Mainframe zOS

Back to Basics

Natabar Sahoo

natabarss@yahoo.com

V01
October 2024

(This book is dedicated to all zOS Systems Programmers, especially the young ones.)

Disclaimer

All the details described in this book are the notes captured by the author over many years of working as a zOS Systems Programmer. IBM manuals, Redbooks, IBM Training courses and many other presentations and articles in the internet, have been the source of information which has been compiled in this book. You may use this book as a reference to learn and refresh your basic knowledge of zOS. The information presented here could be obsolete or changed and you must use the latest information available against your current mainframe configuration.

The copyright of the information, provided here, remains with the original creator. I have included them here for reference only.

Comments and Feedback

I want this book to be useful to the zOS Systems Programmers (Sysprog) community starting from Level ZERO or other experienced technical Subject Matter Experts (SMEs). So. I very much welcome your comments and feedback on the contents of this book.

I have started this book with very limited information and will keep expanding with additional details. At the same time, I will be happy and look forward to your contribution which you feel will help the zOS Sysprog community and I will include them in this book. A short write up to expand the existing topics or to be included as a new topic will be of help. I wish to include all relevant zOS basics only without going through much technical details.

Please do write to me at *natabarss@yahoo.com* and I look forward to hearing from you.

Acknowledgements

I would sincerely like to thank IBM and the entire zOS community, directly or indirectly involved, in preparing and making millions of pages of information available to the zOS Systems Programmers.

Contents

Introduction

As a zOS Sysprog, you get lost in the ocean of information available to you from different sources. The zOS manuals have millions of pages of information and we hardly use less than five percent of it during our whole professional career.

So, my simple advice to any zOS Sysprog is:

- Understand the basics clearly and then forget the details.
- Do not try to remember a lot except the basics.
- Know where to find the relevant information quickly in specific context.
- Know the zOS manuals classifications such as Installation, Customization, Guide, Reference, Messages, Diagnosis, Data Areas etc.
- Keep track of all new developments to refresh your basics up-to-date.
- Explore through old and new manuals, Redbooks, presentations, announcements, and other relevant information.
- There is no substitute to self-study and hands on in the system.

During the 40 years of my professional career, not a single day has gone when I have not taken the help of my basic knowledge. A single diagram or mind map on different areas, remained always in my mind from where I always navigate to more details. This experience has been the inspiration for compiling this book. The topics are not in order and you may refer to any topic of your interest. Many of the diagrams are self-explanatory and so I will not describe a lot everywhere. You can always refer to various manuals for more details.

The information provided here are the notes captured by me over a long period of time and could be old and some may be obsolete. So please refer to the latest information published by IBM and others.

I had few postings in Linkedin which I have also included here.

What is mainframe zOS

Very often, my friends ask me how to understand/explain mainframe zOS in simple terms.

I tell them, if you understand what is a ‘Dataset’ and an ‘Address Space,’ then the answer is very simple, means zOS is a lots of **datasets** and **address spaces**.

1. At lower level, it is thousands of **datasets** e.g. SYS1.PARMLIB, SYS1.PROCLIB, SYS1.MACLIB, A.B.C, X.Y.Z etc. consisting of load modules, macros, scripts, parameters etc. In summary, the datasets contain all required elements to build zOS system and support execution of all workloads under it.

2. When you look at an active zOS system, it is hundreds of the **address spaces** created at the time of IPL or later.

But, how to create, update and use these data sets? And, which address space does what? In reality, there are hundreds and hundreds of jigsaw puzzles, some very discrete and many interconnected, to understand and solve. If you know the techniques to build the jigsaw puzzles, then understanding zOS is extremely simple. At the same time, please remember, a simple error anywhere may cause a disaster. But don’t worry, you can always replace/rebuild/customize any single piece of the puzzles to bring back everything in order. As a support personnel, you must know what you do and always focus on the context instead of always thinking big.

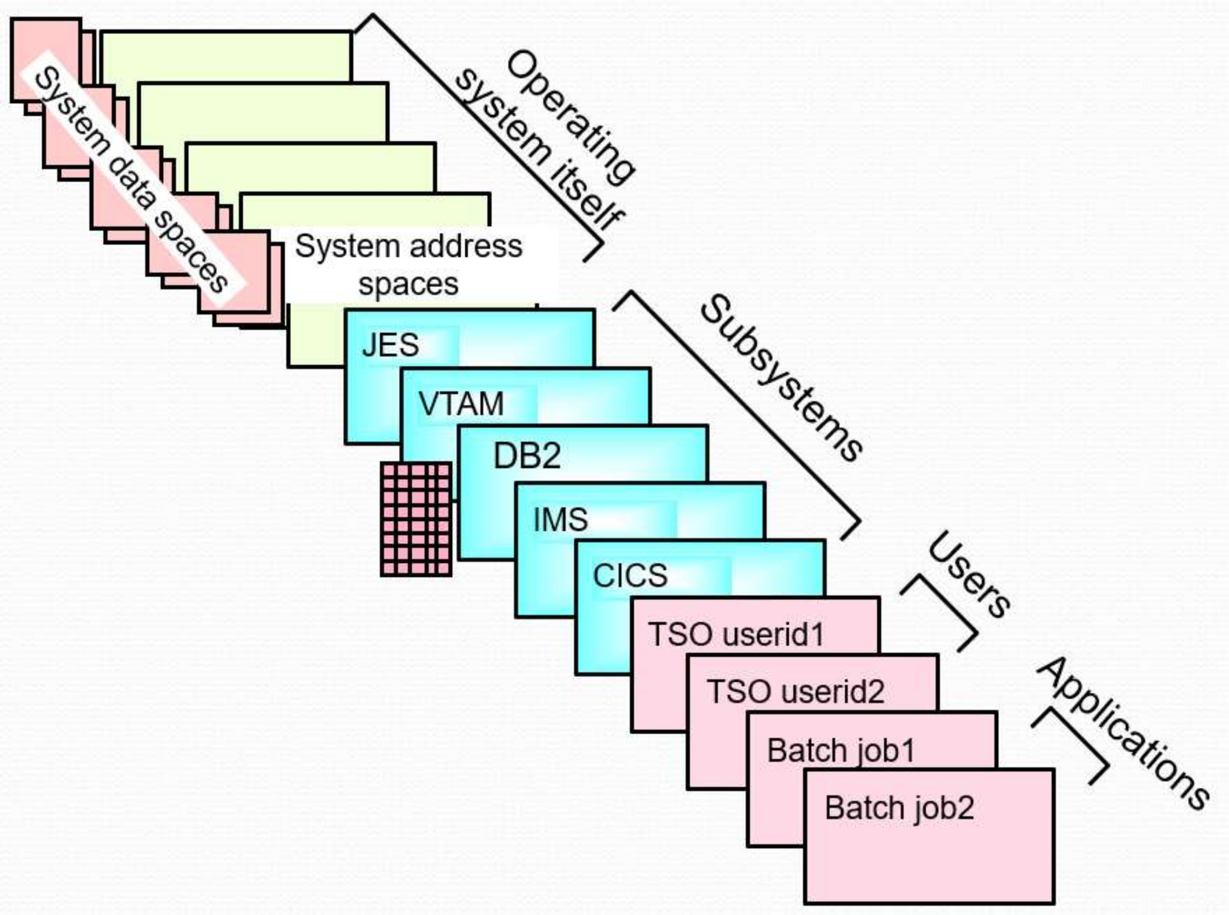
Today, zOS is a 64-bit Operating System, but still supports the previous 24-bit and 31-bit architecture. It performs multiprogramming and multiprocessing and supports;

- 1000s of users concurrently
- I/O and numeric intensive computing
- Processing very large heterogeneous workloads
- Running mission critical applications securely

Historically, zOS has undergone many changes since 1964 i.e. “**OS/360 -> OS/VS -> MVS -> MVS/SP -> MVS/XA -> MVS/ESA -> OS/390 -> z/OS**” but the core still remains **MVS** (Multiple Virtual Storage).

Address spaces under zOS

In general, the zOS address spaces are classified as follows.



- Systems address spaces
  - System address spaces are started after initialization of the master scheduler. These address spaces perform functions for all the other types of address spaces that start in zOS.
- Sub-system address space
  - Subsystem address spaces are major system functions and middleware products such as DB2, CICS, and IMS.

- User address spaces
    - User address spaces are known as TSO/E address spaces and are created for every user who logs on to z/OS.
- Application address spaces
    - These are address spaces created for every batch job that runs on z/OS.

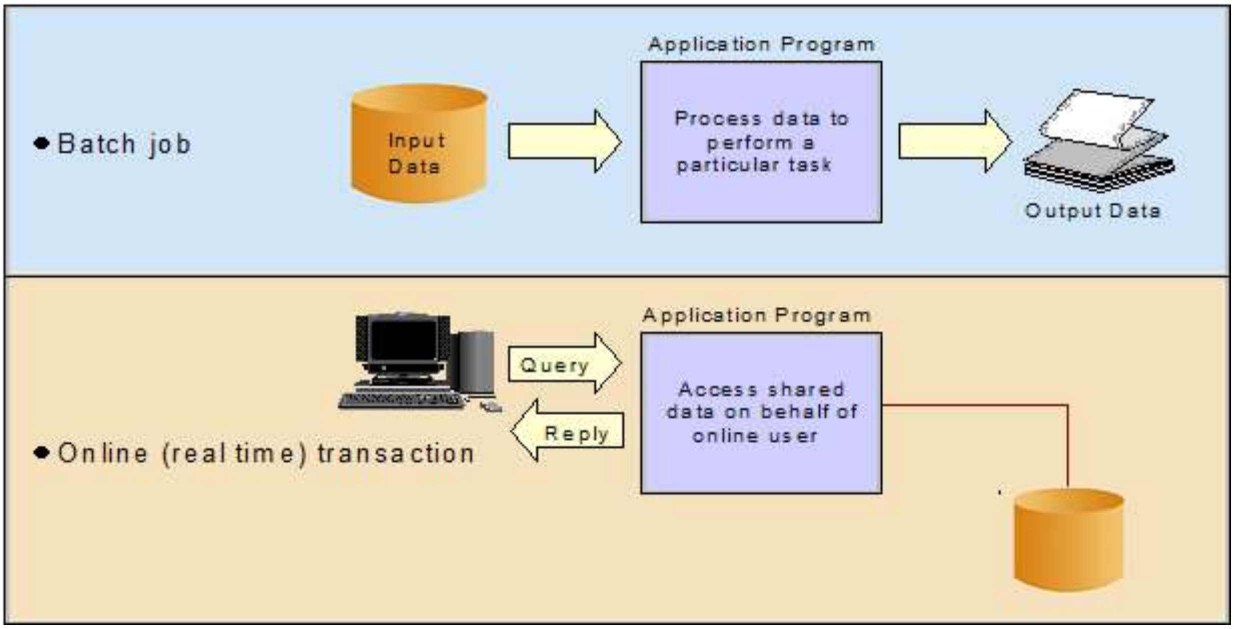
Under SDSF, you can easily identify these address spaces from the JES job IDs.

- **Blank** – Started task created under Master Scheduler
- Eight-character JES job ID starting with;
    - **STC** or **S** – Started tasks
    - **JOB** or **J** – Batch job
    - **TSU** or **T** – TSO user

and these address spaces created under JES.

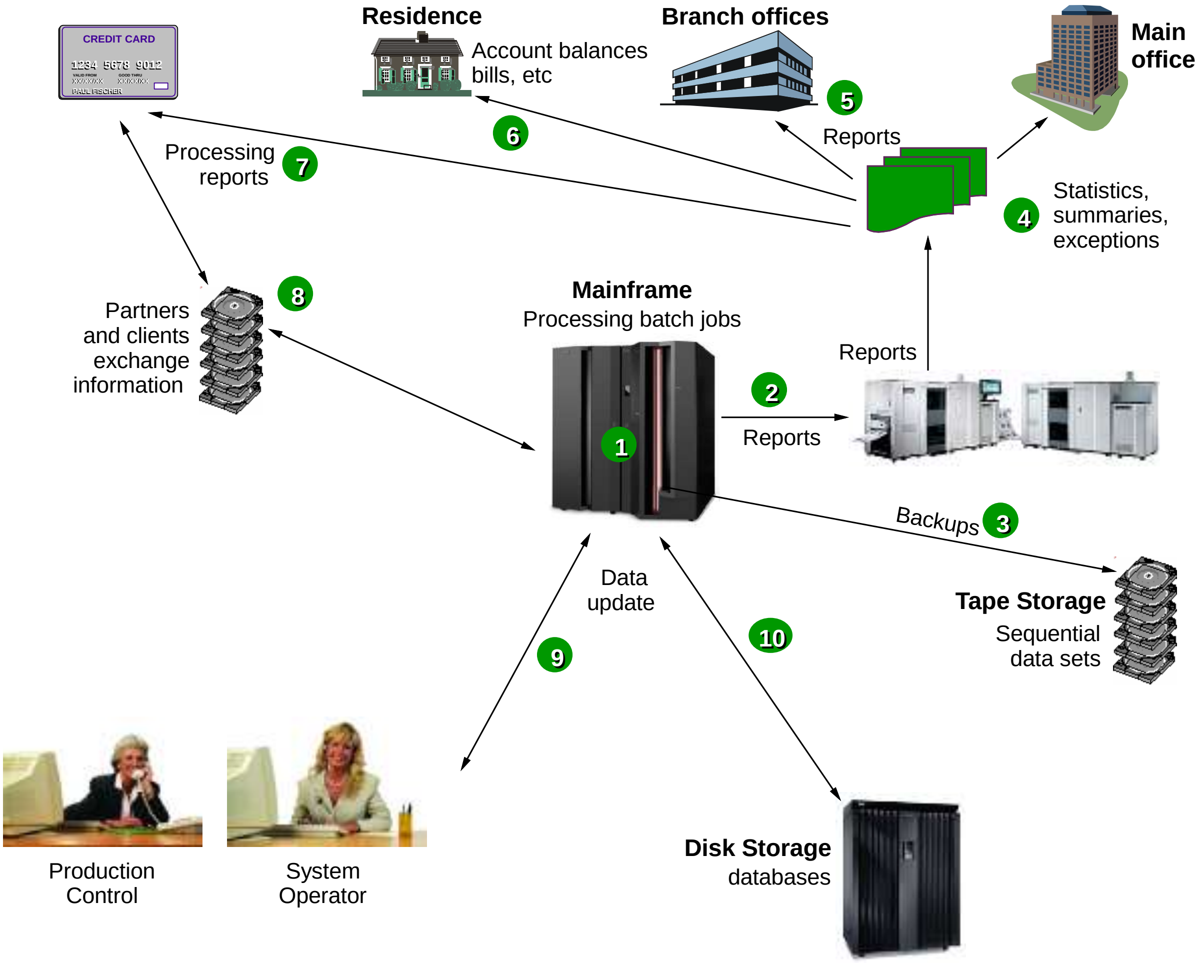
Typical workloads running under zOS

There are typically two types of business workload running under zOS, Batch and Online. But form overall processing point of view, we can add 'Systems' and 'TSO' workload to it.


• Batch job
Input Data
Application Program
Process data to perform a particular task
Output Data
• Online (real time) transaction
Query
Reply
Application Program
Access shared data on behalf of online user

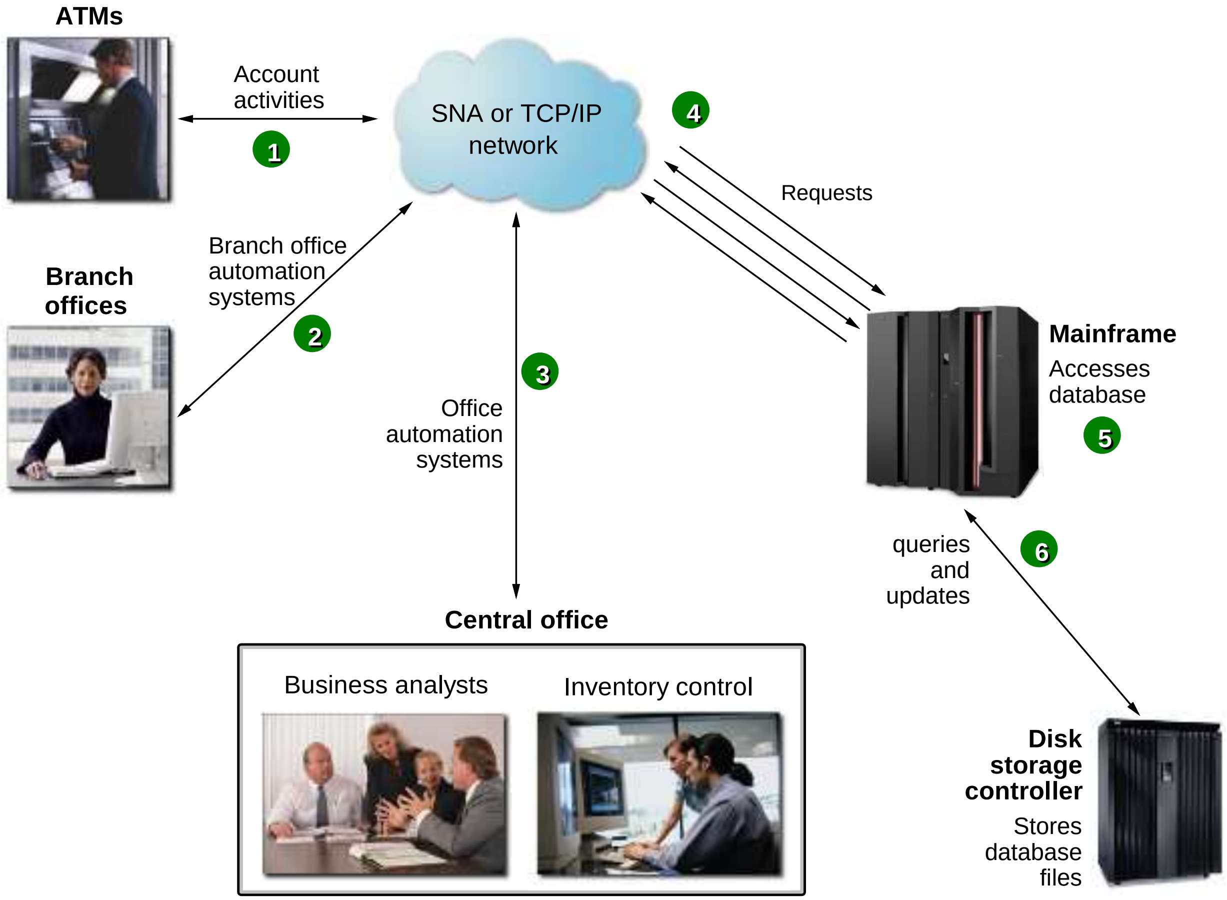
Typical Batch workload

Batch workload is also known as the workload running in the background. Unlike earlier days, processing of batch workload is not confined to a so called ‘Batch Window’ in the night. But, in present days, the batch processing continues 24 x 7, though much of the batch processing still happens in the night. The distribution and printing of reports are no more centralized, but sent to the end point as soon as the processing is completed.



Typical Online workload

The online workload is no more concentrated to day time only. It is running in the systems 24 x 7, though the peak processing happens during the day when everyone is awake. But, most of the business have gone global with the online workload coming to the system, from different regions with different time zone, whole day. So, the online peak workload has really become variable over the day with multiple peaks.



zOS elements and features

Please refer to latest zOS version for details. You may refer to ‘zOS Introduction and Release Guide’, ‘zOS planning for Installation’, and ‘zOS Program Directory’ for details.

z/OS consists of base elements and optional features:

- The *base elements* (or simply *elements*) deliver essential operating system functions. When you order z/OS, you receive all of the base elements.
- The *optional features* (or simply *features*) are orderable with z/OS and provide additional operating system functions.

Optional features are unpriced or priced:

- *Unpriced features* are shipped to you **only if** you order them. If you plan to use any unpriced features, you should order them when you order your base elements. You must not wait until the next release becomes available. Once a release's base elements are no longer orderable, neither are its unpriced features.

  To make ordering easier, the number of unpriced features has been reduced from time to time, mainly through consolidation. In zOS 2.4, the number of unpriced features were two: Communications Server Security Level 3 and z/OS Security Level 3.

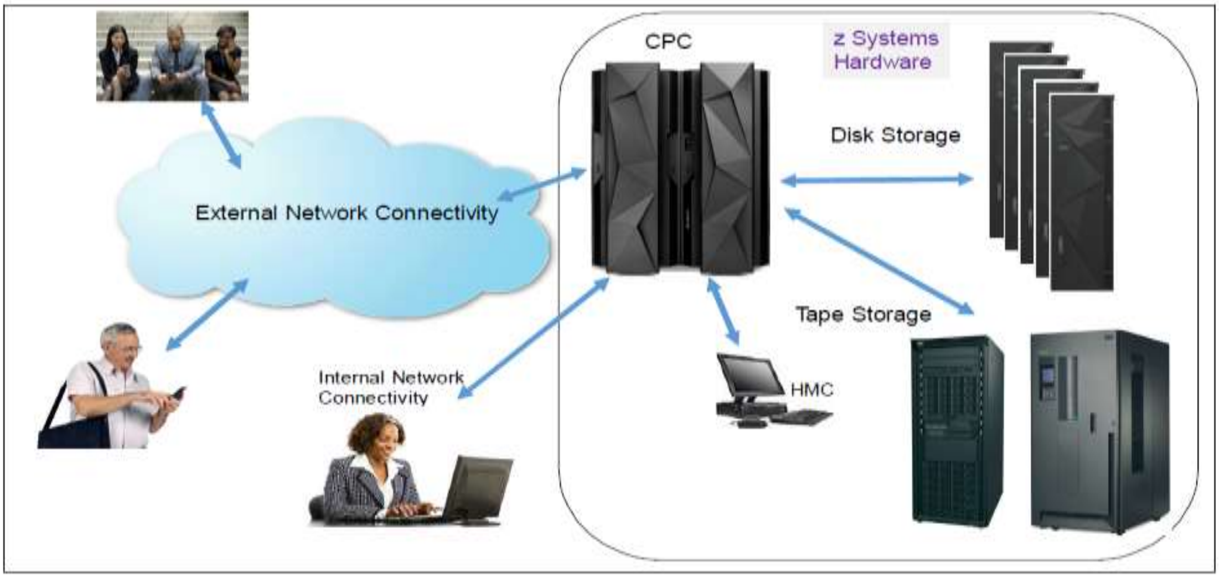
- *Priced features* are **always** shipped to you. When IBM packages your order, it *enables* the priced features that you ordered. These features are ready to use after you install z/OS (and customize it as needed). IBM *disable* the priced features that you did not order. Although they are installed on your system, you cannot use them. Later, if you decide to use them, you notify IBM and you enable them dynamically (which is known as *dynamic enablement*). You dynamically enable by updating parmlib member IFAPRD**xx** and you notify IBM by contacting your IBM representative.

Elements and features may be exclusive or nonexclusive:

- An element or feature is called *exclusive* to z/OS if it exists only within z/OS (not also as a separately orderable product) and if future functional enhancements will occur only within z/OS.
- An element or feature is called *nonexclusive* if it exists both (1) within z/OS and (2) as a separate product.

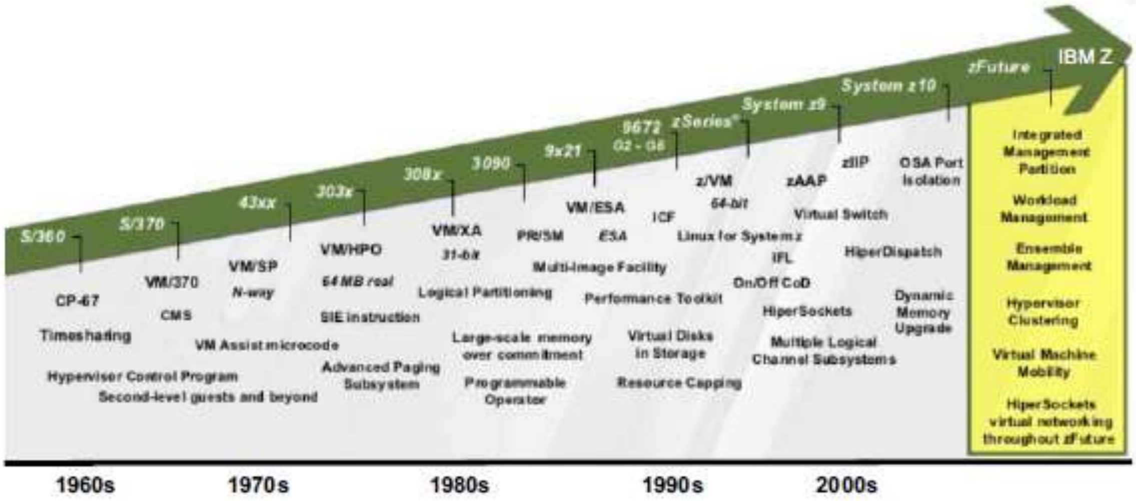
Introduction to mainframe Hardware

The zOS is hosted in one of the LPARs created in system z, which is a physical implementation of z/Architecture. The current system z is called z16.

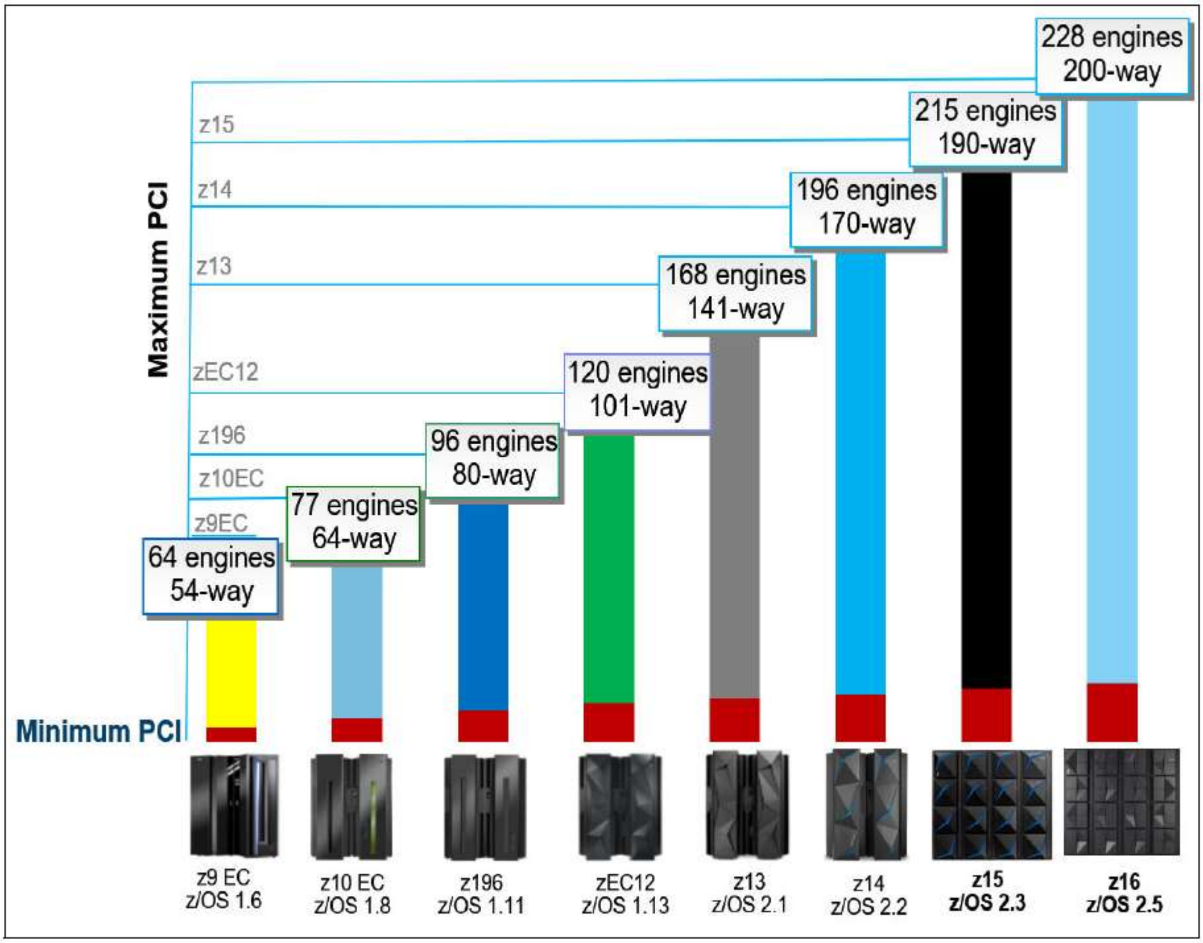


A short History

It represents the history of mainframe over past 60 years.



