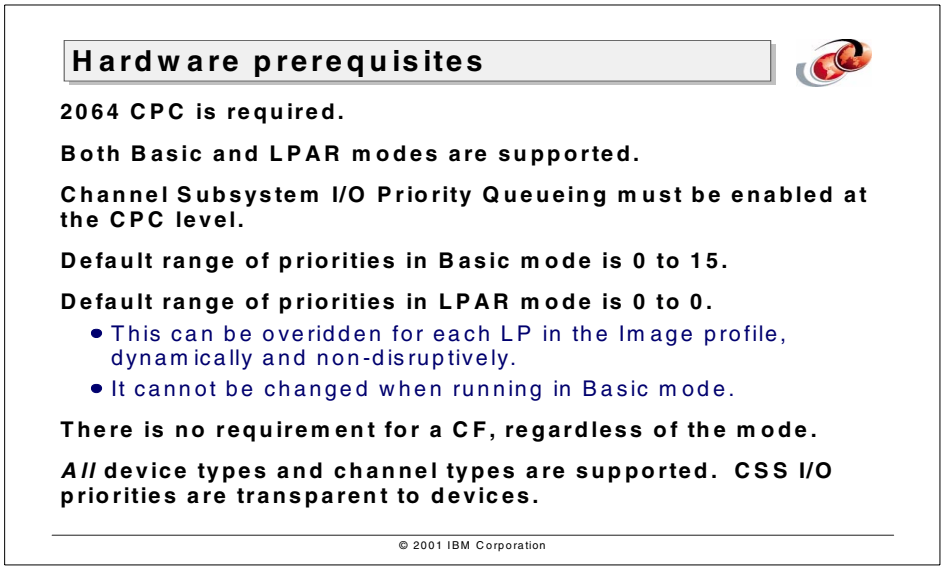
In addition, WLM must be in Goal mode, with I/O Priority Management enabled.

There are no other prerequisites. You do not have to make any changes to HCD. Channel Subsystem I/O Priority Queueing does not use a CF structure, so there are no CFRM policy changes required. The system does not have to be part of a sysplex. And there are no parameters related to Channel Subsystem I/O Priority Queueing in SYS1.PARMLIB.

Channel Subsystem I/O Priority Queueing is transparent to all other products.

For the latest information regarding software service that may apply to any of the Intelligent Resource Director functions, refer to the IRD/PSP bucket. This is available in the IRD subset of the 2064DEVICE upgrade.

15.5 Hardware prerequisites



In order to use Channel Subsystem I/O Priority Queueing, you must be running on an IBM zSeries 900 or later CPC.

Channel Subsystem I/O Priority Queueing must be enabled at the CPC level.

Each LP has a default range of priorities of 0 to 0 assigned to it. This is controlled in the Image profile, and can be altered dynamically and non-disruptively for each LP.

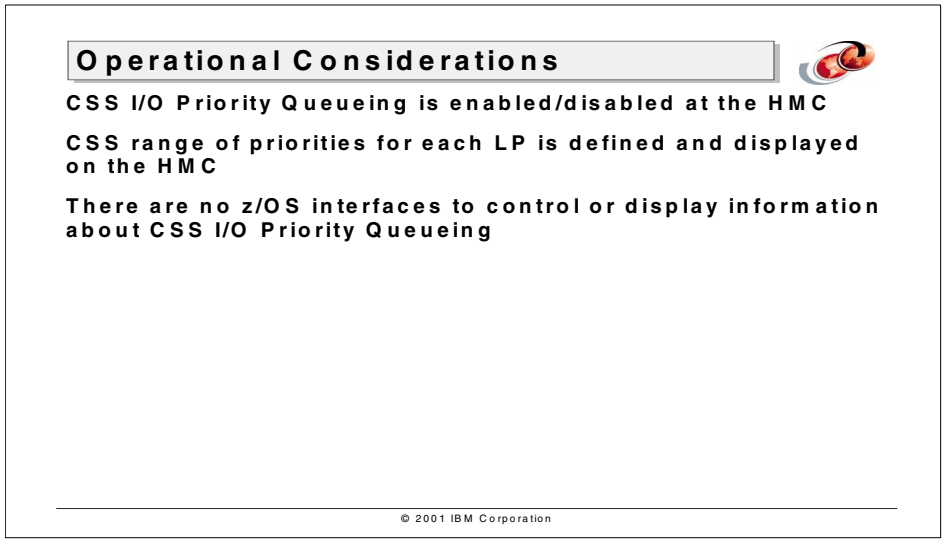
When running in Basic mode, the range of priorities is 0 to 15, and this cannot be overridden by the installation.