Evolution of System z



Goals of System z

Integrity of data

A key hardware function is to validate the data it is reading or writing and ensure that it maintains integrity throughout the process. If a physical component fails, the hardware might be able to reconstruct the data. Here, we are referring to the physical aspects of the reading, writing, and movement of the data between hardware components.

Availability of resources

Hardware is designed to have resilience and provide availability levels, depending on customer requirements and configuration options.

Performance

Hardware must perform its duties in a period that meets the expectations of its requesters. To meet the requester's expectations, the hardware must do the following tasks:

- Understand which resources are needed
- Understand the priority of the request
- Allocate resources to requests
- Monitor the progress to complete the request
- Reallocate resources as and when necessary to higher priority requests

Security

Protection from intrusion is paramount. Mechanisms must be in place to detect tampering, provide encryption, and safeguard against unauthorized access. The hardware is securable to various levels of protection.

Communication

The hardware must communicate with other systems by using industry standard protocols that enable the exchange of information or data.

Flexible configuration

As needs change, so might the computer system configuration. The option to deploy specialized processors, add new storage devices, connect to other systems, and reconfigure dynamically to meet the business demands is essential. All these processes must be done with minimal or no disruption to services. This activity often is accomplished by using the Support Element (SE) or the Hardware Management Console (HMC).

Scaling to meet demand

Longer term trends that cause increases in demand for computing resources can be predicted. However, spikes in demand do occur, which cause the need to dynamically allocate further computing resources. The dynamic allocation of these extra resources to solve a spike in demand must be balanced such that the spike does not move to another resource type. Provisioning can be achieved by adding specific resources or through a combination of resources that might provide a cloud solution.

Virtualization of resources

The IBM z Systems platform features unparalleled levels of virtualization at the hardware and software levels. Virtualization provides resources optimization and introduces the granularity that is required by organizations to create and separate environments as needed to perform different functions.

Compatibility

New hardware that adheres to the z/Architecture includes component compatibility and is designed to offer enrichment through new capabilities to applications.

Control

Control points to monitor, configure, maintain, and add components must be available to record and take actions where appropriate on the hardware.

Operating system integration

IBM z Systems hardware integrates with several operating systems and must service requests from each operating system and allocate resources according to each operating system's needs.

Processing Units (PUs)

All IBM z Systems CPCs are assigned a unique machine type. Each machine type is available in different models. Each model has a different number of PUs, memory, and other resources.

The model that you choose depends on how many PUs you require to run your workloads. There are different types of PUs. You must also define the type of PU you want (they do not have to be all the same type). You can have a combination of different types of PU.

The PUs essentially are physically the same, but differentiated by their characteristics. The PUs can be characterized in advance or dynamically. Certain PU characterizations are better-suited to specific types of tasks than others. The following PU characterizations used:

- Central processor - **CPU/CP/GP/GCP** - A CPU is a general-purpose processor that can execute all the possible workloads.
- Integrated Facility for Linux - **IFL** - This type of PU is only able to execute native Linux and Linux under z/VM.
- Integrated Coupling Facility - **ICF** - This type of PU is only able to execute the CFCC operating system (Coupling Facility).
- z Integrated Information Processor - **zIIP** - This type of PU is to run in z/OS only, for eligible workloads such as DB2, DDF and many more.
- System Assist Processor **- SAP -** A System Assist Processor (SAP) is a PU that runs the Channel Subsystem Licensed Internal Code.
- Integrated Firmware Processor – **IFP -** Used for infrastructure management. It is predefined and standard with the platform.
- Spare Processor - A spare PU is a PU that can replace, automatically and transparently, any falling PU in the same drawer, or in a different drawer.

Cryptographic Processors

Transactions and sensitive data require higher securable options. The CPC has multiple cryptographic levels. The first level is the cryptographic assist implementation, which is known as *CPACF*. It delivers cryptographic and hashing functions in support of clear-key operations.

It is physically implemented in the chip by the compression and cryptography accelerators. Each core has one dedicated coprocessor (CoP) that integrates the CPACF and the compression unit. The CPACF offers the full complement of the AES algorithm, SHA, and DES algorithm.

CPACF must be explicitly enabled by using a no-charge enablement feature. This requirement means that although the hardware has this feature, you must request its activation or it cannot be used. The CPACF is available for every PU that is characterized as CP, IFL, or zIIP.

Increased cryptography is available with the Crypto Express5S feature, and is a card that is installed in the PCIe I/O drawer. The Crypto Express5S offers a state-of-the-art, tamper-resistant cryptographic coprocessor for secure-key operations along with new hardware assists to encrypt data quickly. This capability allows data transfer across the internet to support public and private cloud and mobile workloads.

zEDC Compression

zEDC compression is a mechanism for representing the same amount of information in a smaller number of bytes. This mechanism is an optional feature and provides hardware-based acceleration for data compression and decompression. It helps to improve cross platform data exchange, reduce CPU consumption, and save disk space. Similar to the Crypto Express 5S card, it is also installed in the PCIe drawers.

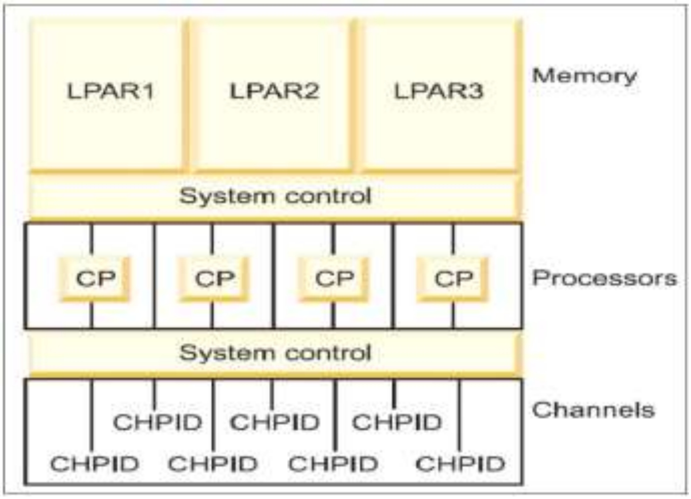
For resilience and to maintain availability, it is recommended that a minimum of two zEDC Express features be installed.

Virtualization

The ability to share resources is one of the major strengths of the mainframe. z/Architecture contains many facilities in its hardware and software that facilitate resource virtualization.

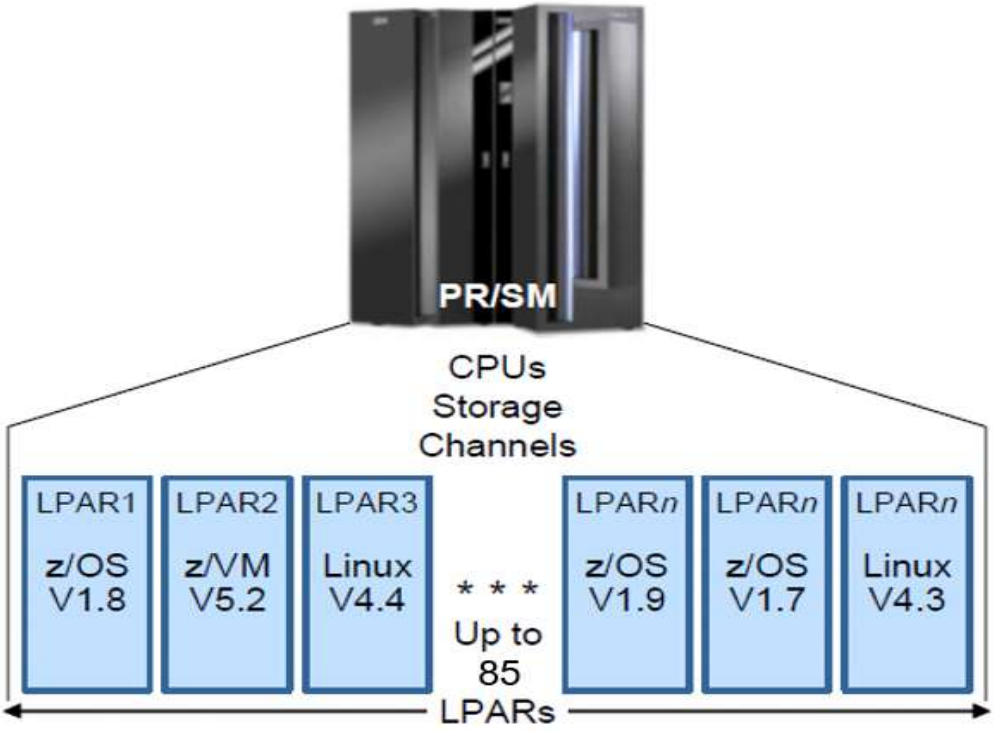
Overutilizing the mainframe environment involves creating virtual systems through logical partitions (LPARs) and assigning virtual resources, such as memory and I/O channels, to those systems. Depending on configuration options, resources can be dynamically added to or removed from these logical partitions.

Processor Resource/Systems Manager (PR/SM) is a hypervisor that is integrated with all mainframe elements. PR/SM maps physical resources into virtual resources so that many logical partitions can share the physical resources.



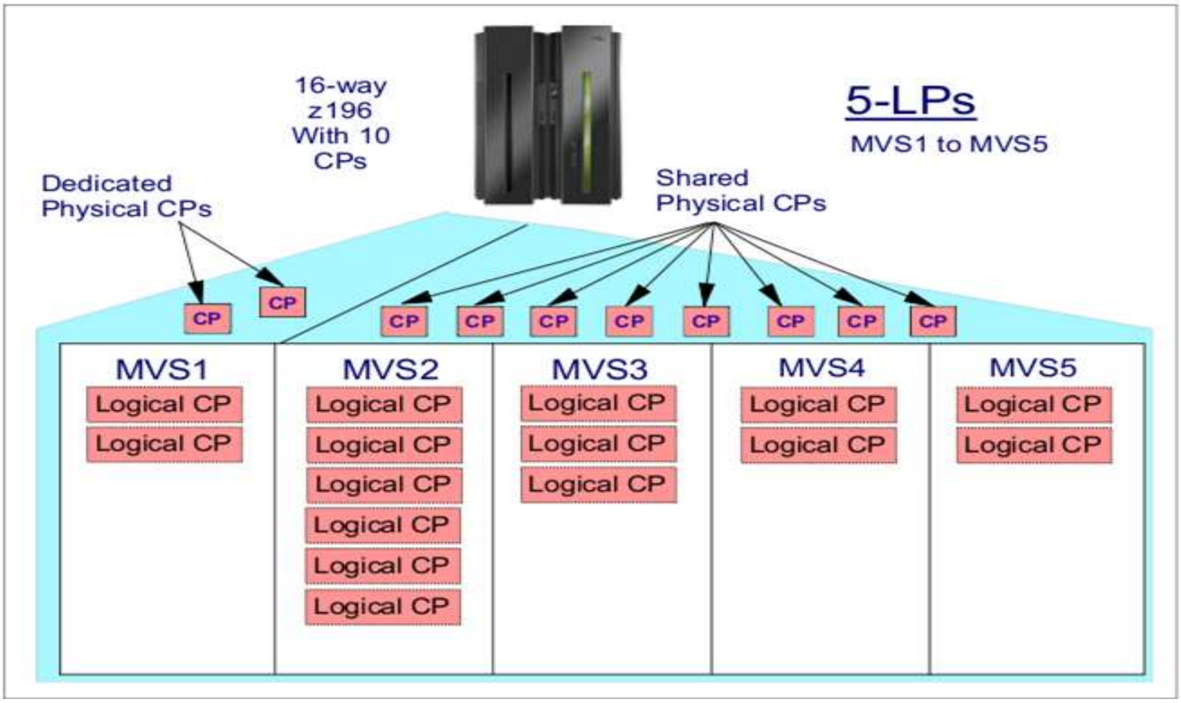
PR/SM

PR/SM (Processor Resource/System Manager) is a type-1 Hypervisor (a virtual machine monitor) that allows multiple logical partitions to share physical resources such as CPUs, memory, I/O channels, and LAN interfaces.



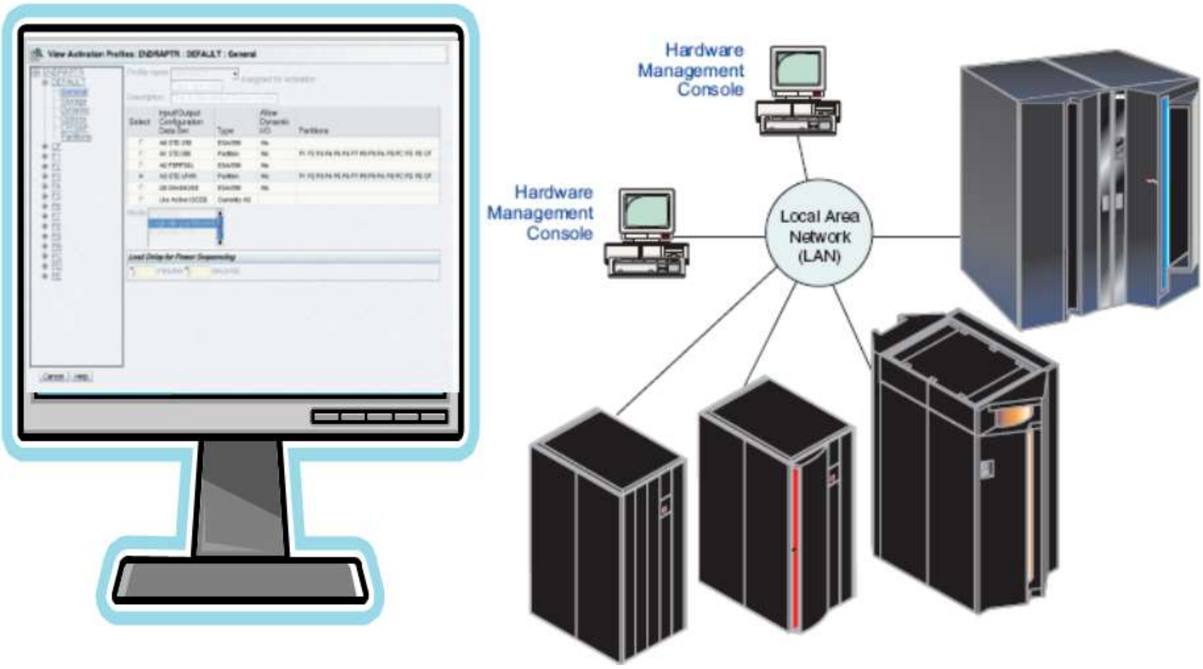
Physical and Logical CPs

The physical CPs which are available in the CEC while the logical CPs are allocated to the LPAR. The total number of allocated Logical CPs to various LPARs could be much more than the number of shared Physical CPs. In the picture below, the CEC has 10 Physical CPs out of which 2 CPs are dedicated to MVS1 LPAR. And the other 8 Physical CPs are shared among other four LPARs with an allocation of total number of 13 Logical CPs.



Hardware Management Console (HMC)

The Hardware Management Console (HMC) acts as the operational focal point for one or multiple IBM Z mainframes, that are attached to a mainframe cluster.



Support Element (SE)

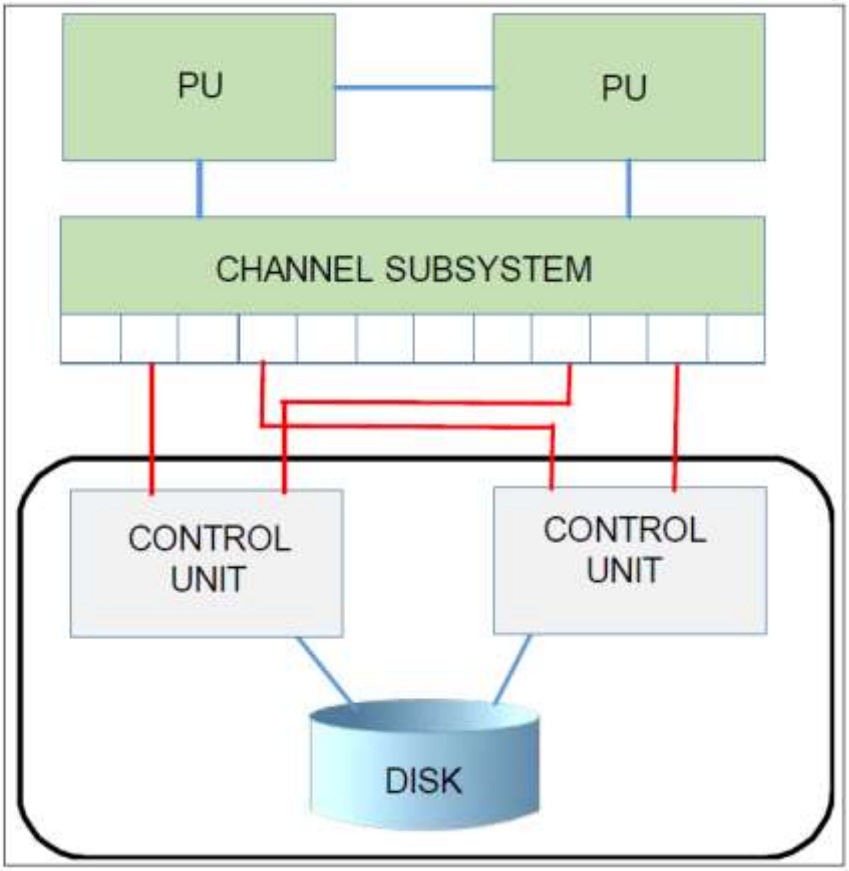
The Support Element (SE) console acts as the single point of control for one IBM z mainframe. It is physically connected to the processor and located in a CPC frame. In the following picture, the SEs are the two ThinkPads sitting inside the CEC.

IO Channel Subsystems

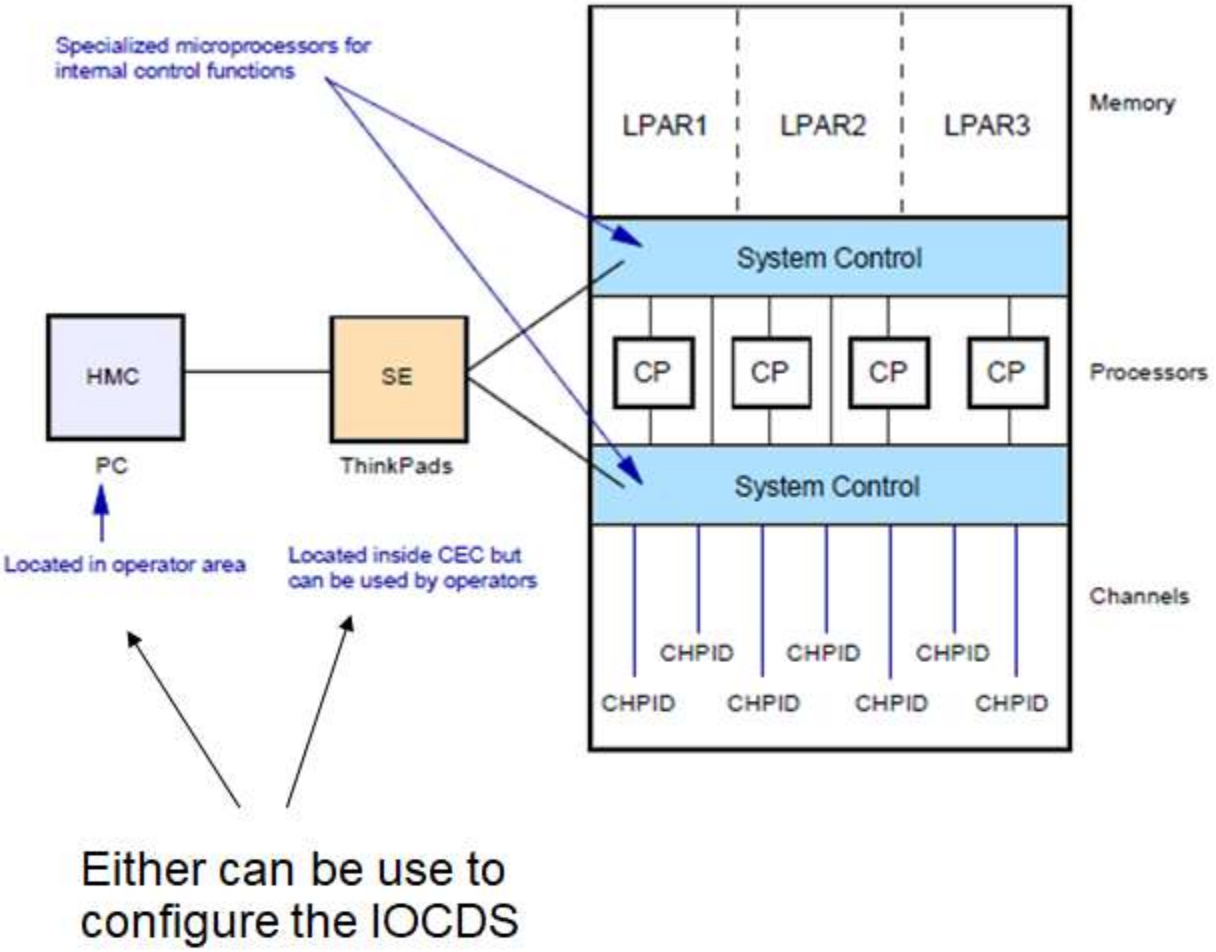
The mechanisms that are used to control I/O operations are collectively called the *channel subsystem* (CSS). The CSS directs the flow of information between I/O devices and main storage. A CSS relieves the PUs of the tasks of communicating directly with I/O devices and permits data processing to proceed on the PUs while other data is transferred concurrently to and from the I/O devices.

Channel Path: A channel path is a single interface between a system and one or more control units. A CSS has up to 256 channels attached to it.

Subchannels: A subchannel provides the logical representation of an I/O device to the program and contains the necessary information to sustain an I/O operation to the device it represents.



System Control and Partitioning

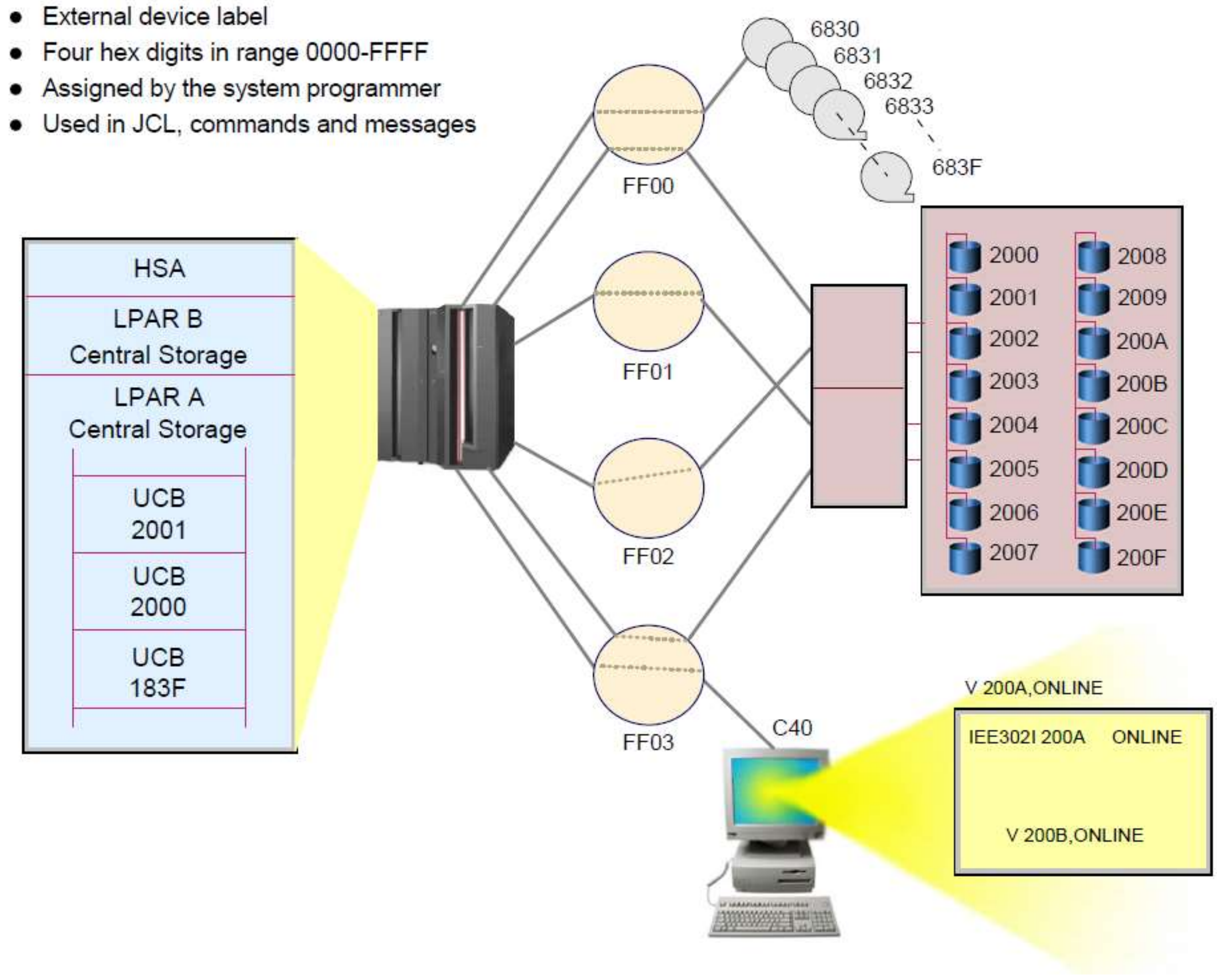


Logical Partions (LPARs)

- Logical partitions are, in practice, equivalent to separate mainframes.
- Each LPAR runs its own operating system (OS).
    - This OS can be any mainframe operating system; there is no need to run z/OS only, for example, in each LPAR. The installation planners may elect to share I/O devices across several LPARs, but this is a local decision.

- The system administrator can assign one or more system processors for the exclusive use by an LPAR.
  - Alternately, the administrator can allow all processors to be used on some or all LPARs.
  - Here, the system control functions (often known as microcode or firmware) provide a dispatcher to share the processors among the selected LPARs.
- The administrator can specify a maximum number of concurrent processors executing in each LPAR.
- The administrator can provide weightings for different LPARs, for example, specifying that LPAR1 should receive twice as much processor time as LPAR2.
- The operating system in each LPAR performs an IPL separately, has its own copy of its operating system, has its own operator console (if needed), and so on.
- If the system in one LPAR fails or is taken down for maintenance, it has no effect on the other LPARs.

Device Addressing



- External device label
- Four hex digits in range 0000-FFFF
- Assigned by the system programmer
- Used in JCL, commands and messages

```
V 200A,ONLINE
IEE302I 200A     ONLINE
V 200B,ONLINE
```

Network Connectivity

The network connectivity covers external connectivity and specialized internal connections for intra-system communication.

The IBM Open Systems Adapter-Express (OSA-Express) is a card that can be installed into the PCIe drawers. It can provide LAN connectivity. In addition, the OSA-Express can assume some of the PUs workload for several functions of the TCP/IP stack, which results in significant performance benefits by offloading processing from the operating system.

HiperSockets

HiperSockets provides LAN connectivity across multiple system images on the same z Systems platform (CEC) by performing memory-to-memory data transfers in a secure way. The HiperSockets function eliminates the use of I/O subsystem operations and the use of an external network connection to communicate between LPARs in the same CPC.

IBM 10 GbE RoCE Express

The 10 Gigabit Ethernet (10 GbE) RoCE Express feature uses Remote Direct Memory Access (RDMA) over Converged Ethernet (RoCE) to provide fast memory-to-memory communications between two CPCs or within the same CPC.

Shared Memory Communications - Direct Memory Access

The platform features a communications protocol that is called Shared Memory Communications - Direct Memory Access (SMC-D). SMC-D is intended for communications *within* the same CPC, which optimizes operating systems.

Server Time Protocol (STP)

In olden days, we use to have a Sysplex Timer.