It is *not* necessary to have a CF, regardless of the CPC mode (Basic or LPAR), nor is it necessary to be part on an LPAR Cluster.

All channel types are supported: Parallel, ESCON, FICON Bridge, and FICON Native. Also, all device types are supported: it is not just for DASD.

Finally, we want to stress again that Channel Subsystem I/O Priority Queueing affects *every* I/O request. It is transparent to the device, and therefore has no dependencies on any devices outboard of the CPC.

15.6 Operational considerations

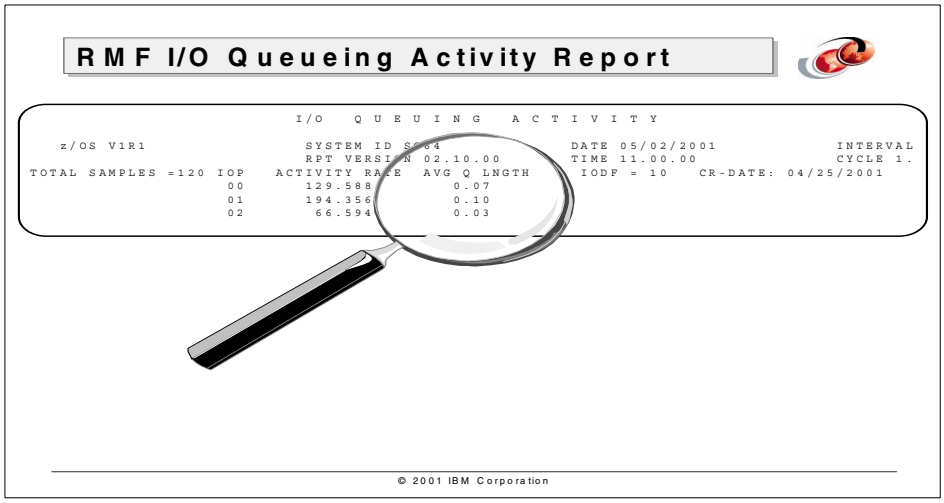


Apart from defining the range of priorities on the HMC, and being able to dynamically turn Channel Subsystem I/O Priority Queueing on and off for the whole CPC (also at the HMC), there are no other operator interfaces related to Channel Subsystem I/O Priority Queueing.

At the time of writing, there is no way, other than using the HMC, to find out if Channel Subsystem I/O Priority Queueing is enabled. In the future, the `D WLM` command may be enhanced to add Channel Subsystem I/O Priority Queueing status information.

Should any diagnostics be required, WLM writes some information about Channel Subsystem I/O Priority Queueing in its Type 99 records.

15.7 Performance and tuning



RMF does not provide a single report that you can use to identify whether you need Channel Subsystem I/O Priority Queueing, nor the benefit provided when you enable it. However, using information that is spread across a number of RMF reports, it is possible to determine if you are experiencing channel subsystem queueing and to calculate the amount of time that I/O requests are queued for each control unit.

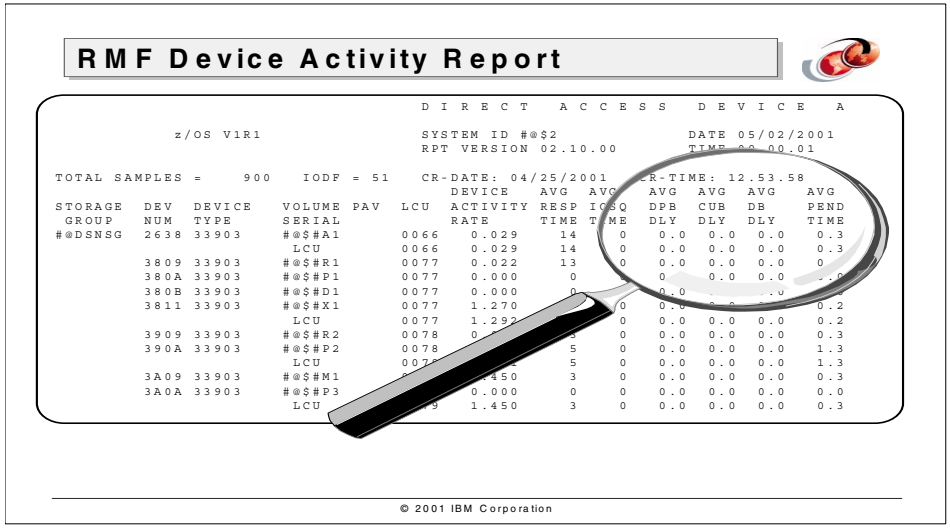
The RMF I/O Queueing Activity Report (shown in the figure above) shows the activity rate and the average queue length for each SAP (described as IOPs in the RMF report). If the average queue length is greater than 0, you have the potential to benefit from Channel Subsystem I/O Priority Queueing.

The next step is to identify the amount of time that is being spent queueing in the channel subsystem. Remember that Channel Subsystem I/O Priority Queueing will *not* reduce the *overall* average queue time--it *will* reduce the average queue time for the *higher-importance workloads*.

Unfortunately, RMF does not directly report channel busy queue time, which is one of the components of Pend time. Pend time actually consists of time spent queueing because:

- All the channels to the required device are busy.
- The request failed because the director port is busy.
- The control unit is busy.
- The device is busy because it is being used by another system.

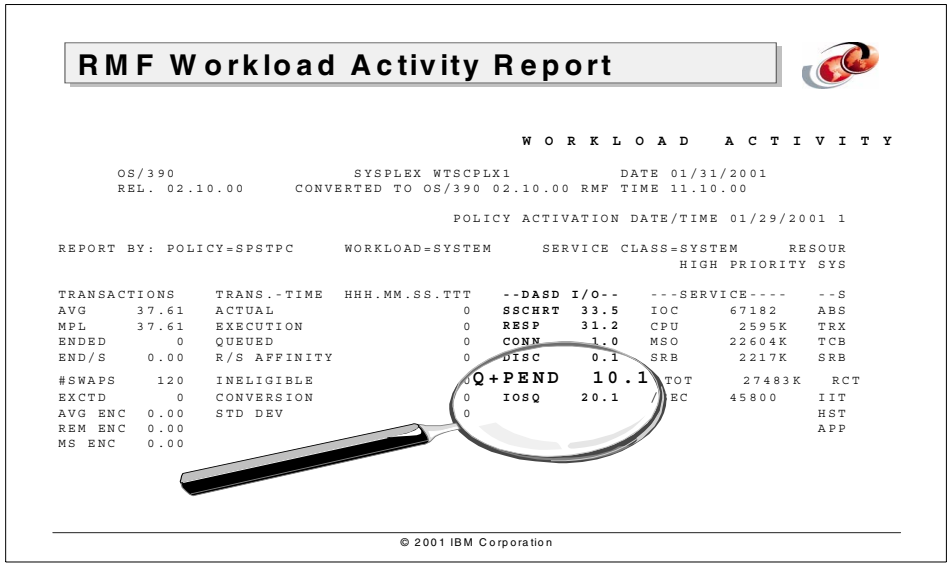
While RMF does not report channel busy times, the RMF Device Activity Report (shown in the figure below) does report all the other components of Pend time, both at the device and LCU level. To determine the channel busy time, subtract director port busy (reported as DP Busy), control unit busy, and device busy from Pend time--the result is the channel busy time. If director port busy, control unit busy, and device busy are all 0 (or a very small number), you can use the Pend time to approximate channel busy time. If a control unit has high channel busy times and is used by high Importance workloads, it is likely that those high Importance workloads would benefit from Channel Subsystem I/O Priority Queueing. Obviously, if you enable Channel Subsystem I/O Priority Queueing it will impact all control units; however, the impact is likely to be less visible on control units that do not have high channel busy time.