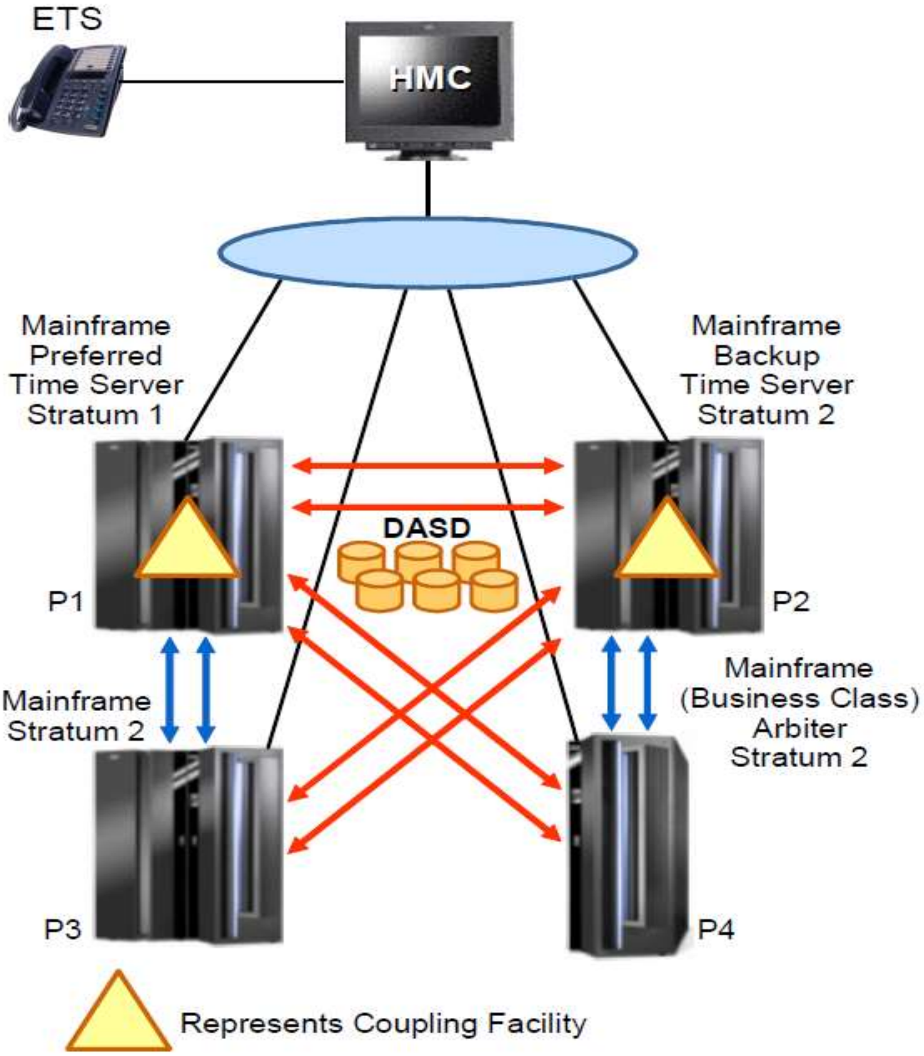
- Today's implementation uses the Server Time Protocol (STP), which is a server-wide facility that is implemented in the Licensed Internal Code (LIC).

- STP presents a single view of time to Processor Resource/Systems Manager (PR/SM), and is designed to provide the capability for multiple mainframe servers to maintain time synchronization with each other.

    - NOTE: It is the follow-up to the Sysplex Timer.

- The Sysplex Timer distributes time to multiple servers in a star pattern, that is, the Sysplex Timer is the star, and its time signals distribute out from it to all attached servers.

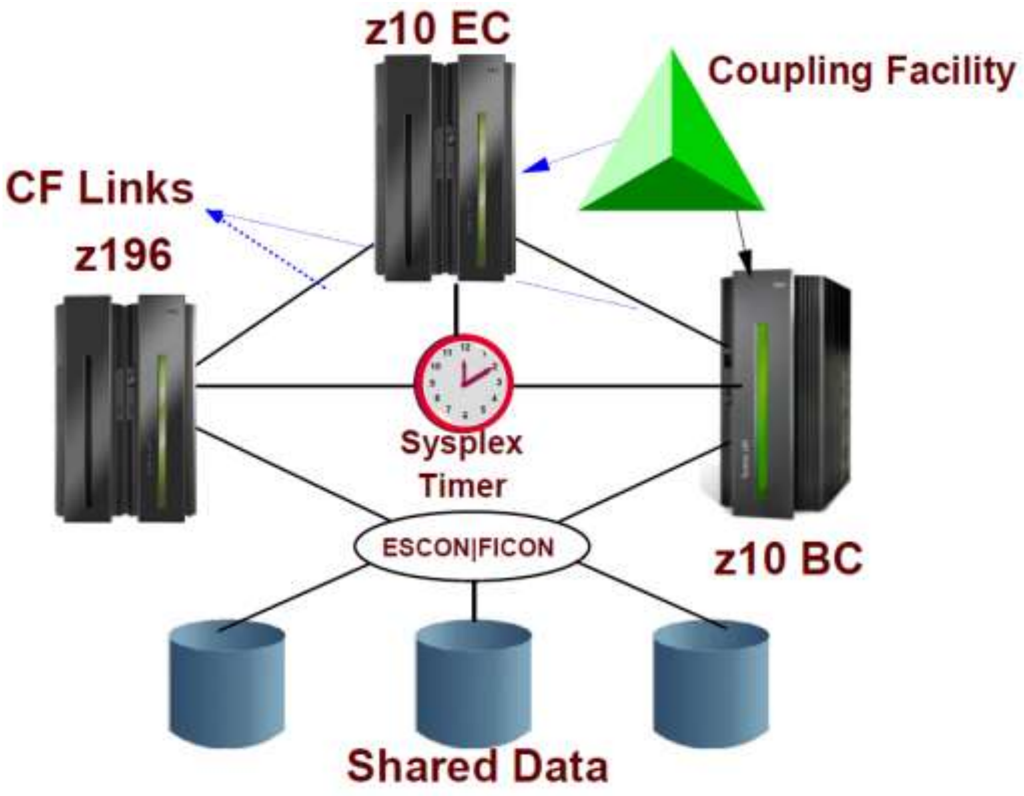
- The signals from the Sysplex Timer are used to increment or step the TOD clocks in the attached server.
    - Unlike the Sysplex Timer, STP passes time messages in layers, or strata. The top layer (Stratum 1) distributes time messages to the layer immediately below it (Stratum 2).
    - Stratum 2 in turn distributes time messages to Stratum 3 and so on.
- In a timing network based on STP, a stratum is used as a means to define the hierarchy of a server in the timing network.
- A Stratum 1 server is the highest level in the hierarchy in the STP network.



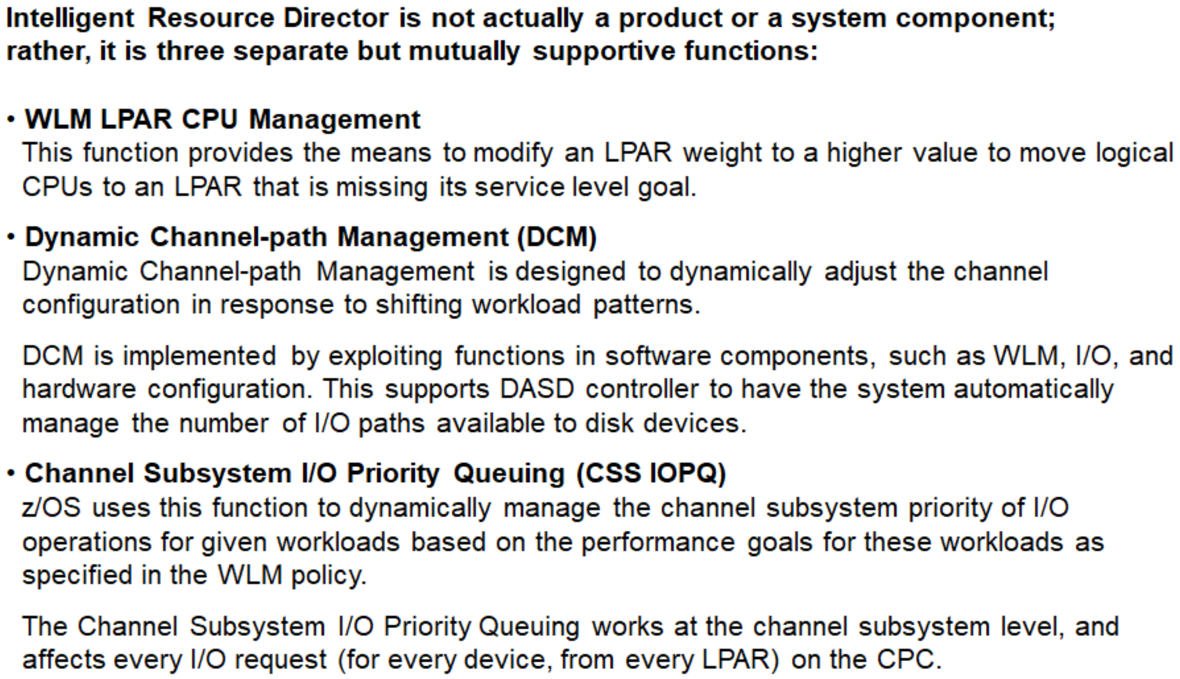
Coupling Facility (CF)

A CF functions largely as a fast scratch pad. It is used for three purposes:

- Locking information that is shared among all attached systems
- Cache information (such as for a database) that is shared among all attached systems
- Data list information that is shared among all attached systems



Intelligent Resource Director (IRD)



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**
  This function provides the means to modify an LPAR weight to a higher value to move logical CPUs to an LPAR that is missing its service level goal.
- **Dynamic Channel-path Management (DCM)**
  Dynamic Channel-path Management is designed to dynamically adjust the channel configuration in response to shifting workload patterns.

  DCM is implemented by exploiting functions in software components, such as WLM, I/O, and hardware configuration. This supports DASD controller to have the system automatically manage the number of I/O paths available to disk devices.
- **Channel Subsystem I/O Priority Queuing (CSS IOPQ)**
  z/OS uses this 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.

  The Channel Subsystem I/O Priority Queuing works at the channel subsystem level, and affects every I/O request (for every device, from every LPAR) on the CPC.

Capacity growth

There are multiple ways you can provision additional processing resources to your systems.

Customer Initiated Upgrade (CIU): The CIU feature enables a customer to order permanent capacity upgrades rapidly and download them without disrupting applications already running on the machine.

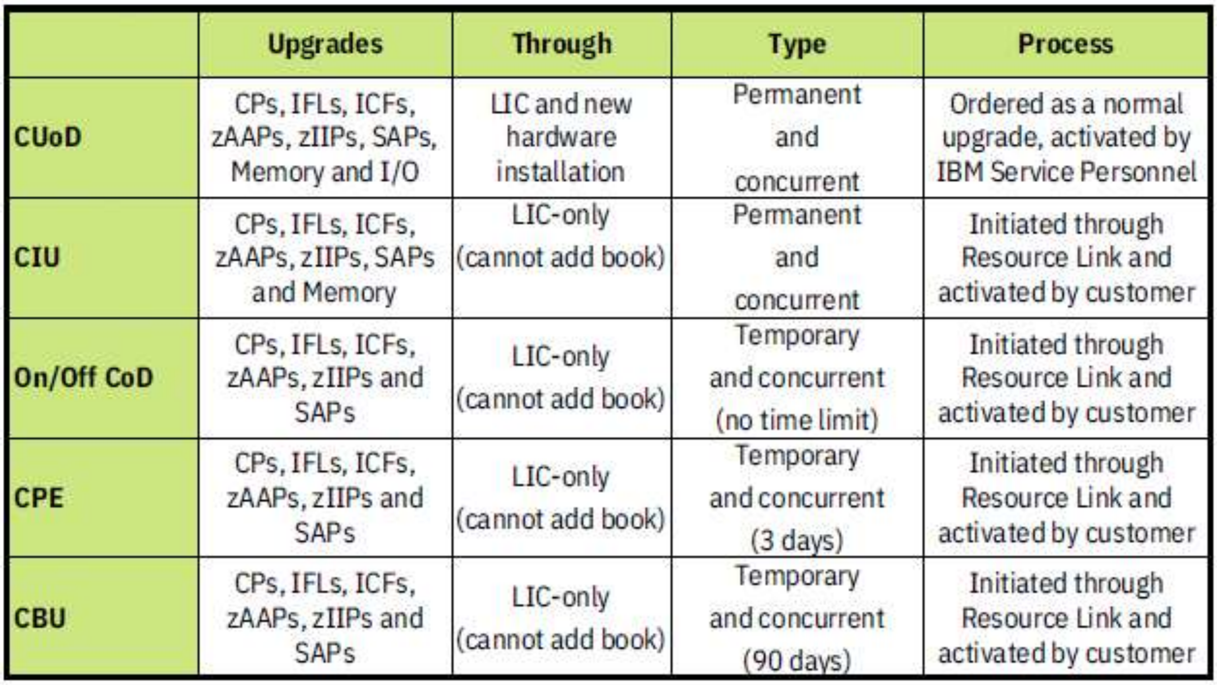
- When extra processing power becomes necessary, an administrator simply uses a two-step process:
  - Navigates to special web-based link to order an upgrade.
  - Uses the Remote Service Facility on the Hardware Management Console (HMC) to download and activate preinstalled inactive processors (uncharacterized engines) or memory.

On/Off Capacity on Demand (On/Off CoD): This feature is available through CIU, and uses On/Off CoD for temporary increases in processor capacity.

- With temporary processor capacity, customers manage both predictable and unpredictable surges in capacity demands.
- They can activate and deactivate quickly and efficiently as the demands on their organization dictates to obtain additional capacity that they need, when they need it, and the machine will keep track of its usage.

Capacity Backup (CBU): Customers can use CBU to add temporary processing capacity to a backup machine in the event of an unforeseen loss of server capability because of an emergency.

- With CBU, customers can divert entire workloads to backup servers for up to <nn> days (please check it in your contract).



	Upgrades	Through	Type	Process
CUoD	CPs, IFLs, ICFs, zAAPs, zIIPs, SAPs, Memory and I/O	LIC and new hardware installation	Permanent and concurrent	Ordered as a normal upgrade, activated by IBM Service Personnel
CIU	CPs, IFLs, ICFs, zAAPs, zIIPs, SAPs and Memory	LIC-only (cannot add book)	Permanent and concurrent	Initiated through Resource Link and activated by customer
On/Off CoD	CPs, IFLs, ICFs, zAAPs, zIIPs and SAPs	LIC-only (cannot add book)	Temporary and concurrent (no time limit)	Initiated through Resource Link and activated by customer
CPE	CPs, IFLs, ICFs, zAAPs, zIIPs and SAPs	LIC-only (cannot add book)	Temporary and concurrent (3 days)	Initiated through Resource Link and activated by customer
CBU	CPs, IFLs, ICFs, zAAPs, zIIPs and SAPs	LIC-only (cannot add book)	Temporary and concurrent (90 days)	Initiated through Resource Link and activated by customer

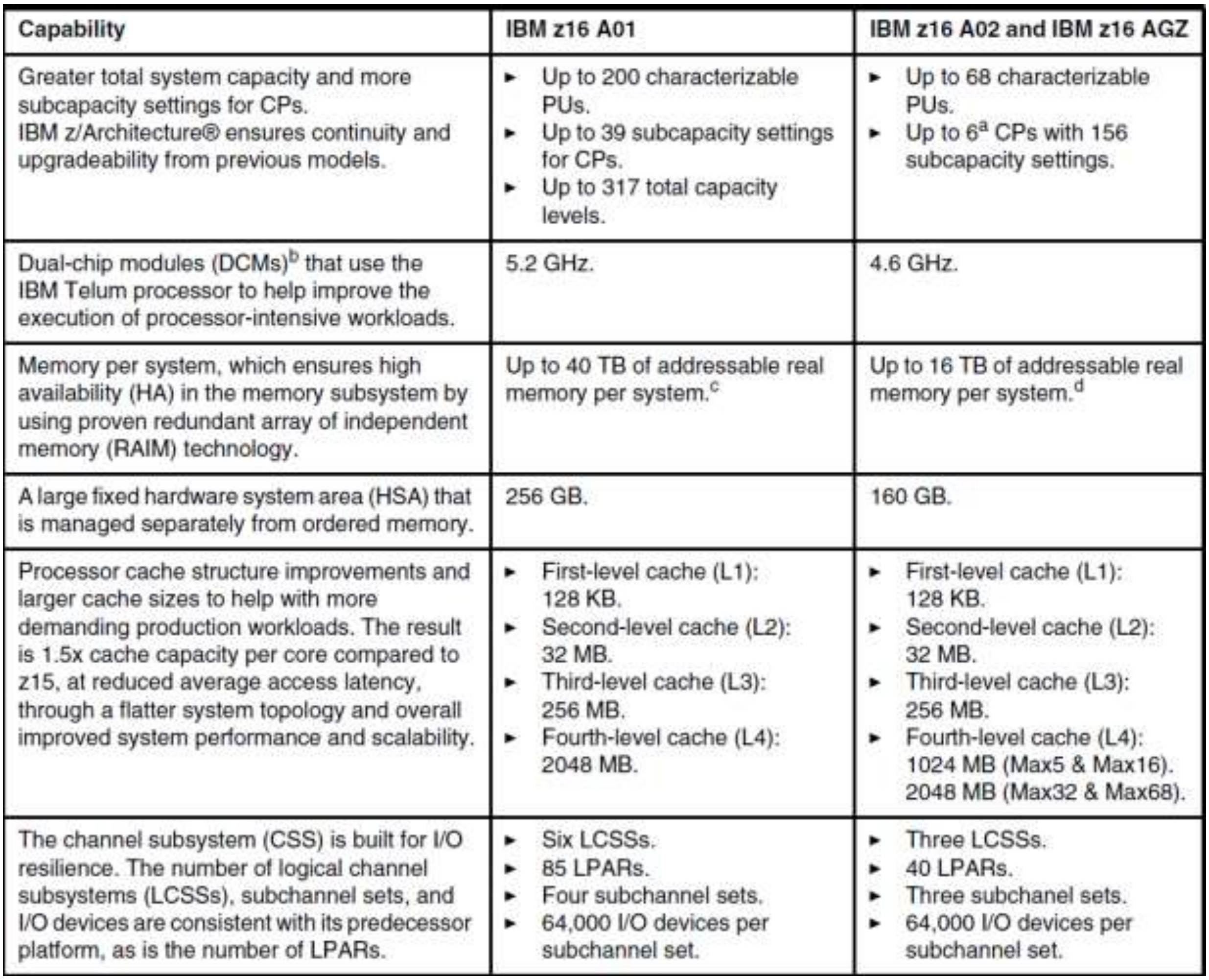
Z16 - 3931

IBM z16 aligns with the American Society of Heating, Refrigerating, and Air-Conditioning Engineers (ASHRAE) Class A3 data center guidelines. IBM z16 is available in three different configuration options:

IBM z16 A01 is built with a 19-inch format that scales 1 - 4 frames, depending on the configuration. IBM z16 A01 ensures continuity and upgradeability from IBM z15 T01 and IBM z14 M0x. It has five orderable features: Max39, Max82, Max125, Max168, and Max200.

IBM z16 A02 is built with a 19-inch format single frame. IBM z16 A02 ensures continuity and upgradeability from IBM z15 T02 and IBM z14 ZR1. There are four orderable features: Max5, Max16, Max32, and Max68.

IBM z16 AGZ indicates a rack-mount configuration that allows the core compute, I/O, and networking features to be installed into and powered by a client-designated rack with power distribution units (PDUs), respectively. IBM z16 AGZ ensures continuity and upgradeability from IBM z15 T02 and IBM z14 ZR1. The rack-mount configuration options are under a combined AGZ warranty umbrella and orderable as Max5, Max16, Max32, and Max68.

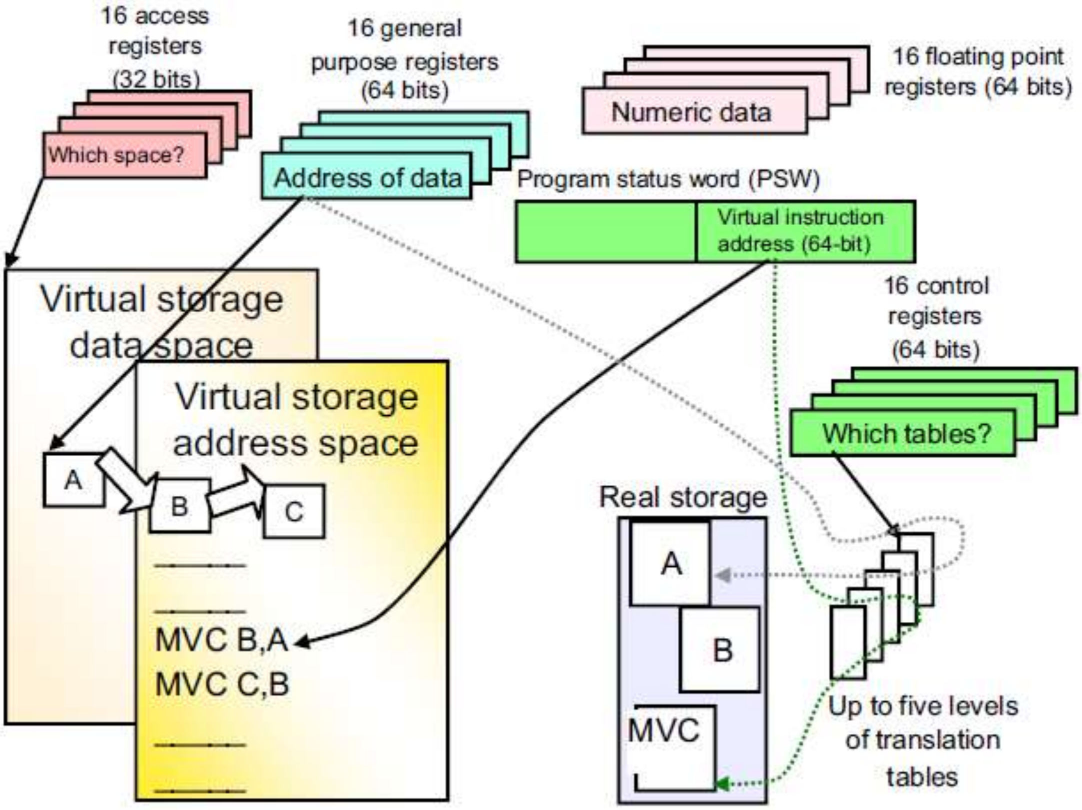


Capability	IBM z16 A01	IBM z16 A02 and IBM z16 AGZ
Greater total system capacity and more subcapacity settings for CPs. IBM z/Architecture® ensures continuity and upgradeability from previous models.	► Up to 200 characterizable PUs. ► Up to 39 subcapacity settings for CPs. ► Up to 317 total capacity levels.	► Up to 68 characterizable PUs. ► Up to 6[a] CPs with 156 subcapacity settings.
Dual-chip modules (DCMs)[b] that use the IBM Telum processor to help improve the execution of processor-intensive workloads.	5.2 GHz.	4.6 GHz.
Memory per system, which ensures high availability (HA) in the memory subsystem by using proven redundant array of independent memory (RAIM) technology.	Up to 40 TB of addressable real memory per system.[c]	Up to 16 TB of addressable real memory per system.[d]
A large fixed hardware system area (HSA) that is managed separately from ordered memory.	256 GB.	160 GB.
Processor cache structure improvements and larger cache sizes to help with more demanding production workloads. The result is 1.5x cache capacity per core compared to z15, at reduced average access latency, through a flatter system topology and overall improved system performance and scalability.	► First-level cache (L1): 128 KB. ► Second-level cache (L2): 32 MB. ► Third-level cache (L3): 256 MB. ► Fourth-level cache (L4): 2048 MB.	► First-level cache (L1): 128 KB. ► Second-level cache (L2): 32 MB. ► Third-level cache (L3): 256 MB. ► Fourth-level cache (L4): 1024 MB (Max5 & Max16). 2048 MB (Max32 & Max68).
The channel subsystem (CSS) is built for I/O resilience. The number of logical channel subsystems (LCSSs), subchannel sets, and I/O devices are consistent with its predecessor platform, as is the number of LPARs.	► Six LCSSs. ► 85 LPARs. ► Four subchannel sets. ► 64,000 I/O devices per subchannel set.	► Three LCSSs. ► 40 LPARs. ► Three subchannel sets. ► 64,000 I/O devices per subchannel set.

PUs

Feature name	Number of CPC drawers	Feature code	Characterizable processor units	Standard SAPs	Spares
Max39	1	0667	0 - 39	5	2
Max82	2	0668	0 - 82	10	2
Max125	3	0669	0 - 125	15	2
Max168	4	0670	0 - 168	20	2
Max200	4	0671	0 - 200	24	2

Registers


16 access registers (32 bits)
Which space?
16 general purpose registers (64 bits)
Address of data
Numeric data
16 floating point registers (64 bits)
Program status word (PSW)
Virtual instruction address (64-bit)
16 control registers (64 bits)
Which tables?
Virtual storage data space
Virtual storage address space
A
B
C
MVC B,A
MVC C,B
Real storage
A
B
MVC
Up to five levels of translation tables

There are 16 general registers (sometimes called *general-purpose registers*) that might be used by programs as accumulators in general arithmetic operations or as base registers or index registers when referencing data in virtual storage.

The access registers are in fact not normally used by ordinary programs, as these programs normally do not use data spaces. When access registers are not used, data are found in the address space. There are some bits in the PSW which indicate if access registers are used or not. Access registers can be manipulated in assembler programs only.

The 16 floating-point registers are for mathematical manipulation of numeric data in floating-point format.

There are 16 control registers that are used by the system software to provide special control information to various system facilities such as DAT. Example: CR1 contains the (real) address of the translation tables for the address space.

On the slide, virtual address-to-real address translation is shown with arrows through the translation tables.

Note: Both the instruction address in the PSW and the data addresses in the registers are translated from virtual to real by the DAT-box.

The PSW is a special hardware register that works in conjunction with the control registers to govern exactly how each instruction is executed.

In general, the PSW is used to control instruction sequencing and to indicate the status of a central processor in relation to the program currently being executed. The contents of the PSW are manipulated by the system software on behalf of application programs.

When an interrupts occurs, the current contents of the PSW are stored and new values are loaded into the PSW. This gives control of the central processor to another program.

Hypervisors

There are different server virtualization approaches that can be used to enable multiple operating systems to run on a physical server.

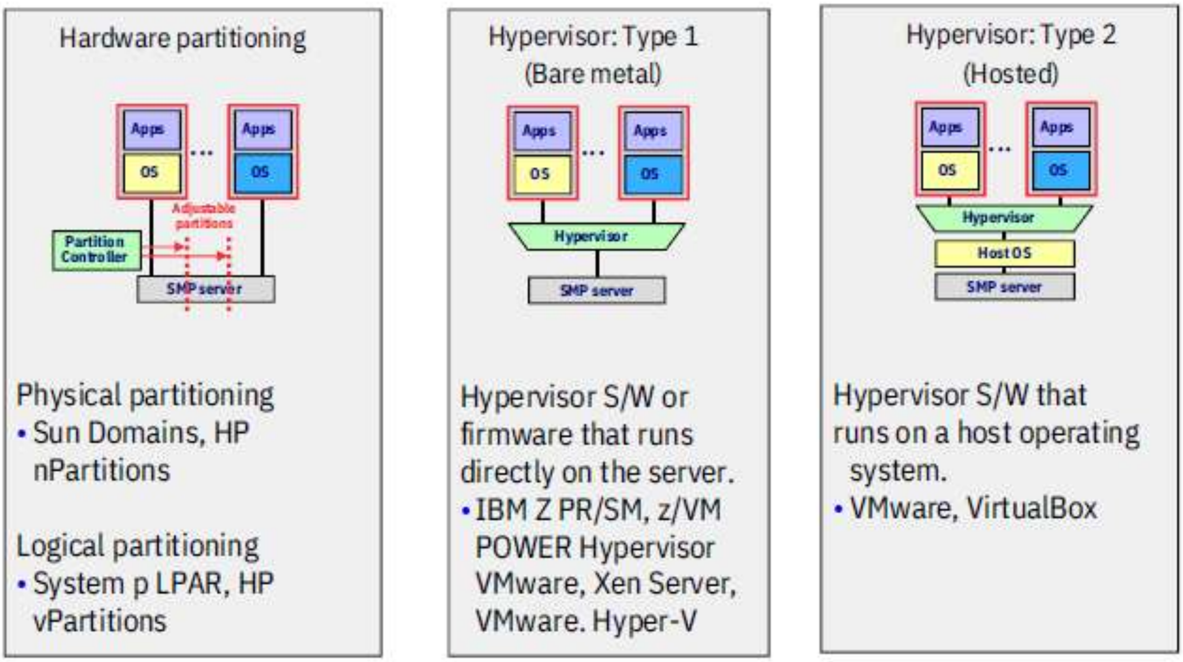
Hardware partitioning (physical or logical) can be used to subdivide the server resources into fractions, each of which can run an operating system. The degree of resource sharing, if any, is dependent on the hardware platform.