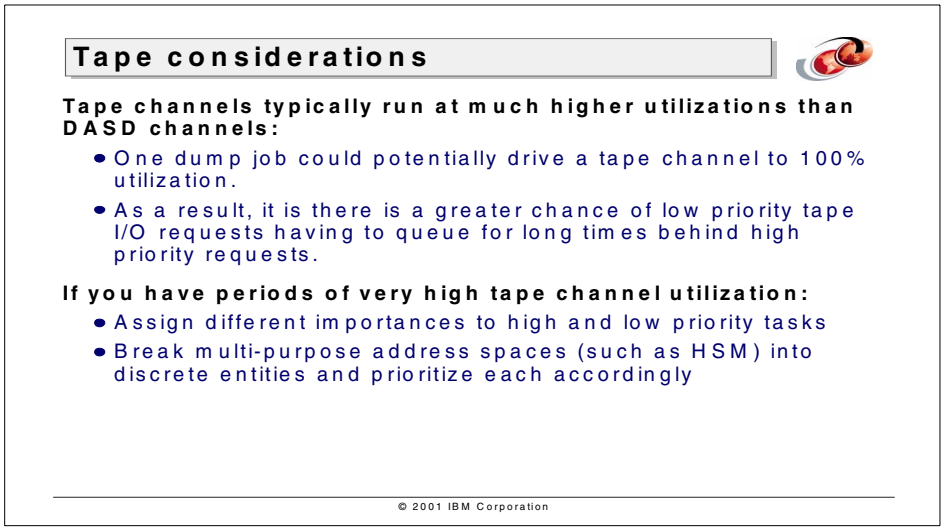
So, if you enable Channel Subsystem I/O Priority Queueing, how do you know what difference it makes? The channels still have to handle the same number of requests, so total Pend time will not change. What *will* change is the channel busy seen by individual I/O requests, based on the WLM Importance of the issuing SCP. Therefore, you need a way of looking at the Pend time for those requests.

If you have devices that are used nearly exclusively by high Importance workloads, and other devices sharing the same set of channels that are used predominately by low Importance workloads, and you are currently suffering from channel subsystem contention, you should see a reduction in Pend time for the devices used by the high Importance workloads, and a corresponding increase in Pend time for the devices used by the low Importance workloads.

Another option is to review the Pend time by Service Class Period, as reported in the RMF Workload Activity Report. An example report is shown in the figure below. Here, you should see that the field entitled Q+PEND decreases for the high Importance workloads, especially those missing their goals, and increases for the lower Importance workloads. The level of improvement obviously depends on how constrained the channel subsystem is. If there is no queueing (the channel subsystem is not busy), then you would not expect to see a reduced Q+PEND time.



15.8 Tape devices



In most installations, the devices that tend to be associated with the highest channel utilizations are tape drives. While DASD I/O requests tend to be short, and response time-driven, tape I/O requests are usually large (32 KB or 64 KB block sizes) and throughput-driven. While no one would ever want to see a DASD channel running at 100% utilization, this level of utilization would be the norm for tape channels during periods of high activity.

Where channels are driven to such high utilizations, prioritizing the I/O requests potentially has a much more significant impact--especially on the lower priority requests. Channel Subsystem I/O Priority Queueing contains logic specifically designed to ensure that low-priority requests don't get locked out indefinitely, however this will not equalize the response times of high and low priority requests.

If you have tape channels that are shared by high and low Importance workloads (in the same or different LPs) and which are both busy at the same time, you may see significant elongation in the response times (and therefore elapsed time) for the lower Importance workloads. On the other hand, the high Importance workloads should see a significant improvement in response and elapsed times. While this is not necessarily a concern, it is something that should be considered if your environment matches this profile.

Specifically, if you have data base log archive jobs that write to tape, you should review the WLM Importance of those jobs in relation to other tape jobs. If you use DFSMShsm, you may consider using the Multiple Address Space HSM (MASH) capability that was introduced in OS/390 V2R10. This gives you the ability to have multiple HSM address spaces, each doing different functions (for example, Backup, Dump, Recall), and each potentially having a different WLM Service Class. In this case, you might want to assign a higher-importance Service Class to the address space that is responsible for data set Recalls.

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

IBM Redbooks

For information on ordering these publications, see “How to get IBM Redbooks” on page 400.

- *IBM Enterprise Storage Server Performance Monitoring and Tuning Guide,* SG24-5656
- *MVS/ESA HCD and Dynamic Reconfiguration Primer*, SG24-4707
- *OS/390 Workload Manager Implementation and Exploitation*, SG24-5326

Other resources

These publications are also relevant as further information sources:

- *DSF User’s Guide and Reference*, GC35-0033
- *Enterprise Systems Architecture/390: Principles of Operation*, SA22-7201
- *Enterprise Systems Architecture/390: Common I/O-Device Commands*, SA22-7204
- *IOCP User's Guide and ESCON Channel-to-Channel Reference,* GC38-0401
- *Planning for Workload Charging*, SA22-7506
- *PR/SM Planning Guide*, GA22-7236
- *Systems Journal Vol30, No 1, 1991 - Article entitled “ESA/390 interpretive-execution architecture, foundation for VM/ESA”*, G321-0102
- *Systems Journal Vol 28, No 1, 1989 - Article entitled “Mutliple operating systems on one processor complex*”, G321-0093
- *z/OS Hardware Configuration Definition Scenarios,* SC33-7987
- *z/OS MVS Data Areas, Volume 5 (SSAG - XTLST)*, GA22-7585
- *z/OS MVS Planning: Workload Management*, SA22-7602
- *z/OS RMF Report Analysis*, SC33-7991
- *z/OS MVS Setting Up a Sysplex*, SA22-7625

Referenced Web sites

These Web sites are also relevant as further information sources:

- Workload Charging:

  `http://www.ibm.com/servers/eserver/zseries/wlc_lm/`

- Workload Manager:

  `http://www.ibm.com/servers/eserver/zseries/software/wlm/`

- IRD:

  `http://www.ibm.com/servers/eserver/zseries/zos/wlm/documents/ird/ird.html`

How to get IBM Redbooks

Search for additional Redbooks or redpieces, view, download, or order hardcopy from the Redbooks Web site:

`ibm.com/redbooks`

Also download additional materials (code samples or diskette/CD-ROM images) from this Redbooks site.

Redpieces are Redbooks in progress; not all Redbooks become redpieces and sometimes just a few chapters will be published this way. The intent is to get the information out much quicker than the formal publishing process allows.

IBM Redbooks collections

Redbooks are also available on CD-ROMs. Click the CD-ROMs button on the Redbooks Web site for information about all the CD-ROMs offered, as well as updates and formats.

Special notices

References in this publication to IBM products, programs or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM's product, program, or service may be used. Any functionally equivalent program that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program or service.

Information in this book was developed in conjunction with use of the equipment specified, and is limited in application to those specific hardware and software products and levels.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM Corporation, Dept. 600A, Mail Drop 1329, Somers, NY 10589 USA.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The information contained in this document has not been submitted to any formal IBM test and is distributed AS IS. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer's ability to evaluate and integrate them into the customer's operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

Any pointers in this publication to external Web sites are provided for convenience only and do not in any manner serve as an endorsement of these Web sites.

The following terms are trademarks of other companies:

Tivoli, Manage. Anything. Anywhere.,The Power To Manage., Anything. Anywhere.,TME, NetView, Cross-Site, Tivoli Ready, Tivoli Certified, Planet Tivoli, and Tivoli Enterprise are trademarks or registered trademarks of Tivoli Systems Inc., an IBM company, in the United States, other countries, or both. In Denmark, Tivoli is a trademark licensed from Kjøbenhavns Sommer - Tivoli A/S.

C-bus is a trademark of Corollary, Inc. in the United States and/or other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and/or other countries.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States and/or other countries.

PC Direct is a trademark of Ziff Communications Company in the United States and/or other countries and is used by IBM Corporation under license.

ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States and/or other countries.

UNIX is a registered trademark in the United States and other countries licensed exclusively through The Open Group.

SET, SET Secure Electronic Transaction, and the SET Logo are trademarks owned by SET Secure Electronic Transaction LLC.

Other company, product, and service names may be trademarks or service marks of others.

Index

D

E

F

G

H

I

L

M

N

O

P

Q

R

S

T

U

V

W

X

Z

z/OS Intelligent Resource Director

(0.5" spine)
0.475"<->0.875"
250 <-> 459 pages

z/OS Intelligent Resource Director

WLM LPAR CPU Management

Dynamic Channel-path Management

Channel Subsystem I/O Priority Queueing

This IBM Redbook describes the new LPAR Clustering technology, available on the IBM @server zSeries processors, and z/OS. The book is broken into three parts:

- Dynamic CHIPD Management
- I/O Priority Queueing
- CPU Management

Each part has an introduction to the new function, planning information to help you assess and implement the function, and management information to help you monitor, control, and tune the function in your environment.

The book is intended for System Programmers, Capacity Planners, and Configuration Specialists and provides all the information you require to ensure a speedy and successful implementation of the functions at your installation.

INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

For more information:
ibm.com/redbooks

SG24-5952-00 ISBN 0738417904