Today, all IBM Z servers use logical partitioning, but in fact that partitioning is provided at the firmware layer with the Processor Resource System Manager (PR/SM) hypervisor code. Hypervisors use a thin layer of code to achieve fine-grained, dynamic resource sharing

There are basically two types of hypervisors that can be identified:

- Type 1 (or native, bare metal) hypervisors run directly on the host's hardware to control the hardware and to manage guest operating systems.
    - A guest operating system thus runs on another level above the hypervisor. some examples include IBM Z, PR/SM, zVM, Power Hypervisor, Oracle VM Server for SPARC, the Citrix XenServer, KVM, VMware ESX/ESXi, and Microsoft Hyper-V hypervisor.
- Type 2 (or hosted) hypervisors run within a conventional operating system environment. With the hypervisor layer as a distinct second software level, guest operating systems run at the third level above the hardware.
    - VMware Workstation and VirtualBox are examples of Type 2 hypervisors.

In other words, Type 1 hypervisor runs directly on the hardware; a Type 2 hypervisor runs on another operating system. Type 1 hypervisors with high efficiency and availability will become dominant for enterprise type servers, whereas type 2 hypervisors will be mainly for clients where host operating system integration is desirable.



Hardware partitioning

Physical partitioning

- Sun Domains, HP nPartitions

Logical partitioning

- System p LPAR, HP vPartitions

Hypervisor: Type 1 (Bare metal)

Hypervisor S/W or firmware that runs directly on the server.

- IBM Z PR/SM, z/VM POWER Hypervisor VMware, Xen Server, VMware. Hyper-V

Hypervisor: Type 2 (Hosted)

Hypervisor S/W that runs on a host operating system.

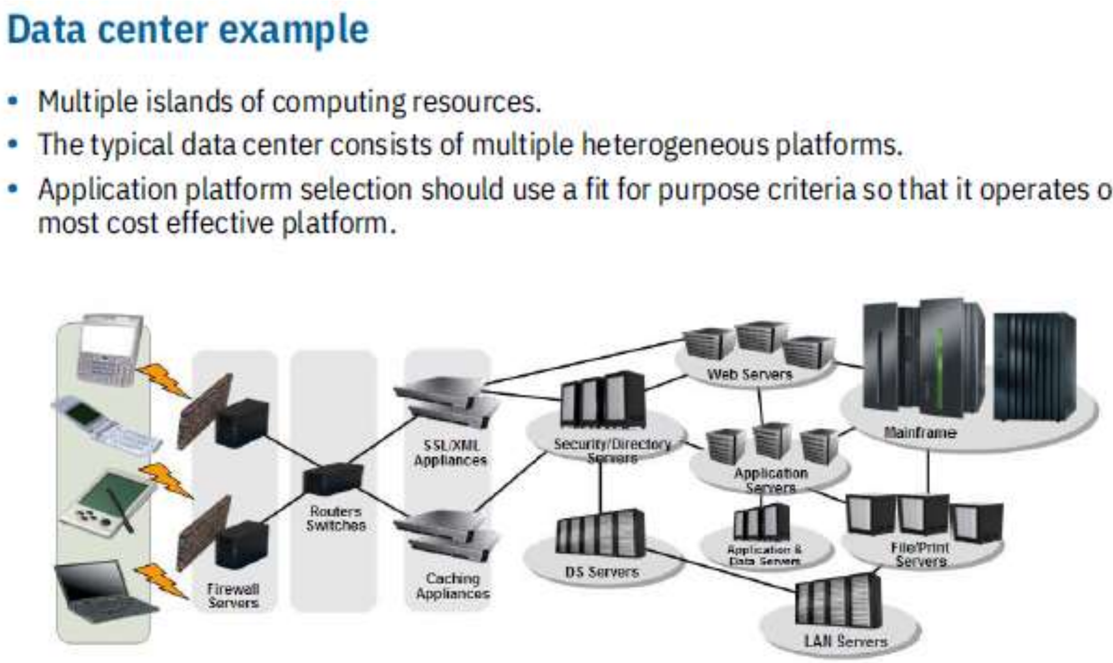
- VMware, VirtualBox

Data Centre

At its simplest, a data center is a physical facility that organizations use to house their critical applications and data. A data center's design is based on a network of computing and storage resources that enable the delivery of shared applications and data. The key components of a data center design include routers, switches, firewalls, storage systems, servers, and application-delivery controllers.

Organizations may have multiple data centers to support their business and name their data centers as:

- Primary Data Centre
- Secondary Data Centre
- Tertiary Data Centre
- Production Data Centre
- Disaster Recovery Data Centre
- Etc.

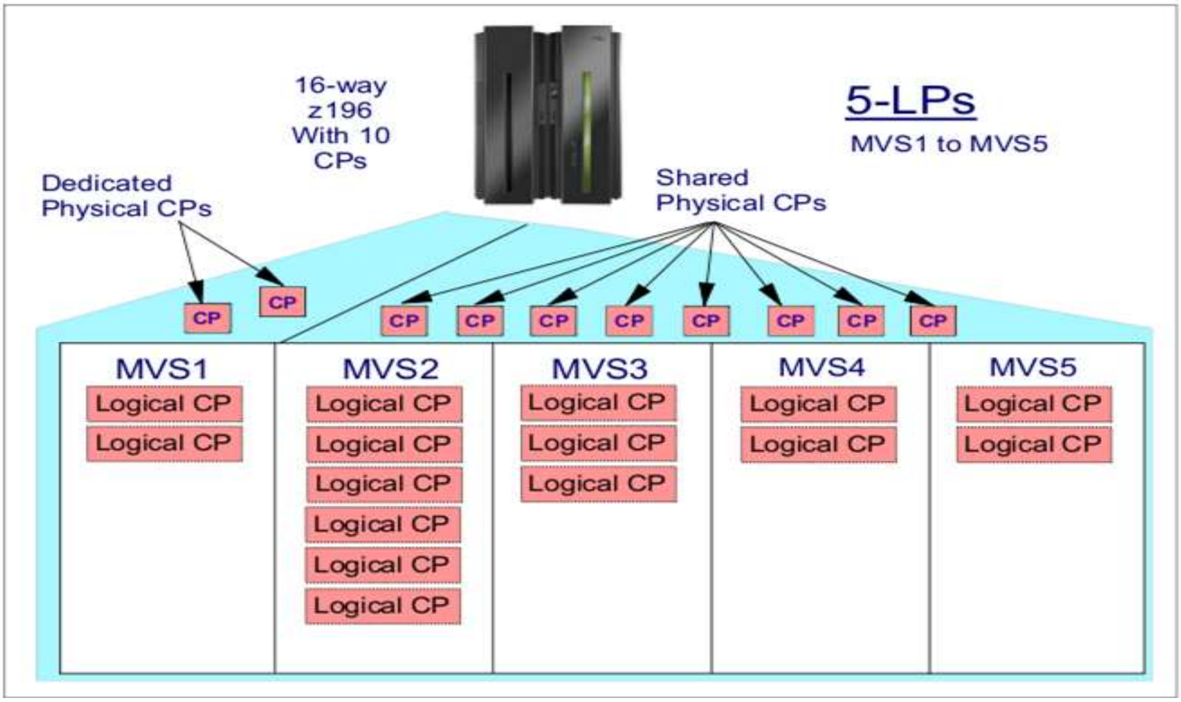


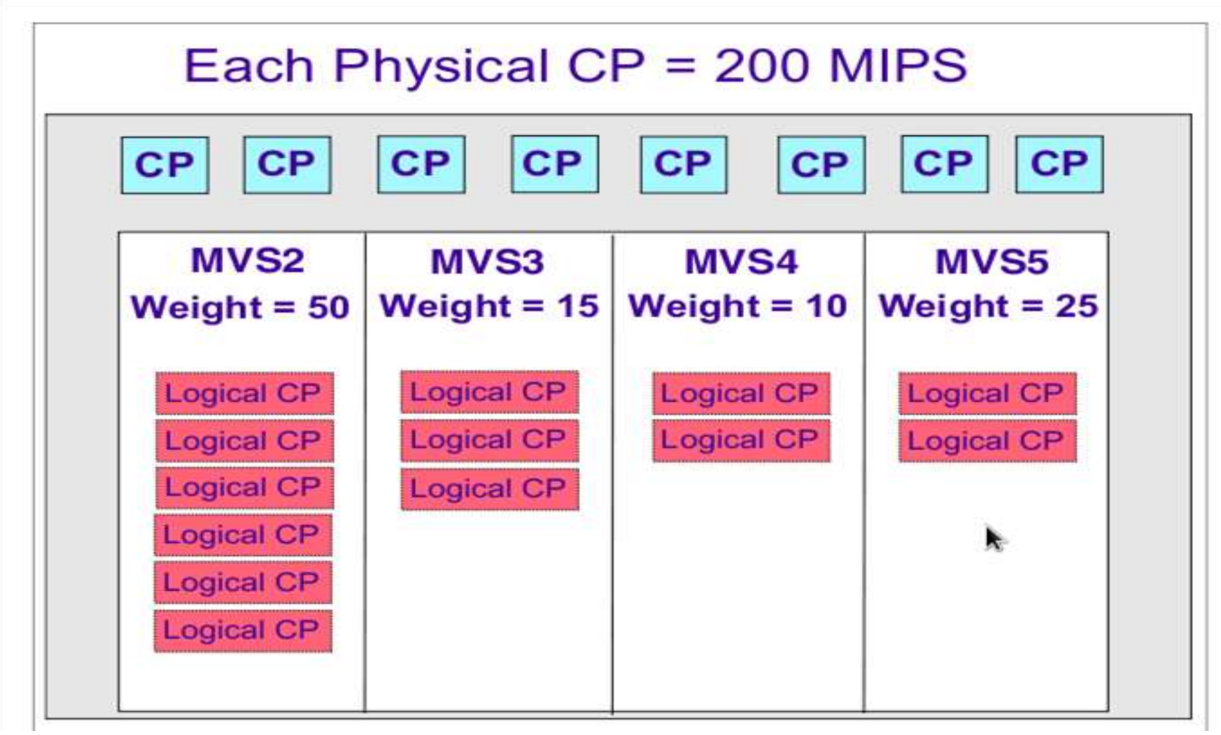
Data center example

- Multiple islands of computing resources.
- The typical data center consists of multiple heterogeneous platforms.
- Application platform selection should use a fit for purpose criteria so that it operates on the most cost effective platform.

LPAR slicing and weight – An example

In this example, the CEC has 10 physical processors. Two of it are dedicated to MVS1 LPAR. The rest 8 physical processors are shared among other 4 LPARs with an assignment of different weights. The following picture and the table will describe the resource allocation to different LPARs.





Each Physical CP = 200 MIPS

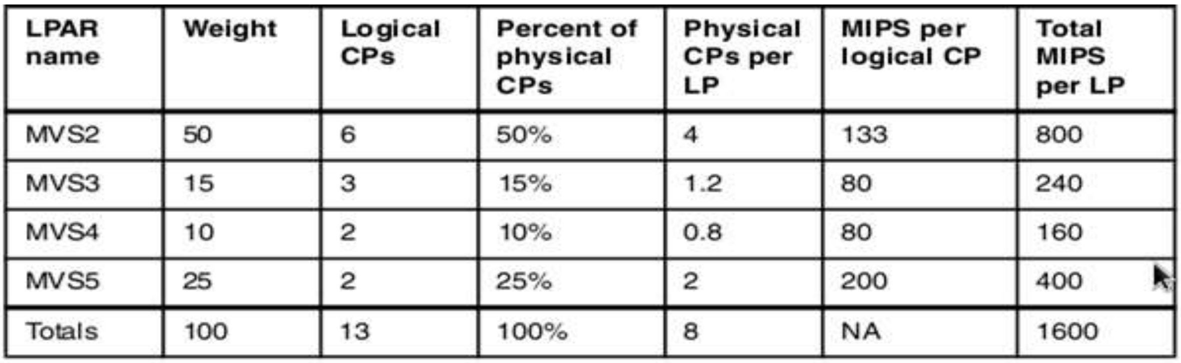
CP CP CP CP CP CP CP CP

MVS2
Weight = 50
Logical CP
Logical CP
Logical CP
Logical CP
Logical CP
Logical CP

MVS3
Weight = 15
Logical CP
Logical CP
Logical CP

MVS4
Weight = 10
Logical CP
Logical CP

MVS5
Weight = 25
Logical CP
Logical CP

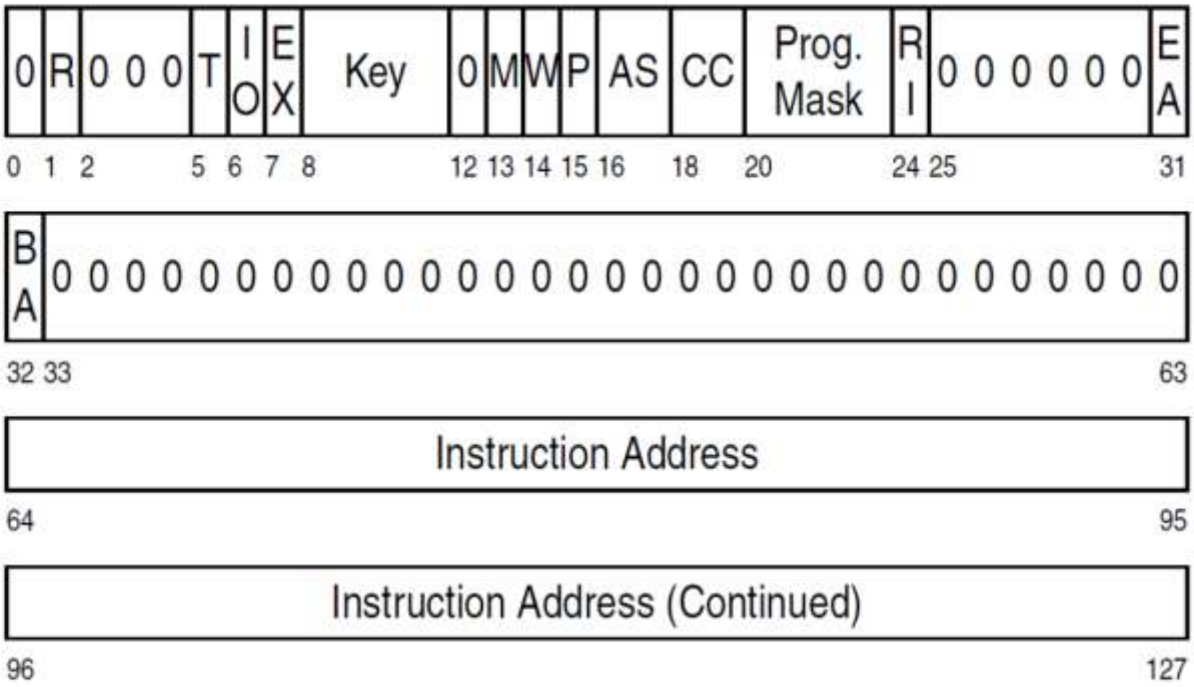


LPAR name	Weight	Logical CPs	Percent of physical CPs	Physical CPs per LP	MIPS per logical CP	Total MIPS per LP
MVS2	50	6	50%	4	133	800
MVS3	15	3	15%	1.2	80	240
MVS4	10	2	10%	0.8	80	160
MVS5	25	2	25%	2	200	400
Totals	100	13	100%	8	NA	1600

LPAR Capping

- LPAR capping limits the PU consumption in an LP through its weights (in HMC LPAR profile)
- Without capping an LP can receive more CP service than its weight allows:
  - When other LPs are not using all of their share
  - When an LP is deactivated
  - This is generally a good thing - you make the best use of the available capacity
- The intelligence and the capping executor are located in the LPAR hypervisor
- The limit of capping is also determined by the number of logical PUs in the LP
- A hard capping does not allow the LPAR to use beyond its allocated capacity even though free capacity is available from other shared LPARs in the CEC.

The PSW



Since the z/Architecture can run in a number of addressing modes, the instruction address is determined by a variable number of bits in the PSW. The current addressing mode is determined by bits 31-32 of the PSW with the following combinations:

00 → 24-bit mode

01 → 31-bit mode

10 → invalid

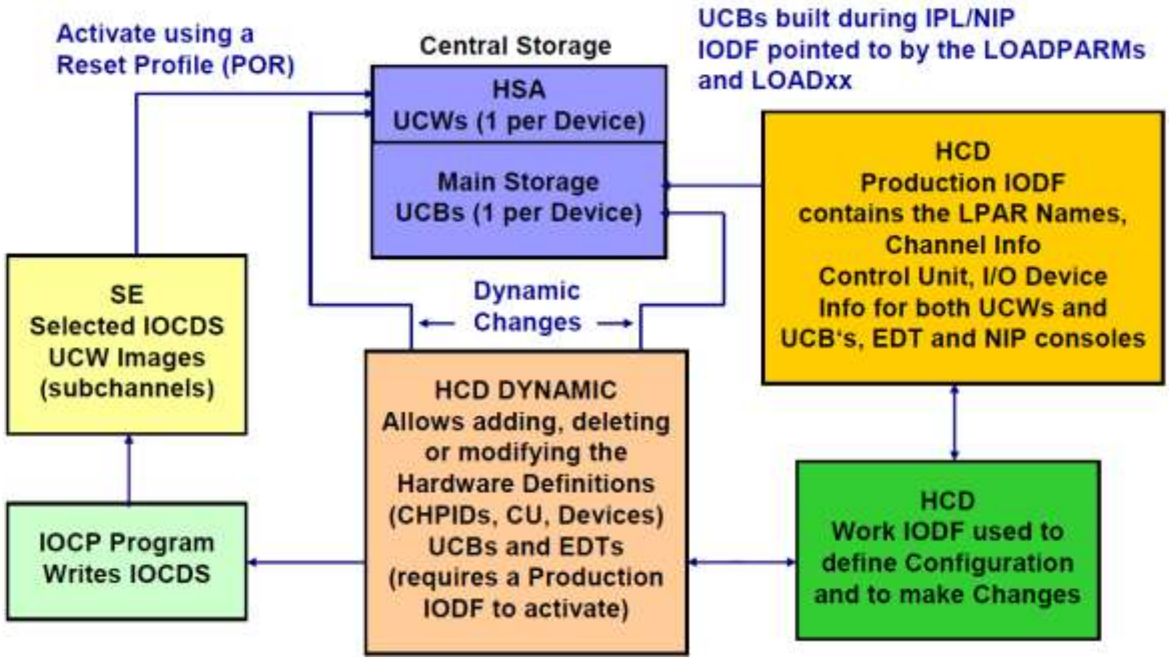
11 → 64-bit mode

UCB and UCW

A Unit Control Block (UCB) is a memory structure, or a *control block*, that describes any single input/output peripheral device (*unit*), or an *exposure* (alias), to the operating system. UCB is created at the time of system IPL (Initial Program Load) taking the necessary information from the IODF file.

A Unit Control Word (UCW) represents a device to the hardware. This is created at the time of POR (Power On Reset) of CEC and resides in the HSA (Hardware Storage Area) of the CEC.

The UCW represents the device from the channel subsystem point of view; an UCB represents the device from the IOS point of view.



Virtual Storage Management

In 1990, my mainframe journey started as a MVS systems programmer. I learnt it all by myself thru self reading (4 to 6 hours reading every day for 3+ months). I started with SRA (Science Research Associate) self study manuals which were fantastic readings. It took me days to months to understand the basic concept of virtual storage management under MVS and took years to get the clarity. It became more clearer to me after performing many problem investigations especially storage related issues. And got much better after I started reading the system dumps. Reading storage dumps (e.g. S878 abend) has been very interesting to me. Till today, it is still fascinating and interesting (sometimes beyond imagination) to know how zOS provides 16EB of storage to each address space in the system with an allocation of few GB or TB of physical central storage.

Explaining virtual storage management to my zOS colleagues has always been a challenge. But, it has been very difficult for me to explain zOS virtual storage management to SMEs in non-mainframe platforms.

Understanding virtual storage management is very complex as it demands to understand a whole lot of terminology and concepts. Just to list few e.g. physical storage, virtual storage, auxiliary storage, DAT, pages, frames, slots, segments, paging, swapping, line, bar, below & above the line/bar, common area, PSA, PLPA, SQA, CSA, private area, SWA, subpool, page table, segment table, region table, real storage manager, virtual storage manager, address space, data space and the list goes on.

But do not worry, start reading and if you proceed patiently and systematically, you will be there eventually.

If you are interested, I will suggest you to start understanding the address space layout well and then deep dive to different areas. You will get lot of good readings in ‘zOS MVS Initialization and Tuning Guide’, ‘z Architecture - Principles of Operation’, ‘zOS MVS Diagnosis - Reference’ manuals. And the information is spread everywhere over many manuals and it should not be difficult for you to find it as you need. Never forget the magic formula of 64 bit addressing (11 + 11 + 11 + 11 + 8 + 12) which is (R1 table + R2 table + R3 table + Segment table + Page table + Data page) which is (2048 + 2048 + 2048 + 2048 + 256 + 4096).

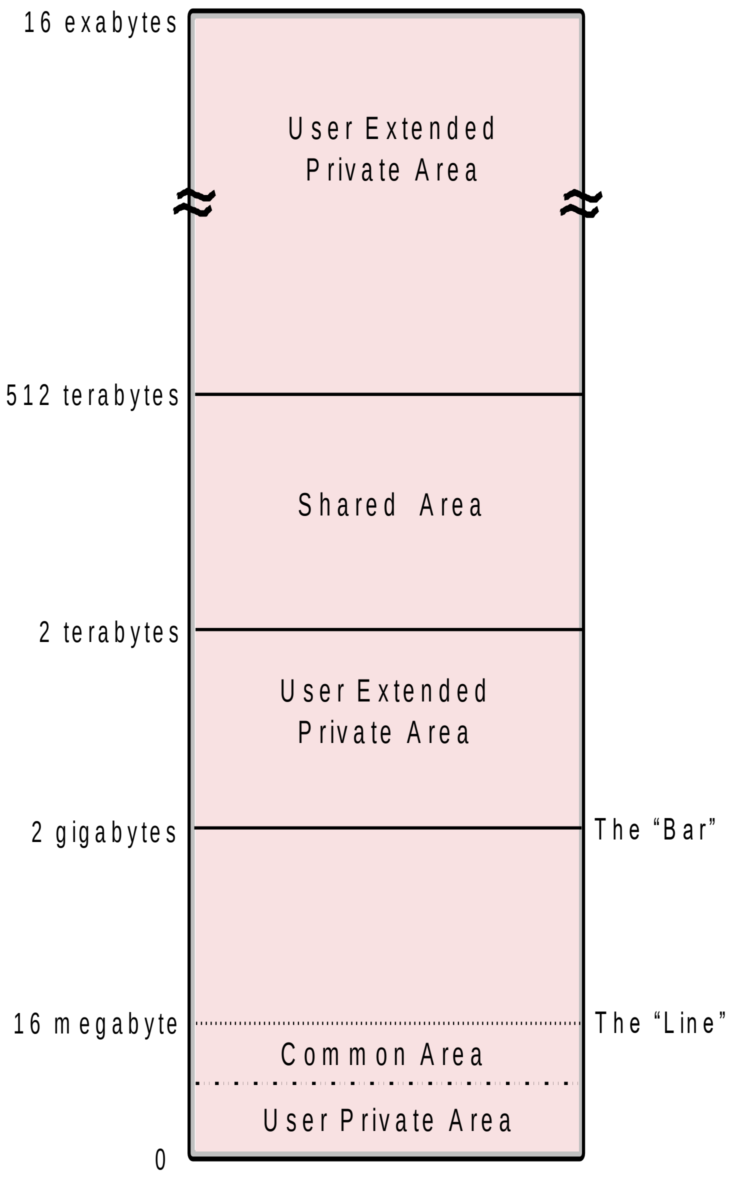
Please refer to this link for a nice reading on virtual storage management. https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&ved=2ahUKEwjX4rOin9iIAxVWWGwGHaKFB5kQFnoECBQQAQ&url=https%3A%2F%2Fshare.confex.com%2Fshare%2F115%2Fwebprogram%2FHandout%2FSession7511%2FBostonSHAREVSM.pdf&usg=AOvVaw0Heq_jX1_B74g7qBk7QV0b&opi=89978449

Proceed slowly. If you have storage related problems/issues at hand to resolve, this will enhance your understanding faster.

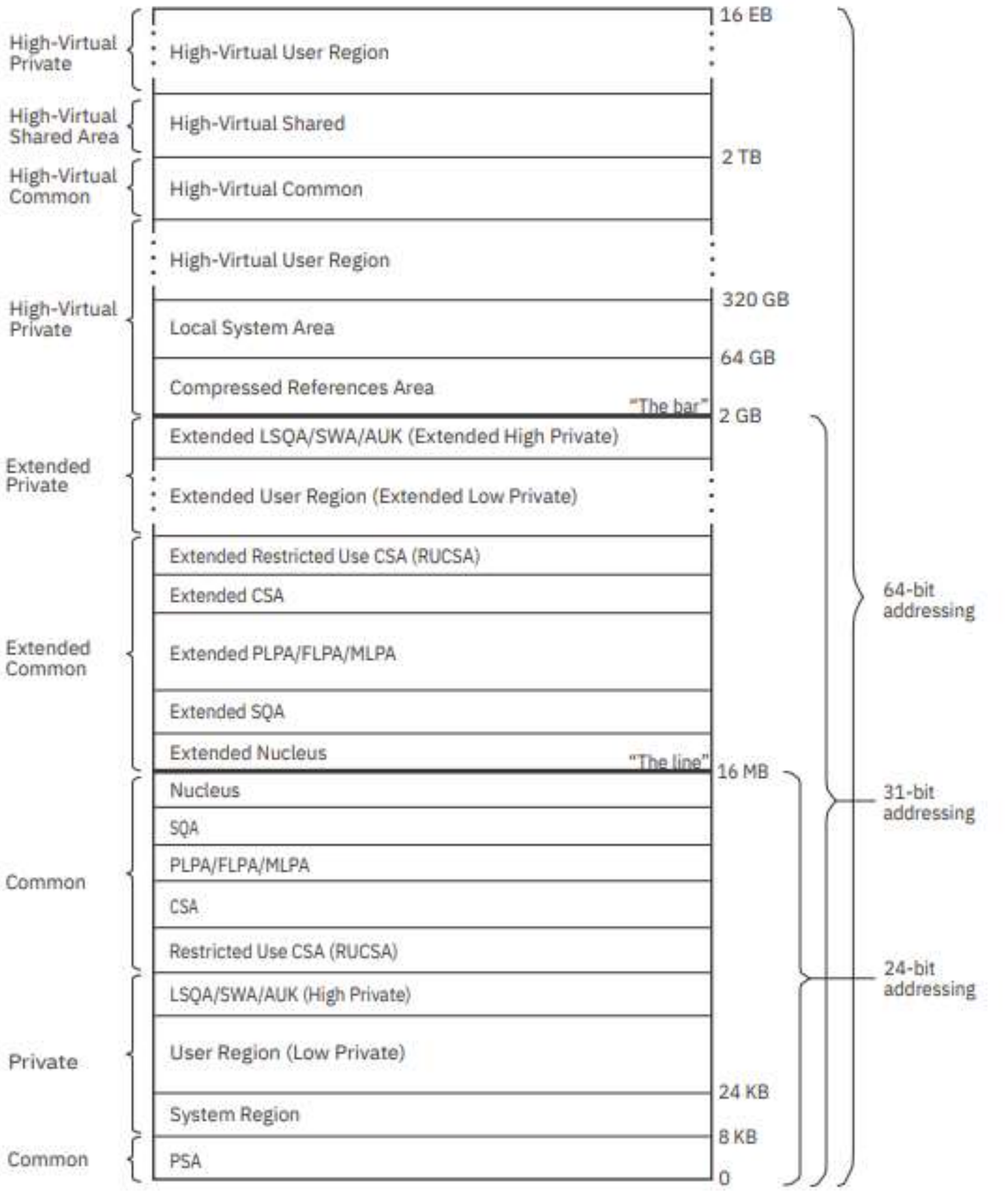
And at the end, please do not expect that someone can teach or explain you virtual storage management very well. Yes, someone can explain you the concept and provide some good details. But, it is you who must read, think, link and create a physical map in your mind to master your understanding. Then only this ‘Virtual’ will appear ’Real’ to you.

Here is a simple picture of the virtual storage layout of an address space.

Address Space Layout



And the expansion follows the in the following diagram.



Nucleus area

The nucleus area contains the nucleus load module and extensions to the nucleus that are initialized during IPL processing.

The nucleus includes a base and an architectural extension.

System Queue Area (SQA/ESQA)

This area contains tables and queues relating to the entire system. Its contents are highly dependent on configuration and job requirements at an installation. The total amount of virtual storage and number of private virtual storage address spaces are two of the factors that affect the system's use of SQA.

The SQA is allocated directly below the nucleus; the extended SQA is allocated directly above the extended nucleus.

The size of the SQA can be specified through the:

- SQA parameter in the **IEASYSxx** member of SYS1.PARMLIB
- NIP or operator's console.

If the specified amount of virtual storage is not available during initialization, a warning message will be issued. The SQA parameter may be respecified at that time from the operator's console.

Virtual SQA is allocated as a number of 64K blocks to be added to the minimum system requirements for SQA. If the SQA required by the system configuration exceeds the amount that has been reserved through the SQA parameter, the system attempts to allocate additional virtual SQA from the CSA area which is known as SQA overflow to CSA.

When certain storage thresholds are reached, the system stops creating new address spaces. When SQA is in use, it is fixed in central storage. The size of the SQA cannot be increased or decreased by the operator during a restart that reuses the previously initialized PLPA (a quick start). The size will be the same as during the preceding IPL.

Pageable Link Pack Area(PLPA/EPLPA)

This area contains SVC routines, access methods, and other read-only system programs along with any read-only reenterable user programs selected by an installation that can be shared among users of the system. Any module in the pageable link pack area will be treated by the system as though it came from an APF-authorized library. Ensure that you have properly protected SYS1.LPALIB and any library named in **LPALSTxx** or on an LPA statement in **PROGxx** to avoid system security and integrity exposures, just as you would protect any APF-authorized library.

It is desirable to place all frequently used refreshable SYS1.LINKLIB and SYS1.CMDLIB modules in the PLPA because of the following advantages:

- If possible, PLPA is backed by central storage above 16 megabytes; central storage below 16 megabytes is then available for other uses.
- The length of time that a page occupies central storage depends on its frequency of use. If the page is not used over a period of time, the system will reuse (steal) the central storage frame that the page occupies.
- The most frequently used PLPA modules in a time period will tend to remain in central storage.
- PLPA paged-in modules avoid program fetch overhead.
- Two or more programs that need the same PLPA module share the common PLPA code, thus reducing the demand for central storage.
- The main cost of unused PLPA modules is paging space, because only auxiliary storage is involved when modules are not being used.
- All modules in the PLPA are treated as refreshable, and are not paged-out. This action reduces the overall paging rate compared with modules in other libraries.

Modified Link Pack Area(MLPA/EMLPA)

This area may be used to contain reenterable routines from either APF-authorized or non-APF authorized libraries that are to be part of the pageable extension to the link pack area during the current IPL.

Any module in the modified link pack area will be treated by the system as though it came from an APF-authorized library. Ensure that you have properly protected any library named in IEALPAxx to avoid system security and integrity exposures, just as you would protect any APF-authorized library.

The MLPA exists only for the duration of an IPL. Therefore, if an MLPA is desired, the modules in the MLPA must be specified for each IPL (including quick start and warm start IPLs).

The MLPA is allocated just below the FLPA (or the PLPA, if there is no FLPA); the extended MLPA is allocated above the extended FLPA (or the extended PLPA if there is no extended FLPA). When the system searches for a routine, the MLPA is searched before the PLPA.

Note: Loading a large number of modules in the MLPA can increase fetch time for modules that are not loaded in the LPA. This could affect system performance.

The MLPA can be used at IPL time to temporarily modify or update the PLPA with new or replacement modules. No actual modification is made to the quick start PLPA stored in the system's paging data sets.

The MLPA is read-only, unless NOPROT is specified on the MLPA system parameter.

Specified by:

- Including a module list as an **IEALPAxx** member of SYS1.PARMLIB; where xx is the specific list.
- Including the MLPA system parameter in **IEASYSxx** or specifying MLPA from the operator's console during system initialization

Fixed Link Pack Area (FLPA/EFLPA)

An installation can elect to have some modules that are normally loaded in the pageable link pack area (PLPA) loaded into the fixed link pack area (FLPA). This area should be used only for modules that significantly increase performance when they are fixed rather than pageable. Modules placed in the FLPA must be reentrant and refreshable.

The FLPA exists only for the duration of an IPL. Therefore, if an FLPA is desired, the modules in the FLPA must be specified for each IPL (including quick-start and warm-start IPLs).

It is the responsibility of the installation to determine which modules, if any, to place in the FLPA. Note that if a module is heavily used and is in the PLPA, the system's paging algorithms will tend to keep that module in central storage. The best candidates for the FLPA are modules that are infrequently used but are needed for fast response to some terminal-oriented action.

Specified by: A list of modules to be put in FLPA must be established by the installation in the fixed LPA list (IEAFIXxx) member of SYS1.PARMLIB. Modules from any partitioned data set can be included in the FLPA. FLPA is selected through specification of the FIX system parameter in IEASYSxx or from the operator's console at system initialization.

Any module in the FLPA will be treated by the system as though it came from an APF-authorized library. Ensure that you have properly protected any library named in IEAFIXxx to avoid system security and integrity exposures, just as you would protect any APF-authorized library. This area may be used to contain reenterable routines from either APF-authorized or non-APF-authorized libraries that are to be part of the pageable extension to the link pack area during the current IPL.

Common Service Area(CSA/ECSA)

This area contains pageable and fixed data areas that are addressable by all active virtual storage address spaces. CSA normally contains data referenced by a number of system address spaces, enabling address spaces to communicate by referencing the same piece of CSA data.

CSA is allocated directly below the MLPA; extended CSA is allocated directly above the extended MLPA. If the virtual SQA space is depleted, the system will allocate additional SQA space from the CSA.

Specified by:

- The SYSP parameter at the operator's console to specify an alternative system parameter list (**IEASYSxx**) that contains a CSA specification.

- The CSA parameter at the operator's console during system initialization. This value overrides the

current system parameter value for CSA that was established by IEASYS00 or IEASYSxx.

Restricted use common service area (RUCSA/EUCSA)

IBM recommends the elimination of all user-key (8 - 15) common storage, as it creates a security risk because the storage can be modified or referenced, even if fetch protected, by any unauthorized program from any address space.

RUCSA is a separate area from CSA that is situated between the CSA and PVT areas. Because it is a separate area, it can be managed as a secure resource. The security administrator can define, via the System Authorization Facility (SAF), which users have access to the RUCSA. Only those with SAF READ authority to the IARRSM.RUCSA profile in the FACILITY class can have access. Once defined, all requests for user key CSA storage will transparently obtain storage from the RUCSA. Application changes might not be necessary.

RUCSA (and extended RUCSA) are similar to CSA (and extended CSA) in that their storage ranges reduce the size of the 24-bit and 31-bit private area ranges for all address spaces. However, RUCSA (and extended RUCSA) differ in the following ways:

- Only required by installations that have programs that require them.
- Exclusively used as a user-key CSA area.
- Accessible only from address spaces that are running under user IDs that have SAF READ authority to the IARRSM.RUCSA profile in the FACILITY class, or, on z/OS V2R3 or earlier systems that have the VSM ALLOWUSERKEYCSA(YES) parameter specified in the DIAGxx member of parmlib. To ensure that a job has a consistent SAF authority while it is running, the authorization is checked at job start and not altered until the job ends. As such, a never-ending job that requires SAF authorization after it starts must be restarted.
- Never converted to SQA or extended SQA.
- The RUCSA is allocated just below the CSA, and the extended RUCSA is allocated above the extended CSA. They are specified in the following ways:

- The SYSP parameter at the operator's console to specify an alternative system parameter list (IEASYSxx) that contains a RUCSA specification.
- The RUCSA parameter at the operator's console during system initialization. This value overrides the current system parameter value for RUCSA that was established in the IEASYSxx member.

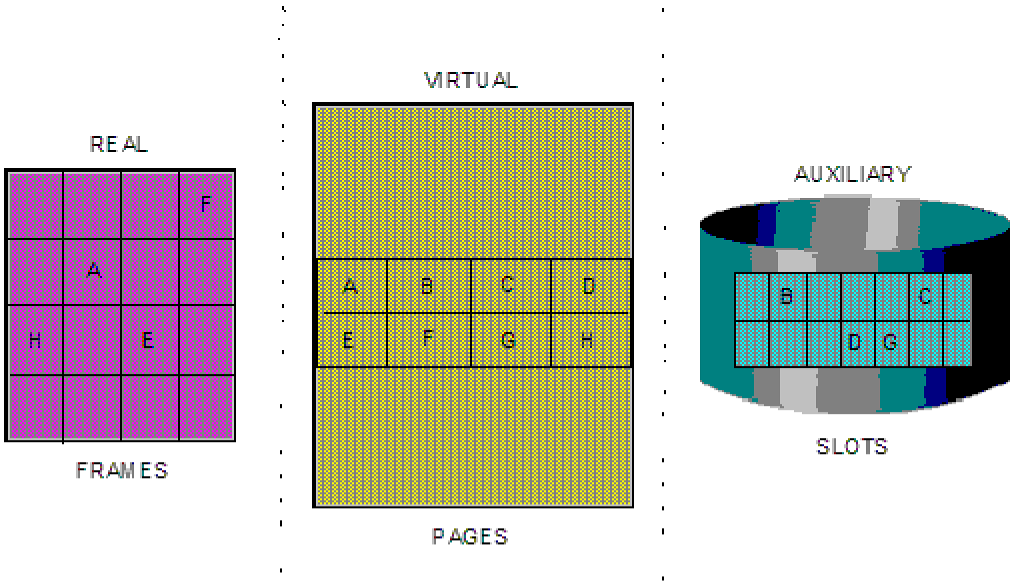
Local system queue area (LSQA/ELSQA)

Each virtual address space has an LSQA. The area contains tables and queues associated with the user's address space.

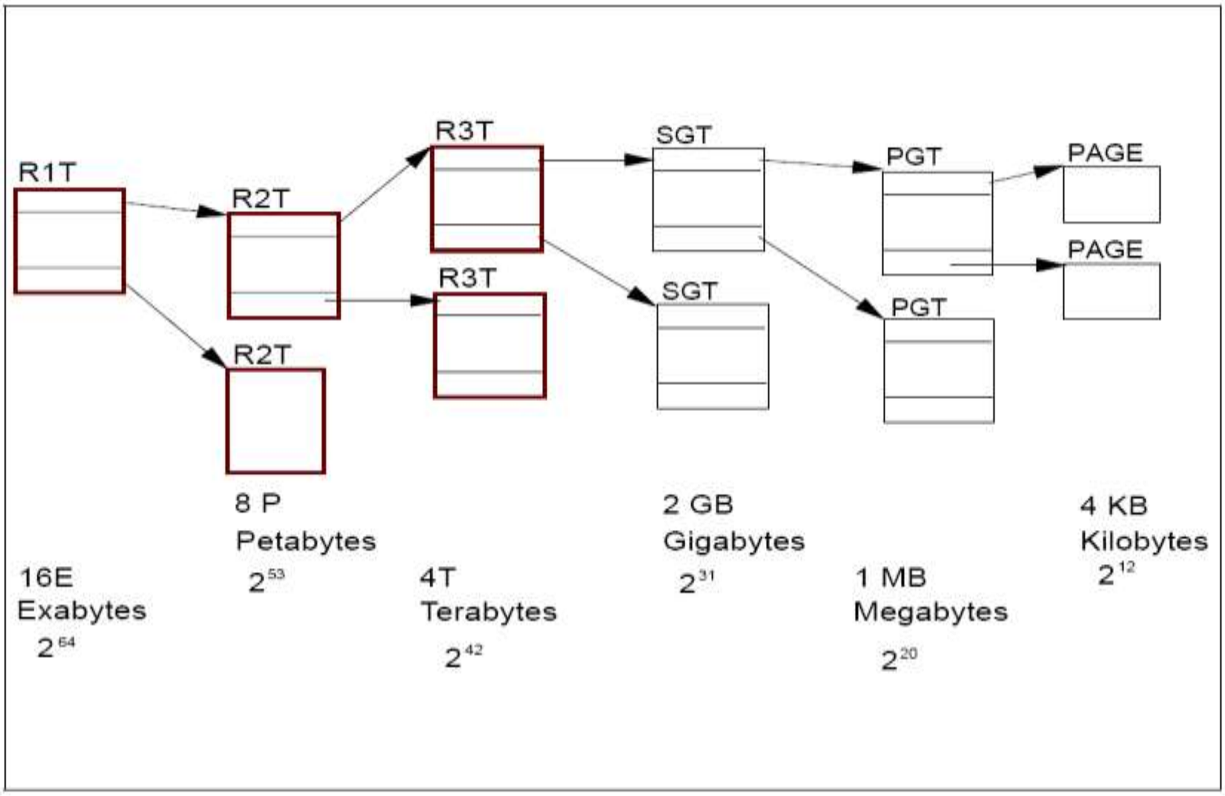
LSQA is intermixed with SWA and subpools 229, 230, and 249 downward from the bottom of the CSA into the unallocated portion of the private area, as needed. Extended LSQA is intermixed with SWA and subpools 229, 230, and 249 downward from 2 gigabytes into the unallocated portion of the extended private area, as needed. LSQA will not be taken from space below the top of the highest storage currently allocated to the private area user region. Any job will abnormally terminate unless there is enough space for allocating LSQA.

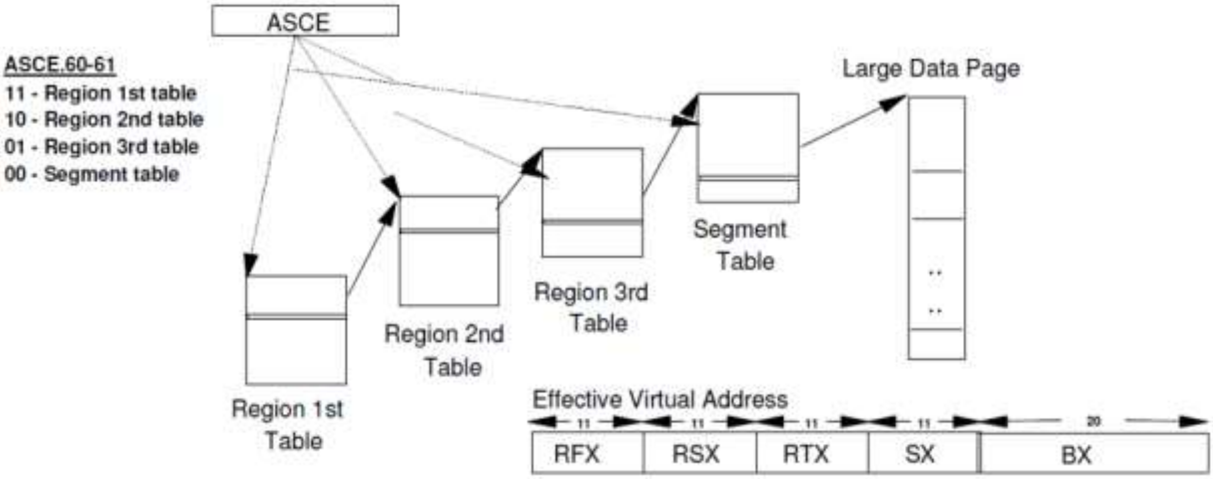
Storage terms - Pages, Frames, and Slots

- The pieces of a program executing in virtual storage must be moved between real and auxiliary storage:
  - A block of real storage is a *frame*.
  - A block of virtual storage is a *page*.
  - A block of auxiliary storage is a *slot*.
  - A page, a frame, and a slot are all the same size: 4096 bytes (4 kilobytes).
- To the programmer, the entire program appears to occupy contiguous spaces in real storage at all times.


REAL
F
A
H
E
FRAMES
VIRTUAL
A
B
C
D
E
F
G
H
PAGES
AUXILIARY
B
C
D
G
SLOTS

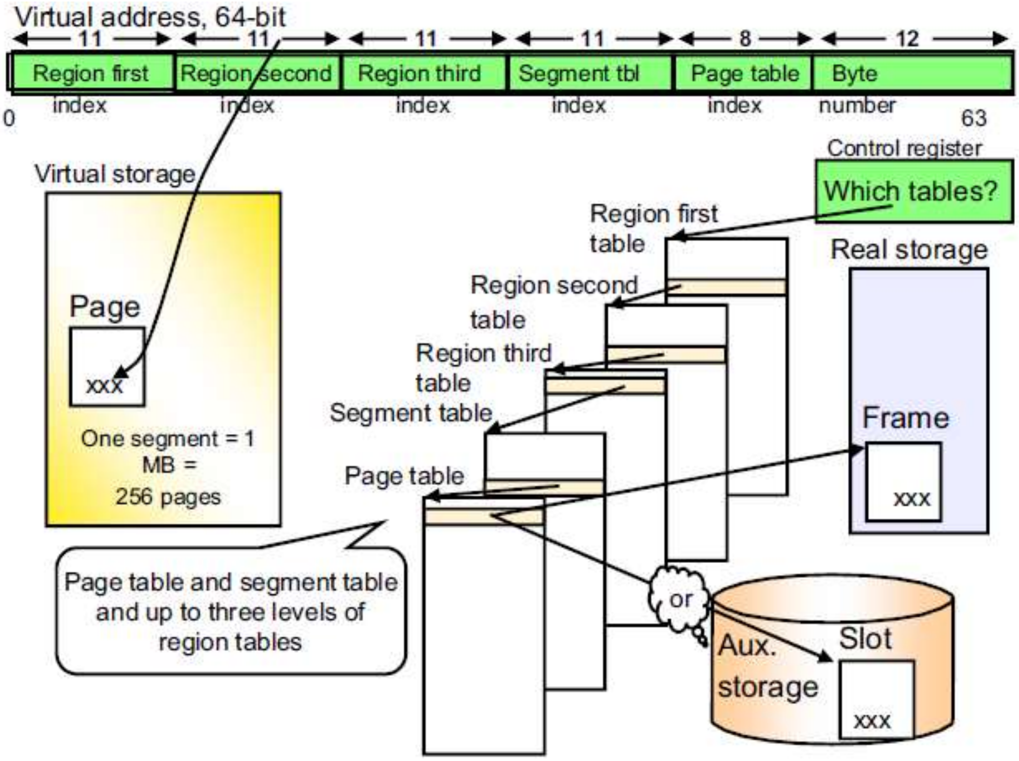
64-bit address layout





- Translation can start at R1T, R2T, R3T or SGT
- The starting point of the translation is designated in the Address Space Control Element (ASCE.60-61)

Dynamic Address translation (DAT)



In z/OS, a virtual address can be 24-, 31-, or 64-bit. This virtual address is found in the PSW (it is an instruction address) or a register (it is a data address). The DAT-box in the CPU translates the virtual storage address to the real address. This is done each time an instruction is executed in the CPU.

Up to five levels of translation tables are used in z/OS for 64-bit addressing. Some part of the address is an index to the first table where the address of the second table is found. Some other part of the address is an index to this second table, from where the third table is found. And so on. The last table, the page table, contains the address of the real storage frame (4 KB block), or a reference to slot (4 KB block) on page data set, if data is paged out.

If data is paged out, the execution of the instruction is interrupted and the operating system will do a page-in of the page. The instruction is then later re-executed.

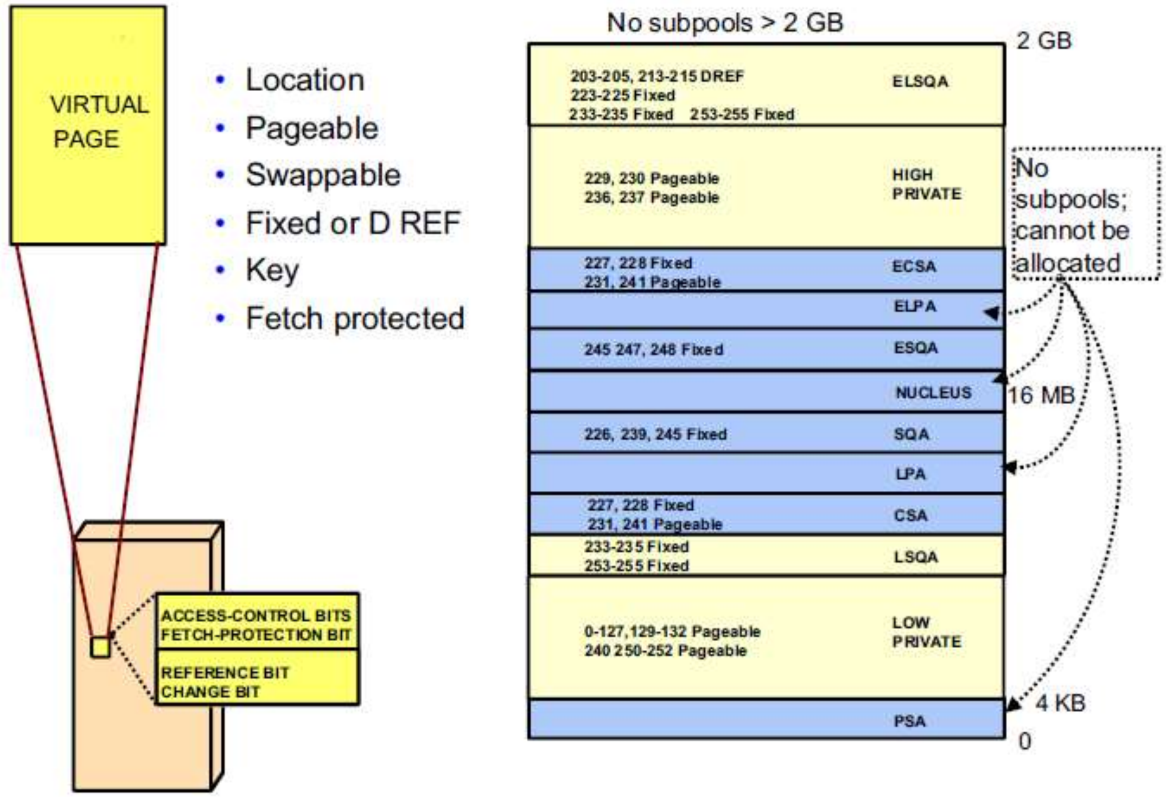
Storage Subpools

Each frame of real storage will be given virtual storage characteristics when allocated to a program. Some of these characteristics are:

- **Pageable**: Can the page in real storage be paged out?
- **Swappable**: Can the page in real storage be swapped out?
- **DREF** (disabled reference): DREF storage cannot be paged to/from page data set, but can be paged to/from expanded storage. Such a page operation does not involve I/O and can be done in disabled state. A supervisor routine can run in disabled state, which means no I/O interrupts are allowed. If no I/O interrupts are allowed, no normal paging can be done.
- **Fixed**: The page in real storage will not be paged out. Fixed storage is mostly used by supervisor routines, which can run disabled, and therefore cannot tolerate page faults.
- **Fetch Protected**: Do you need matching PSW key (or zero) to read this area in storage?
- **Location**: Will the page be located above or below the 16 MB virtual address line and the 16 MB real address line?

When allocating storage, a subpool is specified, and thus the characteristics of storage.

Subpools are only related to storage, which can be allocated (by **GETMAIN** or **STORAGE OBTAIN** macros), so subpools do not apply for PSA, Nucleus, LPA.

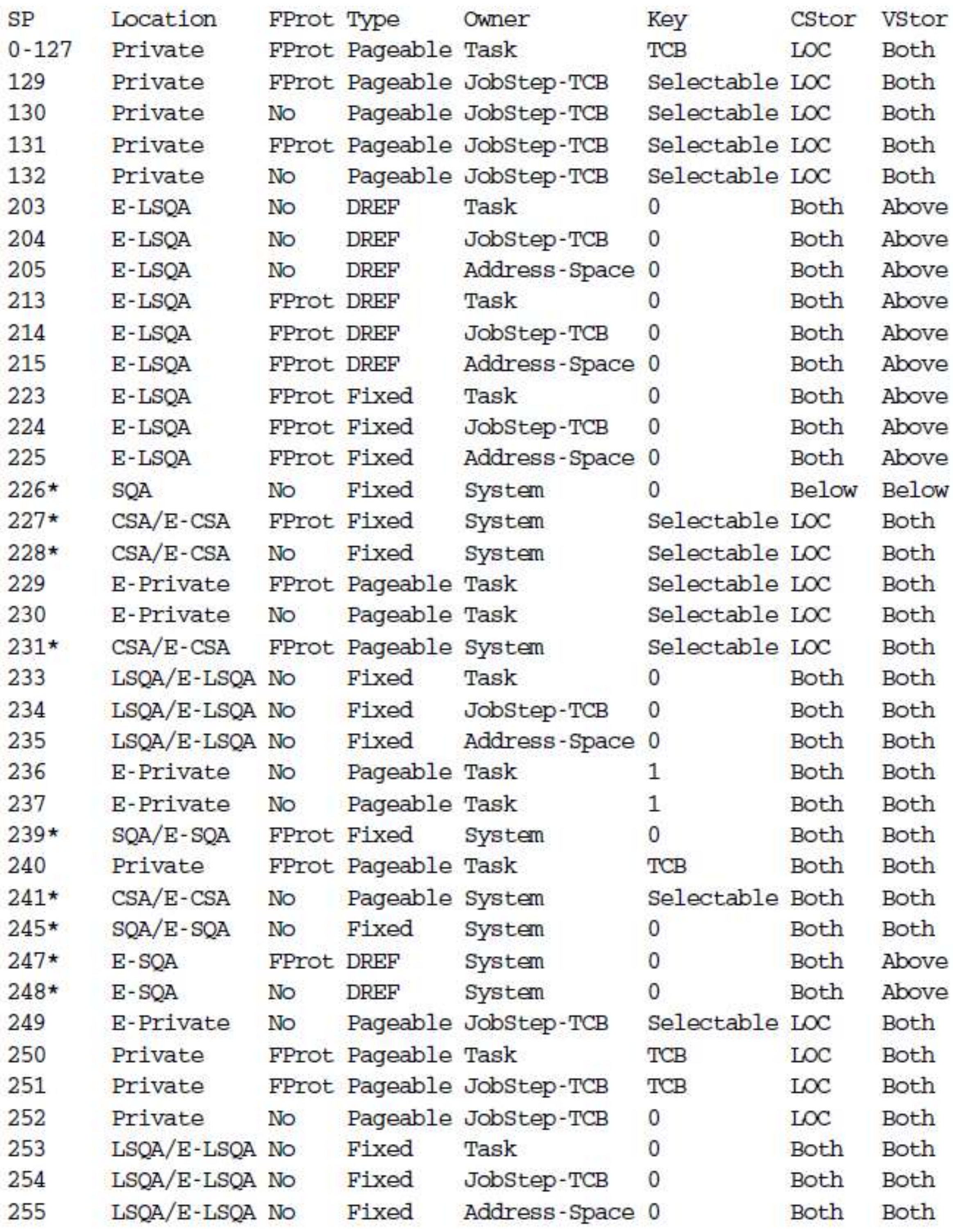


- Location
- Pageable
- Swappable
- Fixed or D REF
- Key
- Fetch protected

No subpools > 2 GB

203-205, 213-215 DREF 223-225 Fixed 233-235 Fixed 253-255 Fixed	ELSQA	2 GB
229, 230 Pageable 236, 237 Pageable	HIGH PRIVATE	
227, 228 Fixed 231, 241 Pageable	ECSA	
	ELPA	
245 247, 248 Fixed	ESQA	
	NUCLEUS	16 MB
226, 239, 245 Fixed	SQA	
	LPA	
227, 228 Fixed 231, 241 Pageable	CSA	
233-235 Fixed 253-255 Fixed	LSQA	
0-127,129-132 Pageable 240 250-252 Pageable	LOW PRIVATE	4 KB
	PSA	0

No subpools; cannot be allocated



SP	Location	FProt	Type	Owner	Key	CStor	VStor
0-127	Private	FProt	Pageable	Task	TCB	LOC	Both
129	Private	FProt	Pageable	JobStep-TCB	Selectable	LOC	Both
130	Private	No	Pageable	JobStep-TCB	Selectable	LOC	Both
131	Private	FProt	Pageable	JobStep-TCB	Selectable	LOC	Both
132	Private	No	Pageable	JobStep-TCB	Selectable	LOC	Both
203	E-LSQA	No	DREF	Task	0	Both	Above
204	E-LSQA	No	DREF	JobStep-TCB	0	Both	Above
205	E-LSQA	No	DREF	Address-Space	0	Both	Above
213	E-LSQA	FProt	DREF	Task	0	Both	Above
214	E-LSQA	FProt	DREF	JobStep-TCB	0	Both	Above
215	E-LSQA	FProt	DREF	Address-Space	0	Both	Above
223	E-LSQA	FProt	Fixed	Task	0	Both	Above
224	E-LSQA	FProt	Fixed	JobStep-TCB	0	Both	Above
225	E-LSQA	FProt	Fixed	Address-Space	0	Both	Above
226*	SQA	No	Fixed	System	0	Below	Below
227*	CSA/E-CSA	FProt	Fixed	System	Selectable	LOC	Both
228*	CSA/E-CSA	No	Fixed	System	Selectable	LOC	Both
229	E-Private	FProt	Pageable	Task	Selectable	LOC	Both
230	E-Private	No	Pageable	Task	Selectable	LOC	Both
231*	CSA/E-CSA	FProt	Pageable	System	Selectable	LOC	Both
233	LSQA/E-LSQA	No	Fixed	Task	0	Both	Both
234	LSQA/E-LSQA	No	Fixed	JobStep-TCB	0	Both	Both
235	LSQA/E-LSQA	No	Fixed	Address-Space	0	Both	Both
236	E-Private	No	Pageable	Task	1	Both	Both
237	E-Private	No	Pageable	Task	1	Both	Both
239*	SQA/E-SQA	FProt	Fixed	System	0	Both	Both
240	Private	FProt	Pageable	Task	TCB	Both	Both
241*	CSA/E-CSA	No	Pageable	System	Selectable	Both	Both
245*	SQA/E-SQA	No	Fixed	System	0	Both	Both
247*	E-SQA	FProt	DREF	System	0	Both	Above
248*	E-SQA	No	DREF	System	0	Both	Above
249	E-Private	No	Pageable	JobStep-TCB	Selectable	LOC	Both
250	Private	FProt	Pageable	Task	TCB	LOC	Both
251	Private	FProt	Pageable	JobStep-TCB	TCB	LOC	Both
252	Private	No	Pageable	JobStep-TCB	0	LOC	Both
253	LSQA/E-LSQA	No	Fixed	Task	0	Both	Both
254	LSQA/E-LSQA	No	Fixed	JobStep-TCB	0	Both	Both
255	LSQA/E-LSQA	No	Fixed	Address-Space	0	Both	Both

Search order of programs

When a program is requested through a system service (like LINK, LOAD, XCTL, or ATTACH) using default options, the system searches for it in the following sequence:

1. Job pack area (JPA)

A program in JPA has already been loaded in the requesting address space. If the copy in JPA can be used, it will be used. Otherwise, the system either searches for a new copy or defers the request until the copy in JPA becomes available.

2. TASKLIB

A program can allocate one or more data sets to a TASKLIB concatenation. Data sets concatenated to TASKLIB are searched for after JPA but before any specified STEPLIB or JOBLIB.

3. STEPLIB or JOBLIB

STEPLIB and JOBLIB are specific DD names that can be used to allocate data sets to be searched ahead of the default system search order for programs. Data sets can be allocated to both the STEPLIB and JOBLIB concatenations in JCL or by a program using dynamic allocation. However, only one or the other will be searched for modules. If both STEPLIB and JOBLIB are allocated, for a particular job step, the system searches STEPLIB and ignores JOBLIB.

4. LPA, which is searched in this order:

- Dynamic LPA modules, as specified in PROGxx members
- Fixed LPA (FLPA) modules, as specified in IEAFIXxx members
- Modified LPA (MLPA) modules, as specified in IEALPAxx members
- Pageable LPA (PLPA) modules, loaded from libraries specified in LPALSTxx or PROGxx LPA modules are loaded in common storage, shared by all address spaces in the system. Because these modules are reentrant and are not self-modifying, each can be used by any number of tasks in any number of address spaces at the same time.

5. Libraries in the linklist, as specified in PROGxx and LNKLSTxx.

By default, the linklist begins with SYS1.LINKLIB, SYS1.MIGLIB, SYS1.CSSLIB, SYS1.SIEALNKE, and SYS1.SIEAMIGE. However, you can change this order using SYSLIB in PROGxx and add other libraries to the linklist concatenation. The system must bring modules found in the linklist into private area virtual storage before the programs can run.

Search order of procedures

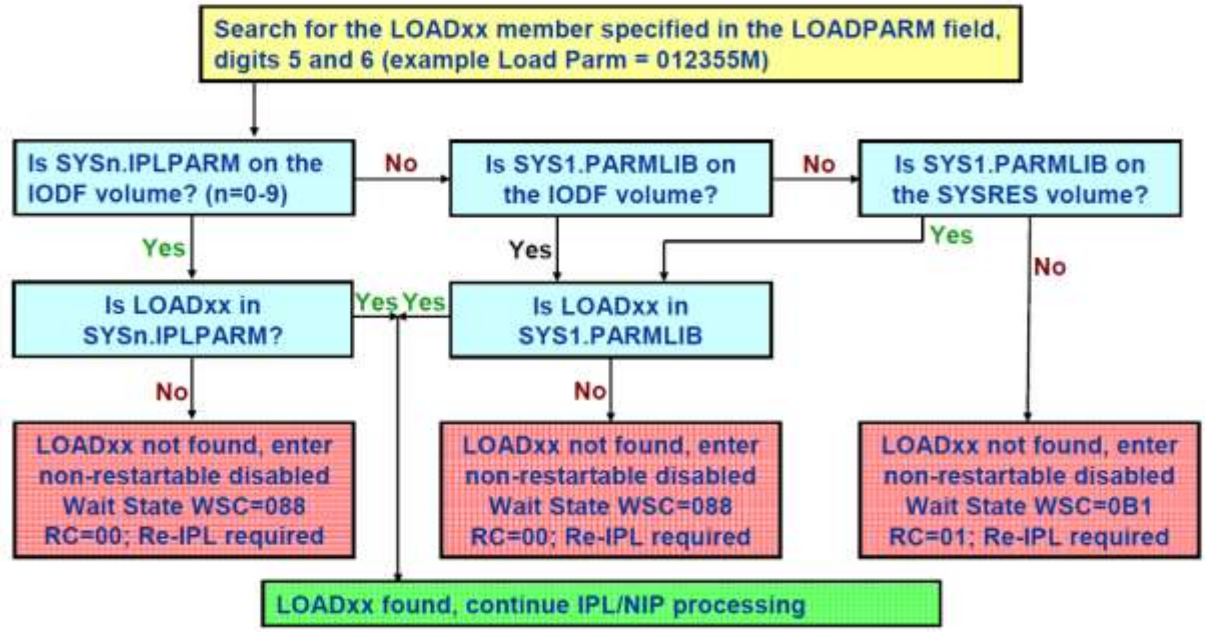
This is an example of invoking a procedure PROC1 for execution.

```
//JOB1JOBMSGCLASS=A

//UPLIBJCLLIBORDER=(USER.PROCLIB,TEST.JCL)

//STP1EXECPROC=PROC1
```

The system searches for PROC1 in these libraries (known as PROCLIBs) in the following order:

- Libraries listed on the JCLLIB statement (in their order of listing)
- Installation-defined proclibs
- SYS1.PROCLIB

Search order of LOADxx



Search for the LOADxx member specified in the LOADPARM field, digits 5 and 6 (example Load Parm = 012355M)

Is SYSn.IPLPARM on the IODF volume? (n=0-9)

Is SYS1.PARMLIB on the IODF volume?

Is SYS1.PARMLIB on the SYSRES volume?

Is LOADxx in SYSn.IPLPARM?

Is LOADxx in SYS1.PARMLIB

LOADxx not found, enter non-restartable disabled Wait State WSC=088 RC=00; Re-IPL required

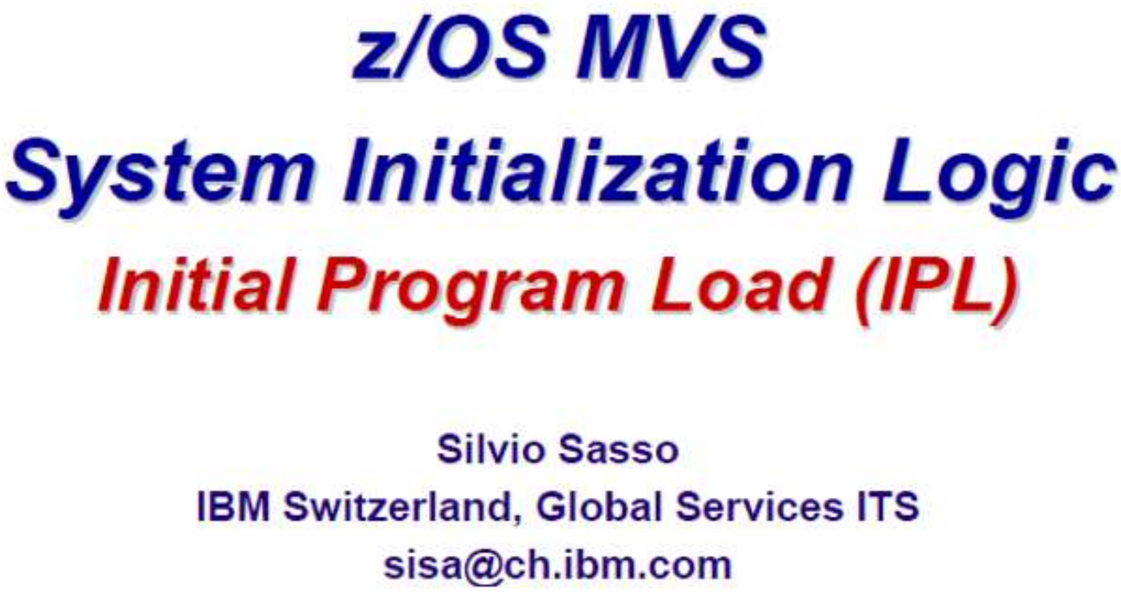
LOADxx not found, enter non-restartable disabled Wait State WSC=088 RC=00; Re-IPL required

LOADxx not found, enter non-restartable disabled Wait State WSC=0B1 RC=01; Re-IPL required

LOADxx found, continue IPL/NIP processing

IPL (Initial Program Load)

This information has been extracted from the following presentation in 2014. You can download the PDF from internet.



z/OS MVS

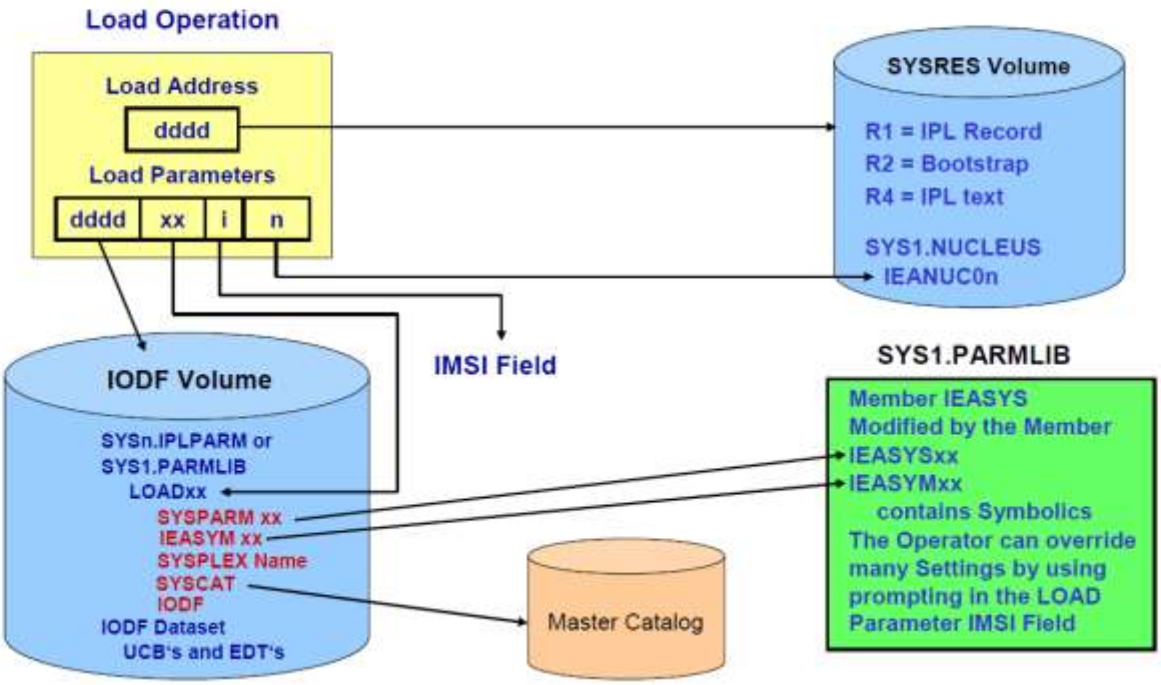
System Initialization Logic

Initial Program Load (IPL)

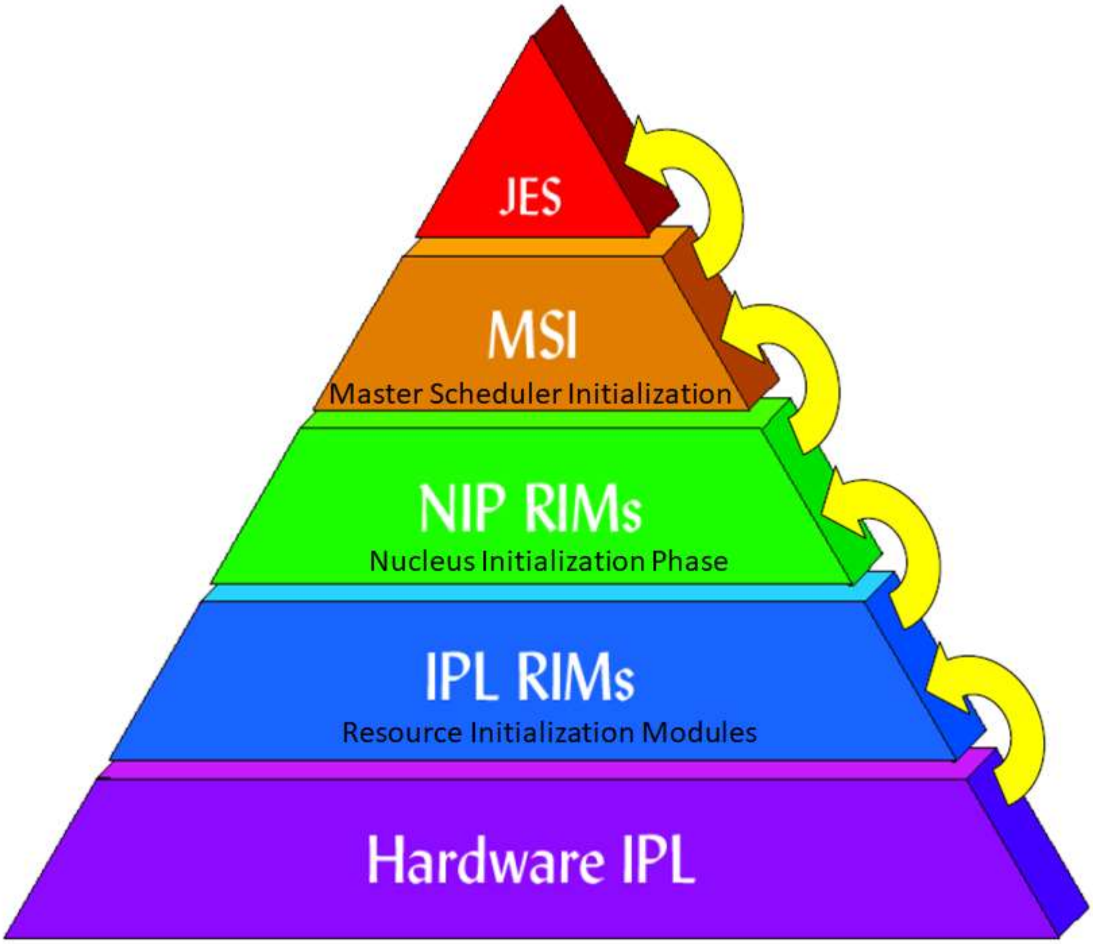
Silvio Sasso
IBM Switzerland, Global Services ITS
sisa@ch.ibm.com

Start the IPL process

The operator or systems programmer IPL the zOS system by providing the Load Address and Load Parameters.

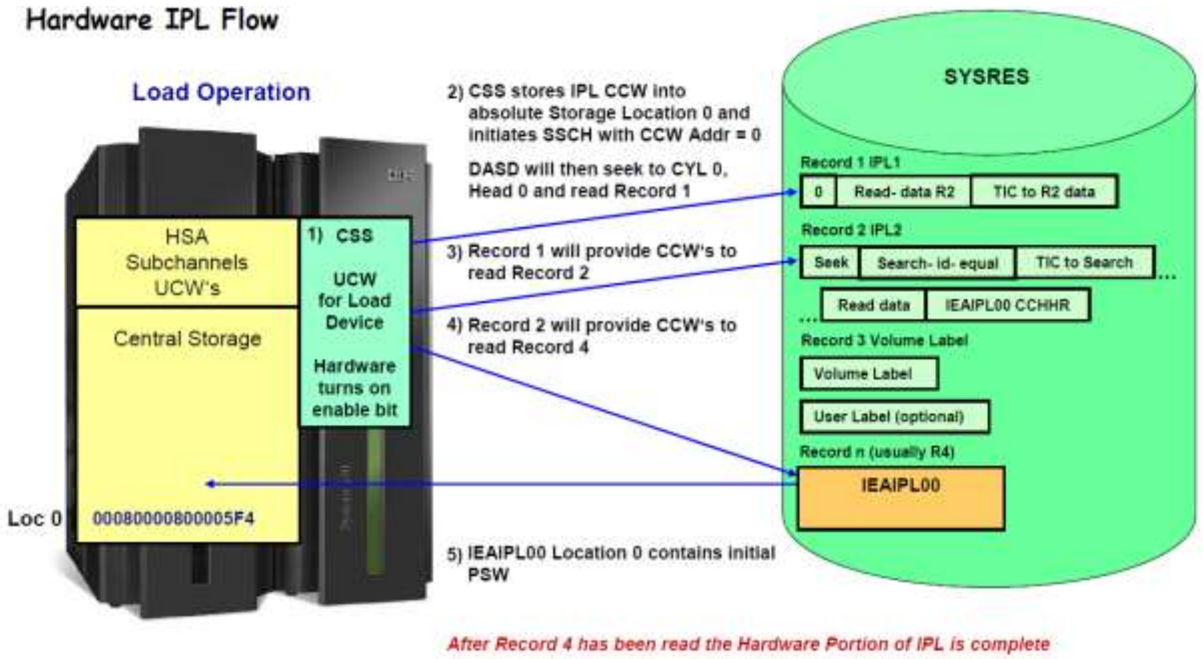


IPL flow


JES
MSI
Master Scheduler Initialization
NIP RIMs
Nucleus Initialization Phase
IPL RIMs
Resource Initialization Modules
Hardware IPL

Hardware IPL

This is the first phase in the IPL process.



Hardware IPL Flow

1) CSS

2) CSS stores IPL CCW into absolute Storage Location 0 and initiates SSCH with CCW Addr = 0

DASD will then seek to CYL 0, Head 0 and read Record 1

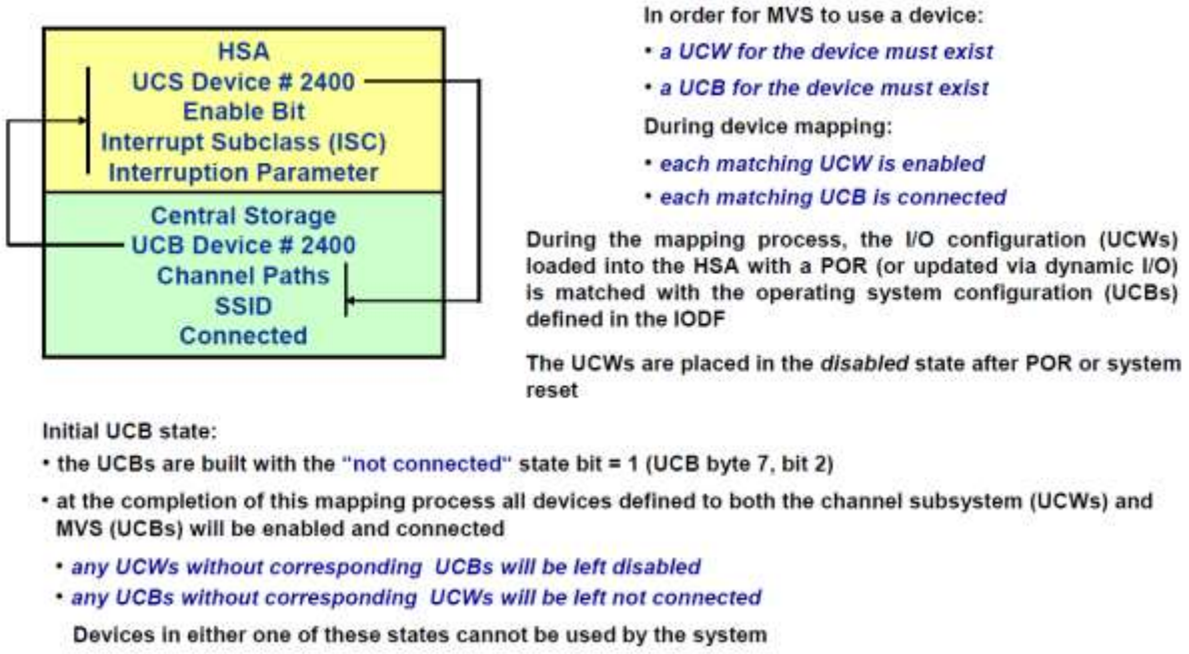
3) Record 1 will provide CCW's to read Record 2

4) Record 2 will provide CCW's to read Record 4

5) IEAIPL00 Location 0 contains initial PSW

After Record 4 has been read the Hardware Portion of IPL is complete

UCW and UCB mapping



HSA
UCS Device # 2400
Enable Bit
Interrupt Subclass (ISC)
Interruption Parameter

Central Storage
UCB Device # 2400
Channel Paths
SSID
Connected

In order for MVS to use a device:

- *a UCW for the device must exist*
- *a UCB for the device must exist*

During device mapping:

- *each matching UCW is enabled*
- *each matching UCB is connected*

During the mapping process, the I/O configuration (UCWs) loaded into the HSA with a POR (or updated via dynamic I/O) is matched with the operating system configuration (UCBs) defined in the IODF

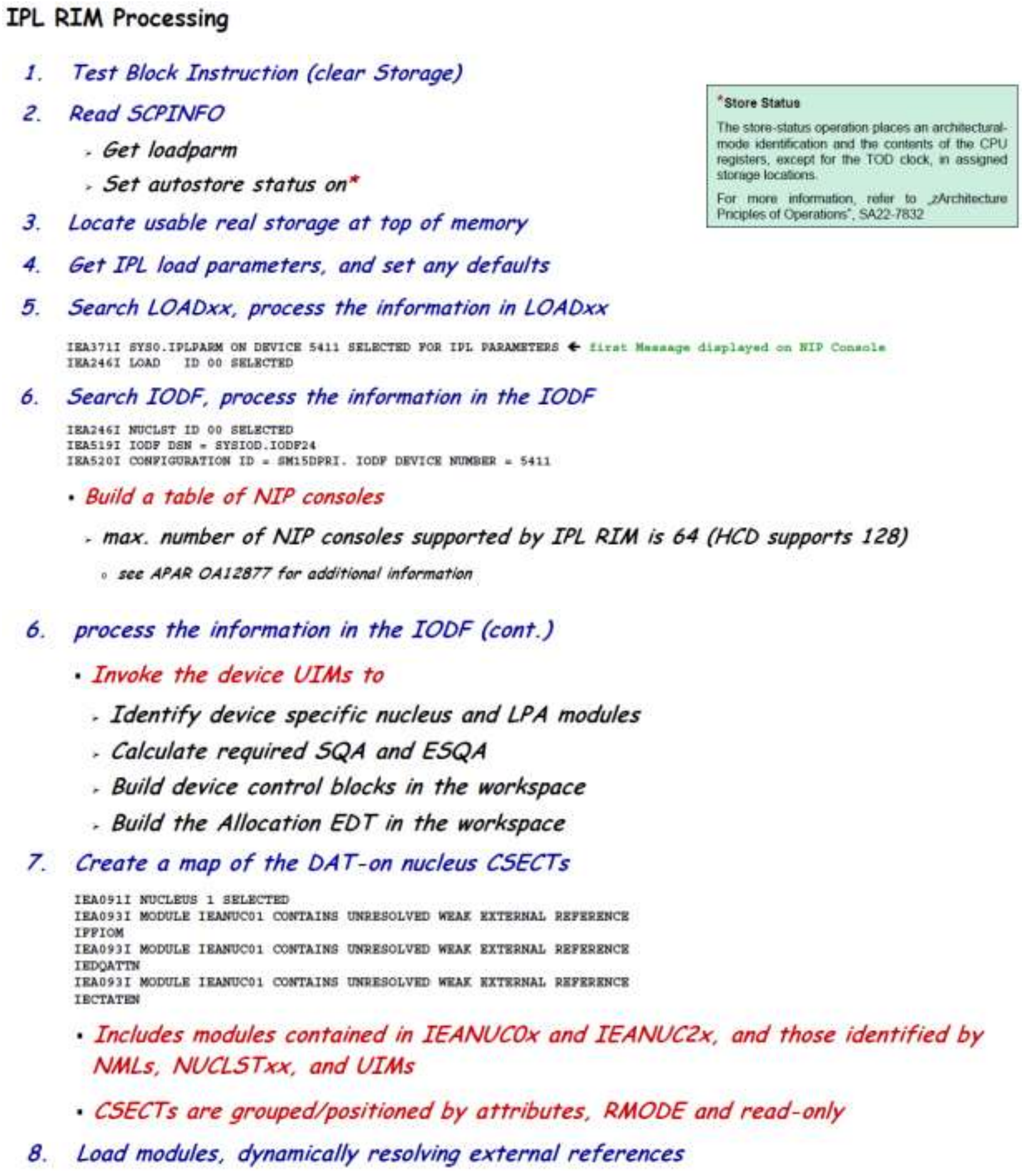
The UCWs are placed in the *disabled* state after POR or system reset

Initial UCB state:

- the UCBs are built with the "not connected" state bit = 1 (UCB byte 7, bit 2)
- at the completion of this mapping process all devices defined to both the channel subsystem (UCWs) and MVS (UCBs) will be enabled and connected
  - *any UCWs without corresponding UCBs will be left disabled*
  - *any UCBs without corresponding UCWs will be left not connected*

  Devices in either one of these states cannot be used by the system

IPL RIMs



IPL RIM Processing

1. *Test Block Instruction (clear Storage)*
2. *Read SCPINFO*
   - *Get loadparm*
   - *Set autostore status on**
3. *Locate usable real storage at top of memory*
4. *Get IPL load parameters, and set any defaults*
5. *Search LOADxx, process the information in LOADxx*

```
IEA371I SYS0.IPLPARM ON DEVICE 5411 SELECTED FOR IPL PARAMETERS ← first Message displayed on NIP Console
IEA246I LOAD   ID 00 SELECTED
```

6. *Search IODF, process the information in the IODF*

```
IEA246I NUCLST ID 00 SELECTED
IEA519I IODF DSN = SYSIOD.IODF24
IEA520I CONFIGURATION ID = SM15DPRI. IODF DEVICE NUMBER = 5411
```

- *Build a table of NIP consoles*
  - *max. number of NIP consoles supported by IPL RIM is 64 (HCD supports 128)*
    - *see APAR OA12877 for additional information*

6. *process the information in the IODF (cont.)*

- *Invoke the device UIMs to*
  - *Identify device specific nucleus and LPA modules*
  - *Calculate required SQA and ESQA*
  - *Build device control blocks in the workspace*
  - *Build the Allocation EDT in the workspace*

7. *Create a map of the DAT-on nucleus CSECTs*

```
IEA091I NUCLEUS 1 SELECTED
IEA093I MODULE IEANUC01 CONTAINS UNRESOLVED WEAK EXTERNAL REFERENCE
IPFIOM
IEA093I MODULE IEANUC01 CONTAINS UNRESOLVED WEAK EXTERNAL REFERENCE
IEDQATTN
IEA093I MODULE IEANUC01 CONTAINS UNRESOLVED WEAK EXTERNAL REFERENCE
IECTATEN
```

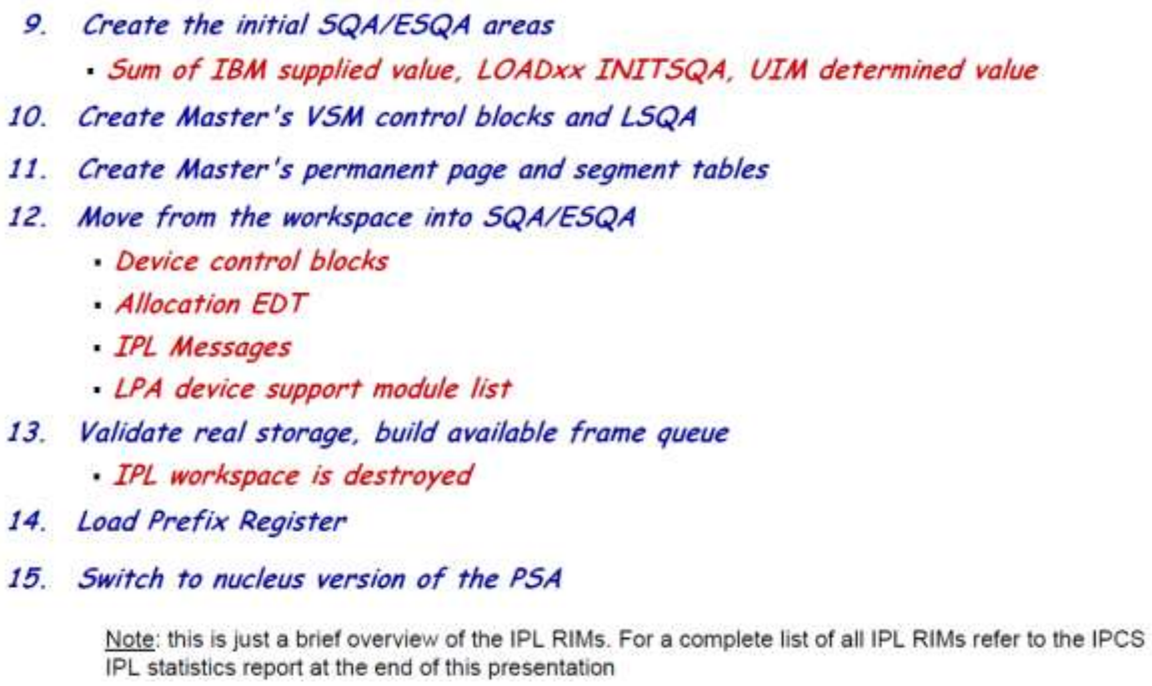
- *Includes modules contained in IEANUC0x and IEANUC2x, and those identified by NMLs, NUCLSTxx, and UIMs*
- *CSECTs are grouped/positioned by attributes, RMODE and read-only*

8. *Load modules, dynamically resolving external references*

***Store Status**

The store-status operation places an architectural-mode identification and the contents of the CPU registers, except for the TOD clock, in assigned storage locations.

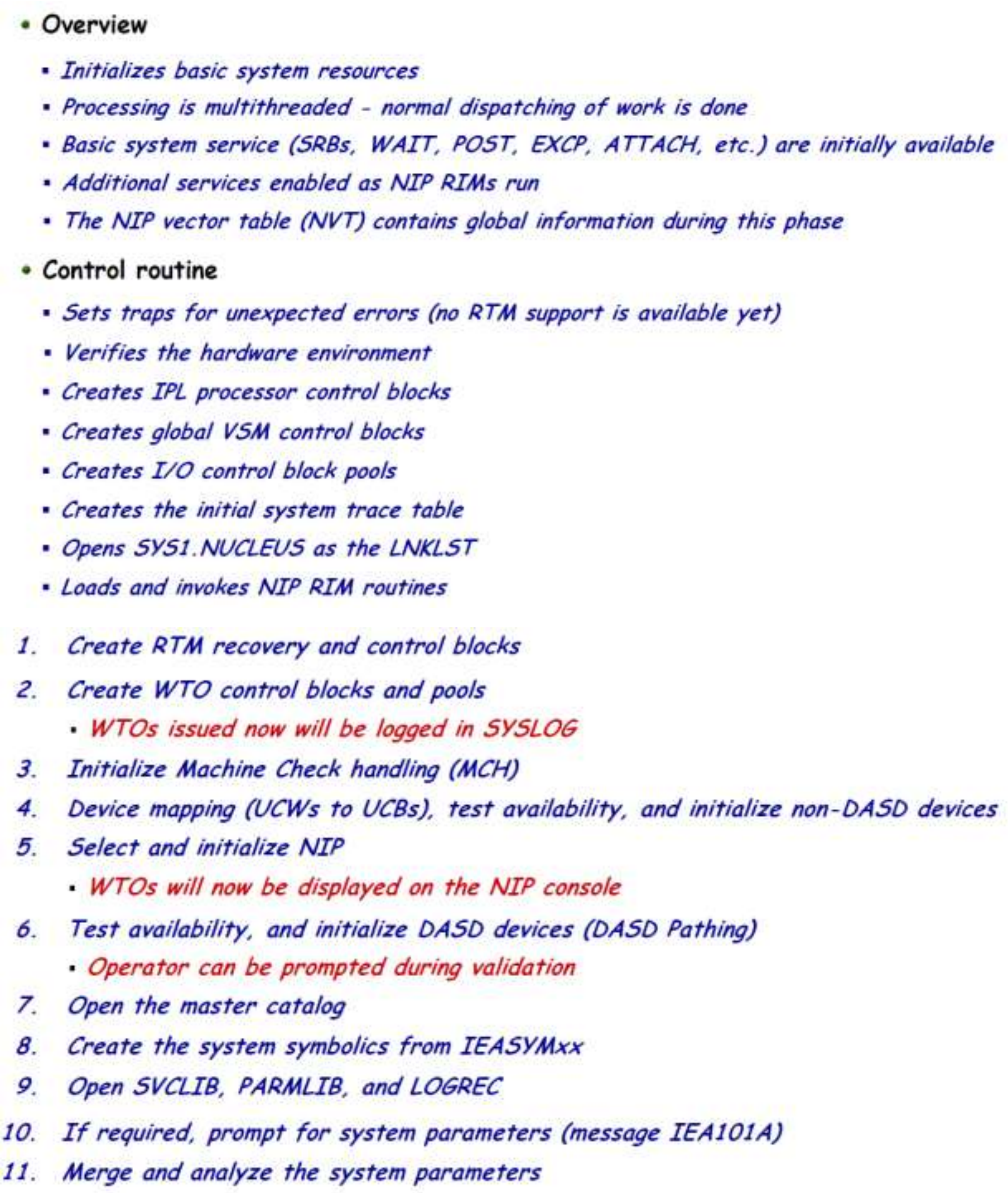
For more information, refer to „zArchitecture Pricipies of Operations", SA22-7832



9. Create the initial SQA/ESQA areas
   - Sum of IBM supplied value, LOADxx INITSQA, UIM determined value
10. Create Master's VSM control blocks and LSQA
11. Create Master's permanent page and segment tables
12. Move from the workspace into SQA/ESQA
    - Device control blocks
    - Allocation EDT
    - IPL Messages
    - LPA device support module list
13. Validate real storage, build available frame queue
    - IPL workspace is destroyed
14. Load Prefix Register
15. Switch to nucleus version of the PSA

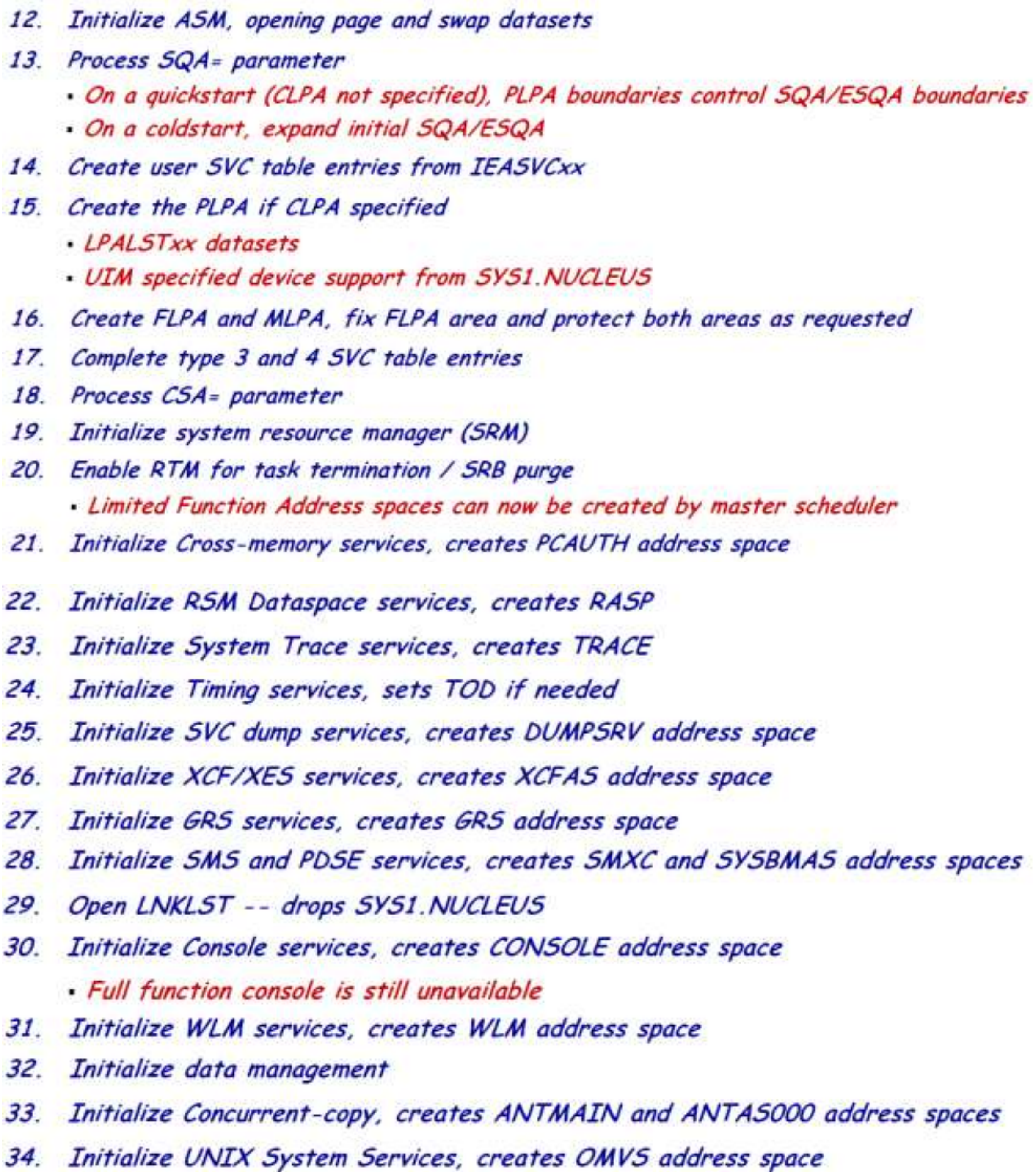
Note: this is just a brief overview of the IPL RIMs. For a complete list of all IPL RIMs refer to the IPCS IPL statistics report at the end of this presentation

NIP RIMs

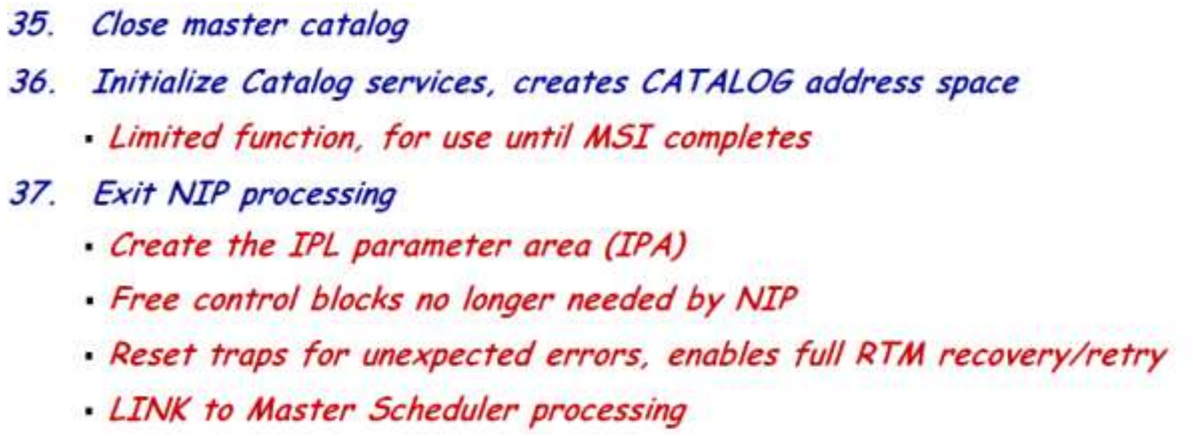


- Overview
  - *Initializes basic system resources*
  - *Processing is multithreaded - normal dispatching of work is done*
  - *Basic system service (SRBs, WAIT, POST, EXCP, ATTACH, etc.) are initially available*
  - *Additional services enabled as NIP RIMs run*
  - *The NIP vector table (NVT) contains global information during this phase*
- Control routine
  - *Sets traps for unexpected errors (no RTM support is available yet)*
  - *Verifies the hardware environment*
  - *Creates IPL processor control blocks*
  - *Creates global VSM control blocks*
  - *Creates I/O control block pools*
  - *Creates the initial system trace table*
  - *Opens SYS1.NUCLEUS as the LNKLST*
  - *Loads and invokes NIP RIM routines*

1. *Create RTM recovery and control blocks*
2. *Create WTO control blocks and pools*
   - *WTOs issued now will be logged in SYSLOG*
3. *Initialize Machine Check handling (MCH)*
4. *Device mapping (UCWs to UCBs), test availability, and initialize non-DASD devices*
5. *Select and initialize NIP*
   - *WTOs will now be displayed on the NIP console*
6. *Test availability, and initialize DASD devices (DASD Pathing)*
   - *Operator can be prompted during validation*
7. *Open the master catalog*
8. *Create the system symbolics from IEASYMxx*
9. *Open SVCLIB, PARMLIB, and LOGREC*
10. *If required, prompt for system parameters (message IEA101A)*
11. *Merge and analyze the system parameters*

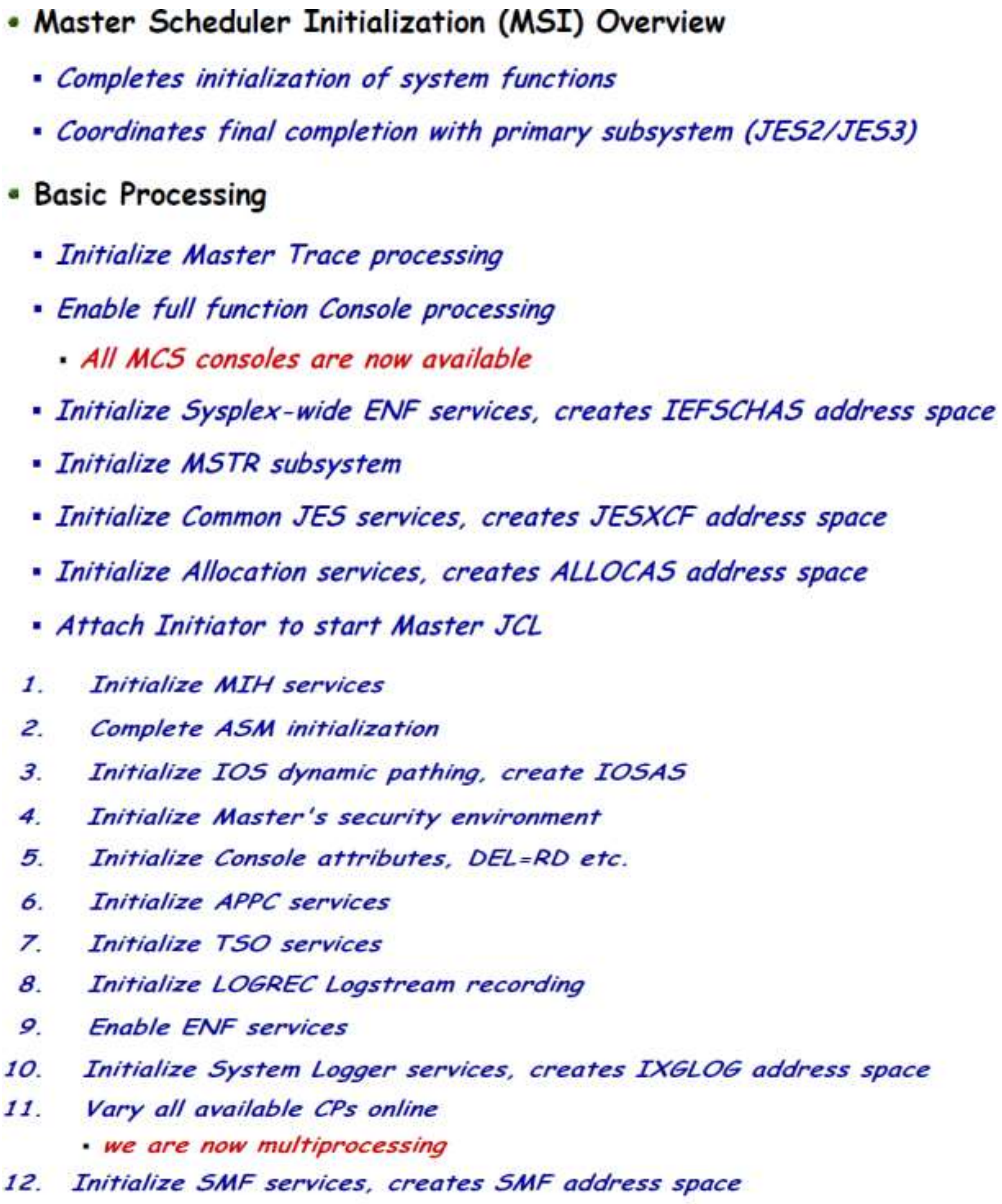


12. Initialize ASM, opening page and swap datasets
13. Process SQA= parameter
    - On a quickstart (CLPA not specified), PLPA boundaries control SQA/ESQA boundaries
    - On a coldstart, expand initial SQA/ESQA
14. Create user SVC table entries from IEASVCxx
15. Create the PLPA if CLPA specified
    - LPALSTxx datasets
    - UIM specified device support from SYS1.NUCLEUS
16. Create FLPA and MLPA, fix FLPA area and protect both areas as requested
17. Complete type 3 and 4 SVC table entries
18. Process CSA= parameter
19. Initialize system resource manager (SRM)
20. Enable RTM for task termination / SRB purge
    - Limited Function Address spaces can now be created by master scheduler
21. Initialize Cross-memory services, creates PCAUTH address space
22. Initialize RSM Dataspace services, creates RASP
23. Initialize System Trace services, creates TRACE
24. Initialize Timing services, sets TOD if needed
25. Initialize SVC dump services, creates DUMPSRV address space
26. Initialize XCF/XES services, creates XCFAS address space
27. Initialize GRS services, creates GRS address space
28. Initialize SMS and PDSE services, creates SMXC and SYSBMAS address spaces
29. Open LNKLST -- drops SYS1.NUCLEUS
30. Initialize Console services, creates CONSOLE address space
    - Full function console is still unavailable
31. Initialize WLM services, creates WLM address space
32. Initialize data management
33. Initialize Concurrent-copy, creates ANTMAIN and ANTAS000 address spaces
34. Initialize UNIX System Services, creates OMVS address space



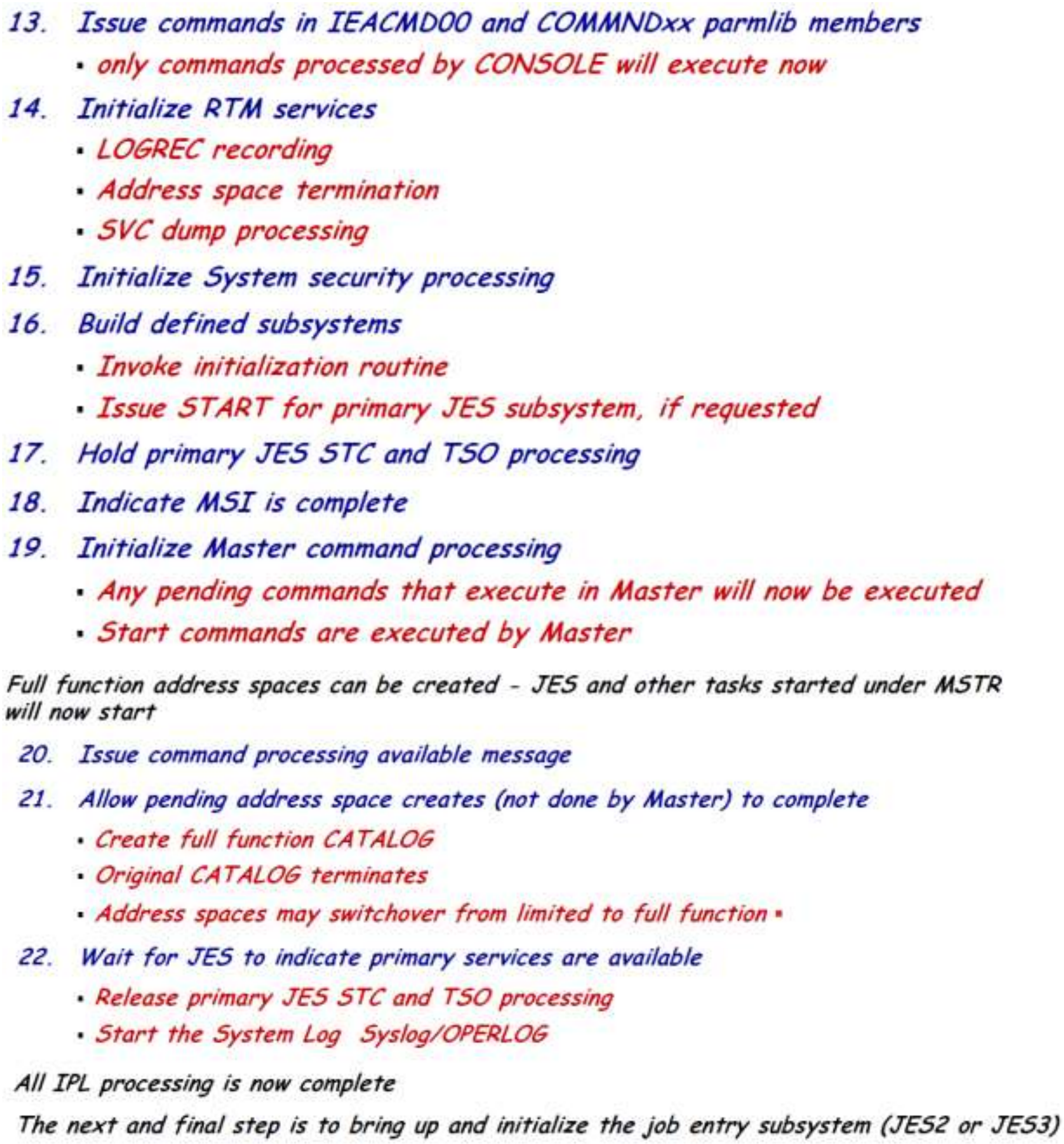
35. Close master catalog
36. Initialize Catalog services, creates CATALOG address space
    - Limited function, for use until MSI completes
37. Exit NIP processing
    - Create the IPL parameter area (IPA)
    - Free control blocks no longer needed by NIP
    - Reset traps for unexpected errors, enables full RTM recovery/retry
    - LINK to Master Scheduler processing

MSI (Master Scheduler Initialization)



- Master Scheduler Initialization (MSI) Overview
  - *Completes initialization of system functions*
  - *Coordinates final completion with primary subsystem (JES2/JES3)*
- Basic Processing
  - *Initialize Master Trace processing*
  - *Enable full function Console processing*
    - *All MCS consoles are now available*
  - *Initialize Sysplex-wide ENF services, creates IEFSCHAS address space*
  - *Initialize MSTR subsystem*
  - *Initialize Common JES services, creates JESXCF address space*
  - *Initialize Allocation services, creates ALLOCAS address space*
  - *Attach Initiator to start Master JCL*

1. *Initialize MIH services*
2. *Complete ASM initialization*
3. *Initialize IOS dynamic pathing, create IOSAS*
4. *Initialize Master's security environment*
5. *Initialize Console attributes, DEL=RD etc.*
6. *Initialize APPC services*
7. *Initialize TSO services*
8. *Initialize LOGREC Logstream recording*
9. *Enable ENF services*
10. *Initialize System Logger services, creates IXGLOG address space*
11. *Vary all available CPs online*
    - *we are now multiprocessing*
12. *Initialize SMF services, creates SMF address space*



13. Issue commands in IEACMD00 and COMMNDxx parmlib members
    - only commands processed by CONSOLE will execute now
14. Initialize RTM services
    - LOGREC recording
    - Address space termination
    - SVC dump processing
15. Initialize System security processing
16. Build defined subsystems
    - Invoke initialization routine
    - Issue START for primary JES subsystem, if requested
17. Hold primary JES STC and TSO processing
18. Indicate MSI is complete
19. Initialize Master command processing
    - Any pending commands that execute in Master will now be executed
    - Start commands are executed by Master

Full function address spaces can be created - JES and other tasks started under MSTR will now start

20. Issue command processing available message
21. Allow pending address space creates (not done by Master) to complete
    - Create full function CATALOG
    - Original CATALOG terminates
    - Address spaces may switchover from limited to full function ▪
22. Wait for JES to indicate primary services are available
    - Release primary JES STC and TSO processing
    - Start the System Log Syslog/OPERLOG

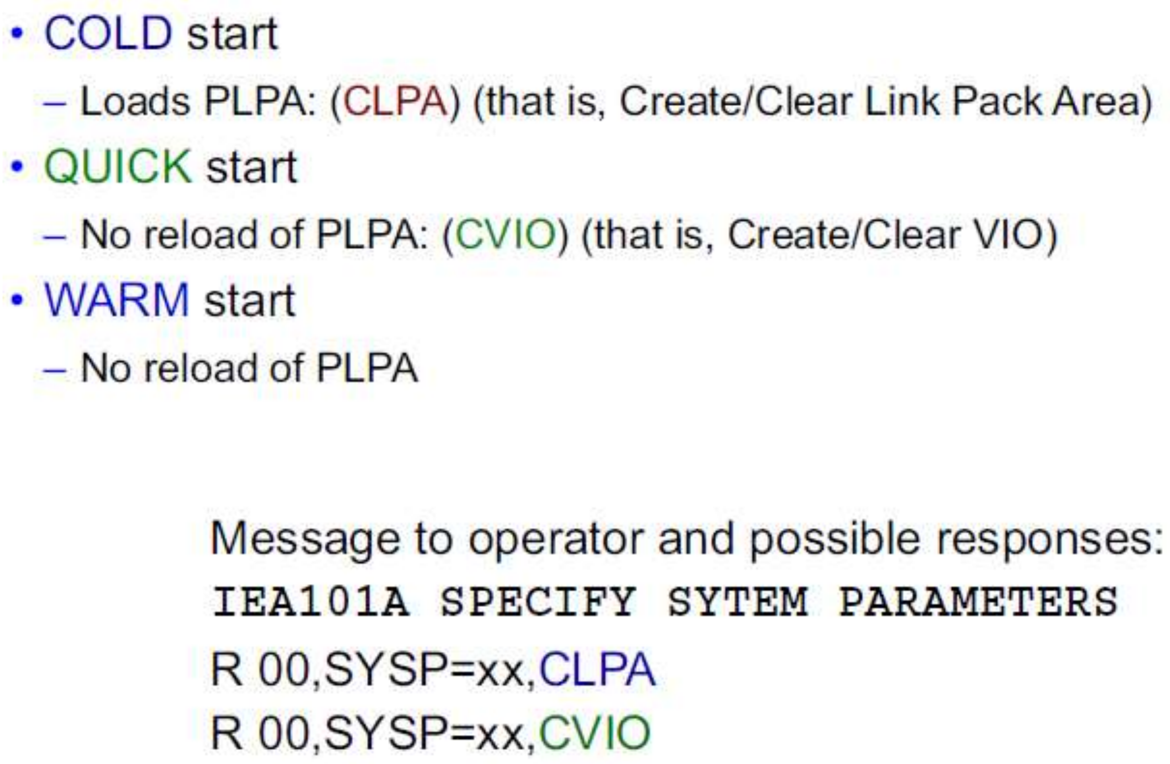
All IPL processing is now complete

The next and final step is to bring up and initialize the job entry subsystem (JES2 or JES3)

IPL - Few important abbreviations

ASM	Auixiliary Storage Manager
DAT	Dynamic Address Translation
ENF	Event Notification Facility
IOCDS	I/O Configuration Data Set
IOCP	I/O Configuration Program
IODF	I/O Definition File
IOS	Input/Output Supervisor
IPL	Initial Program Load
IRIM	IPL Resource Initialization Module
JES	Job Entry Subsystem
MCH	Machine Check Handler
MCS	Multiple Console Support
MIH	Missing Interrupt Handler
MSI	Master Scheduler Initialization
NIP	Nucleus Initialization Phase
POR	Power-on-Reset
RIM	Resource Initialization Module
RSM	Real Storage Manager
RTM	Recovery Termination Manager
SMS	System managed Storage
SRM	System Resource Manager
SVC	Supervisor Call
SYSRES	System residence Volume
TOD	Time of Day Clock
UCB	Unit Control Block
UCW	Unit Control Word
UIM	Unit Information Module
VSM	Virtual Storage Management

Types of IPL



- COLD start
  - Loads PLPA: (CLPA) (that is, Create/Clear Link Pack Area)
- QUICK start
  - No reload of PLPA: (CVIO) (that is, Create/Clear VIO)
- WARM start
  - No reload of PLPA

Message to operator and possible responses:

```
IEA101A SPECIFY SYTEM PARAMETERS
```

R 00,SYSP=xx,CLPA

R 00,SYSP=xx,CVIO

TSO/E

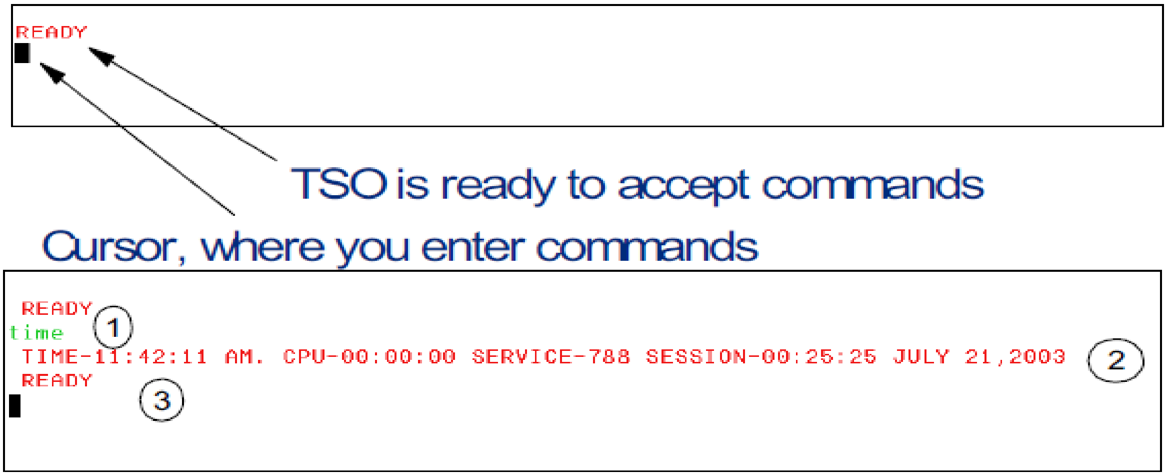
TSO allows users to log on to z/OS and access a limited set of basic TSO commands, which are available as part of the core operating system. Interacting with z/OS in this way is called using TSO in its native mode.

In a z/OS system, each user is granted a user ID and a password authorized for TSO logon. Logging on to TSO requires a 3270-display device or, more commonly, a TN3270 emulator running on a PC. During TSO logon, the system displays the TSO logon screen on the user's 3270 display device or TN3270 emulator. The logon screen serves the same purpose as a Windows logon panel.

z/OS system programmers often modify the particular text layout and information of the TSO logon panel to better suit the needs of the system's users.

Parmlib members – IKJTSOxx, TSOKEYxx,

TSO in native mode



```
READY
█
```

TSO is ready to accept commands

Cursor, where you enter commands

```
 READY
time
 TIME-11:42:11 AM. CPU-00:00:00 SERVICE-788 SESSION-00:25:25 JULY 21,2003
 READY
█
```

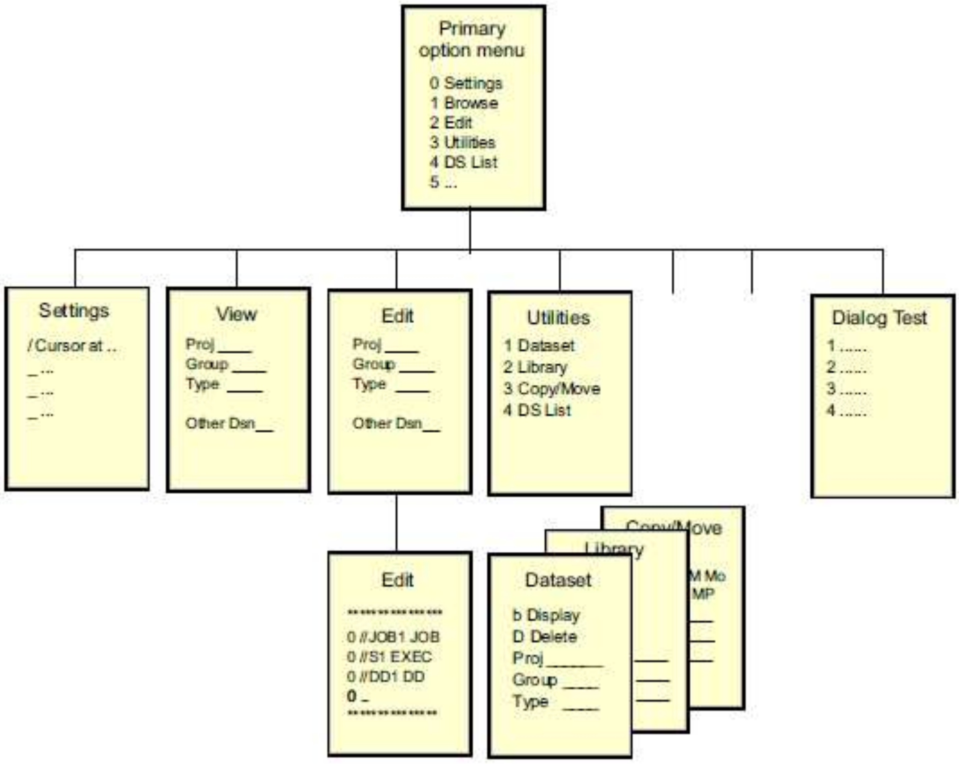
1 - You enter a command (like a DOS prompt)
2 - TSO displays the command output and
3 - TSO is ready to accept new commands

ISPF

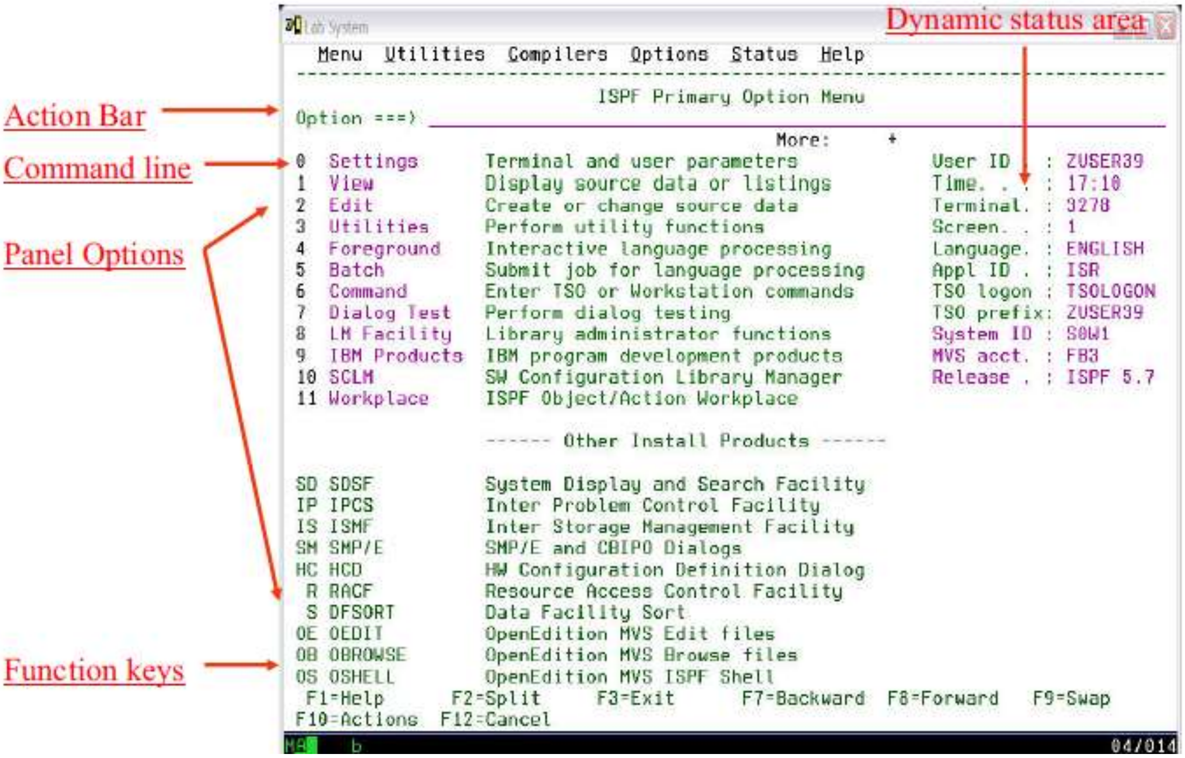
ISPF is an application that runs on z/OS and provides a menu-style shell for TSO users. ISPF menus list the functions that are most frequently needed by online users. ISPF is what many people use exclusively to perform work on z/OS.

After logging on to TSO, users typically access the ISPF menu. In fact, many use ISPF exclusively for performing work on z/OS. ISPF is a full-panel application navigated by keyboard. ISPF includes a text editor and browser, and functions for locating and listing files and performing other utility functions. ISPF menus list the functions that are most frequently needed by online users.

ISPF Menu



ISPF Primary option menu – an example



```
 Menu  Utilities  Compilers  Options  Status  Help
------------------------------------------------------------------------------
                         ISPF Primary Option Menu
Option ===> ___________________________________________________________________
                                                     More:     +
0  Settings      Terminal and user parameters            User ID  . : ZUSER39
1  View          Display source data or listings         Time. . . : 17:10
2  Edit          Create or change source data            Terminal. : 3278
3  Utilities     Perform utility functions               Screen. . : 1
4  Foreground    Interactive language processing         Language. : ENGLISH
5  Batch         Submit job for language processing      Appl ID . : ISR
6  Command       Enter TSO or Workstation commands       TSO logon : TSOLOGON
7  Dialog Test   Perform dialog testing                  TSO prefix: ZUSER39
8  LM Facility   Library administrator functions         System ID : S0W1
9  IBM Products  IBM program development products        MVS acct. : FB3
10 SCLM          SW Configuration Library Manager        Release . : ISPF 5.7
11 Workplace     ISPF Object/Action Workplace

                 ------ Other Install Products ------

SD SDSF          System Display and Search Facility
IP IPCS          Inter Problem Control Facility
IS ISMF          Inter Storage Management Facility
SM SMP/E         SMP/E and CBIPO Dialogs
HC HCD           HW Configuration Definition Dialog
 R RACF          Resource Access Control Facility
 S DFSORT        Data Facility Sort
OE OEDIT         OpenEdition MVS Edit files
OB OBROWSE       OpenEdition MVS Browse files
OS OSHELL        OpenEdition MVS ISPF Shell
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F10=Actions  F12=Cancel
```

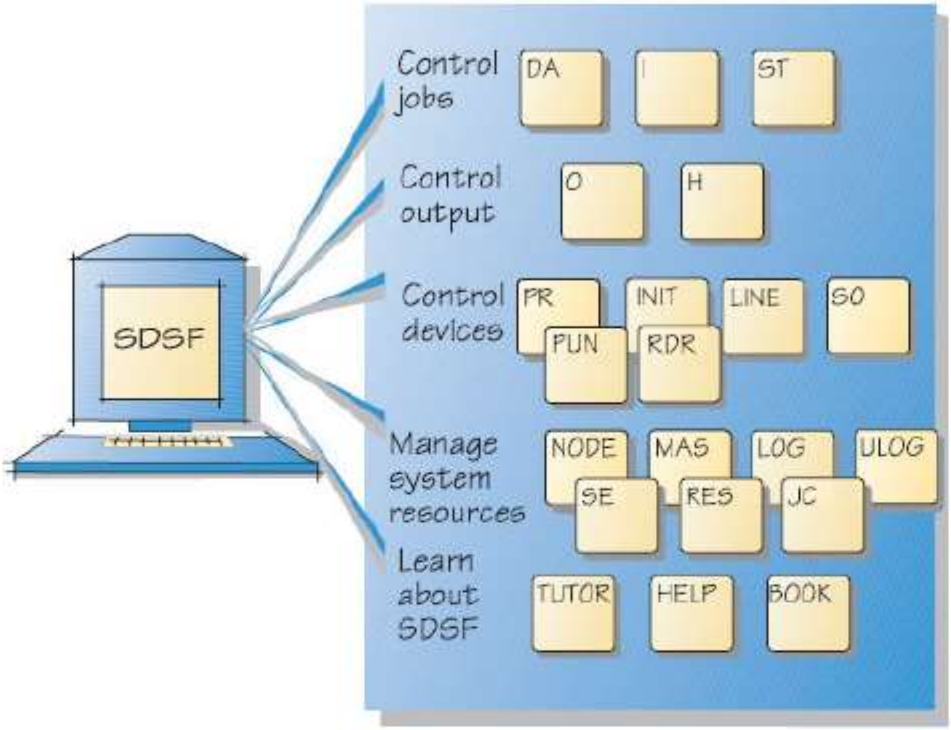
SDSF

SDSF is a software product whose primary purpose is to display printed output held in the JES spool area. Much of the printed output sent to JES by batch jobs (and other jobs) is never actually printed. Instead, it is inspected using SDSF and deleted or used as needed.

Reference: IBM manual ‘SDSF Users Guide’ and ‘SDSF Operation and Customization’.

Parmlib member - ISFPRMxx

SDSF Tasks



Here is a short description of few options in the SDSF **Primary Options** menu.

DA - The Display Active panel shows information about MVS address spaces (jobs, started tasks, and TSO users) that are running.

I - The Input Queue panel displays information about jobs, started tasks, and TSO users on the JES2 input queue or executing.

O - The Output Queue panel displays information about SYSOUT data sets for jobs, started tasks, and TSO users on any *nonheld* JES2 output queue.

H - The Held Output panel shows information about SYSOUT data sets for jobs, started tasks, and TSO users on any *held* JES2 output queue.

ST - The Status panel displays information about jobs, started tasks, and TSO users on the JES2 queues.

LOG - The system Log panel displays the log and lets you search it.

PS - The Processes panel displays information about z/OS UNIX System Services processes.

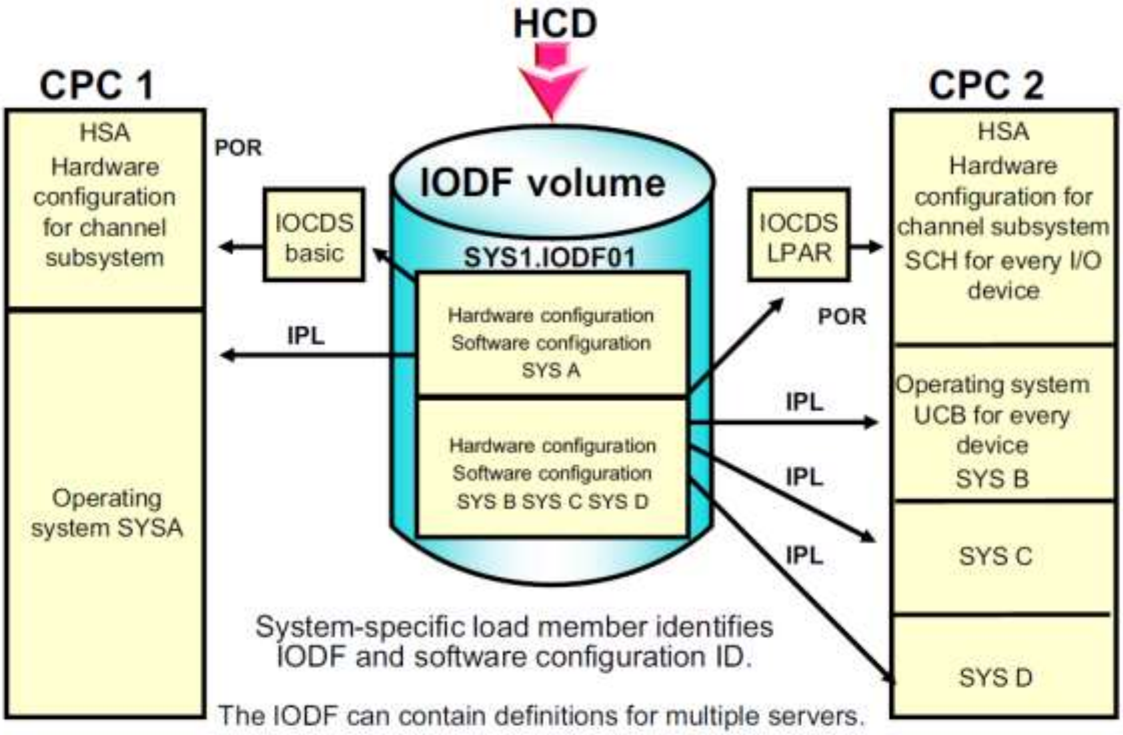
PR - The Printers panel displays information about JES2 printers printing jobs, started task, and TSO user output.

HCD

Definition

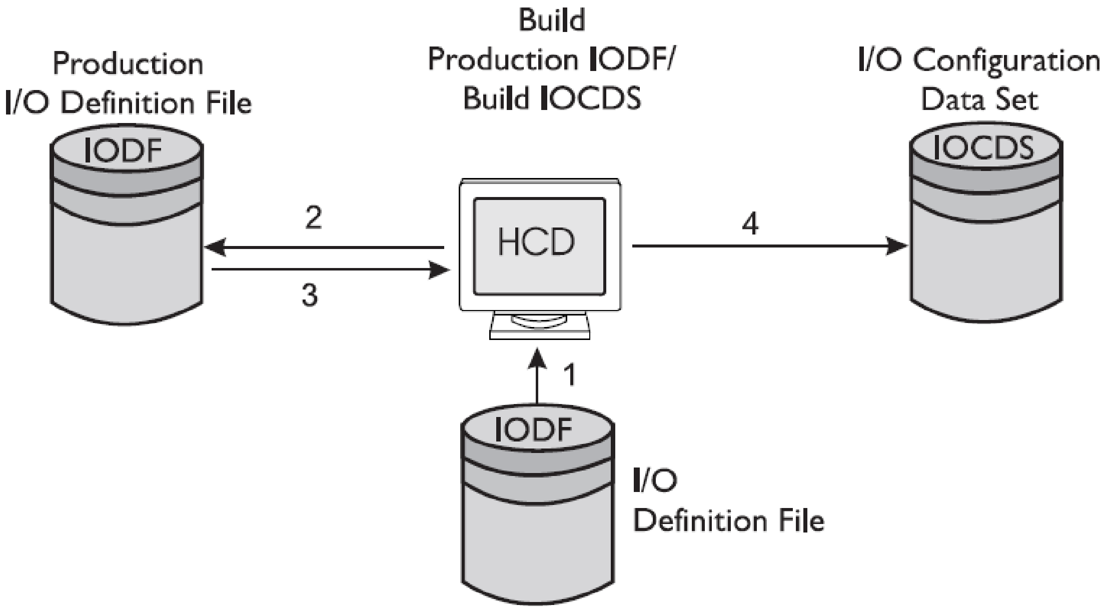
HCD (Hardware Configuration Definition) is an ISPF based element under zOS, used to define configuration information both to the CPC and to the operating system.

HCD allows you to dynamically change the current I/O configuration of the CPC. HCD allows you to dynamically change the current I/O configuration of the operating system and to create an IOCDS and make it the active IOCDS.



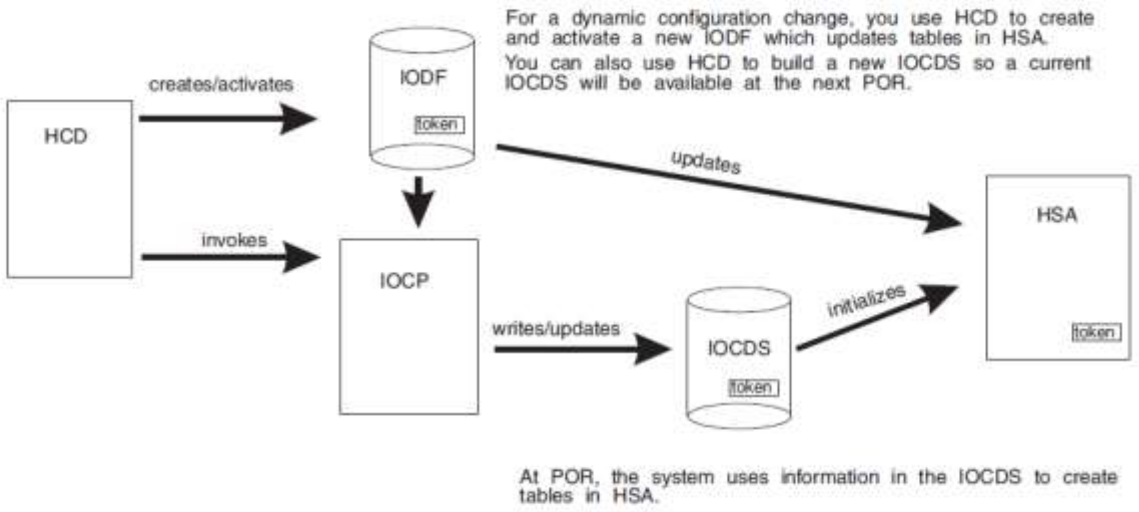
Build Production IODF

You must create a production IODF from the work IODF to be used by the systems. The production IODF cannot be updated (read-only). This ensures that the data in the production IODF used at IPL remains the same during the run time of that system.



Define IO configuration to Hardware

This is done, when you Power On Reset (POR) the CEC.

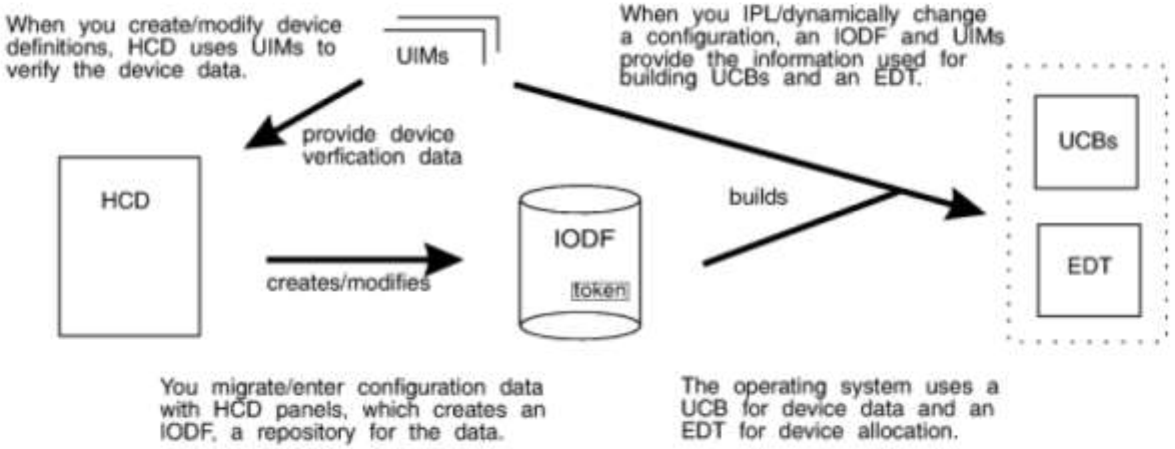


For a dynamic configuration change, you use HCD to create and activate a new IODF which updates tables in HSA. You can also use HCD to build a new IOCDS so a current IOCDS will be available at the next POR.

At POR, the system uses information in the IOCDS to create tables in HSA.

Define IO configuration to Software

This is done, when you IPL your system.



When you create/modify device definitions, HCD uses UIMs to verify the device data.

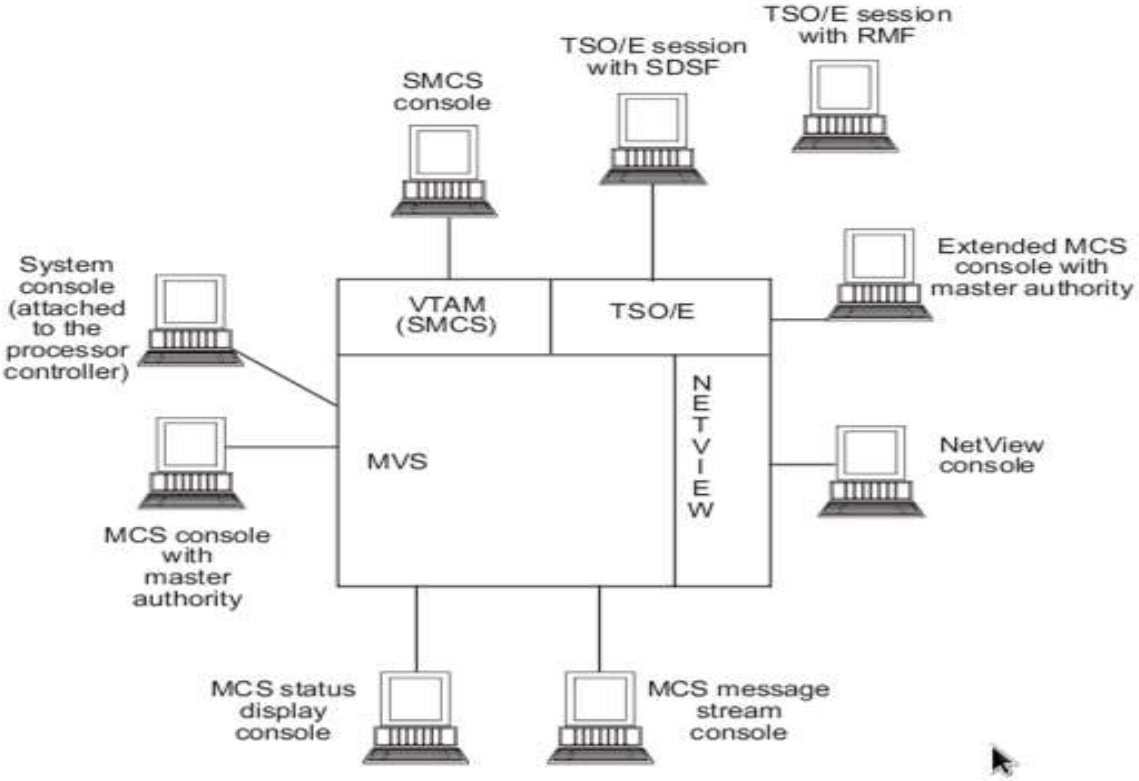
When you IPL/dynamically change a configuration, an IODF and UIMs provide the information used for building UCBs and an EDT.

You migrate/enter configuration data with HCD panels, which creates an IODF, a repository for the data.

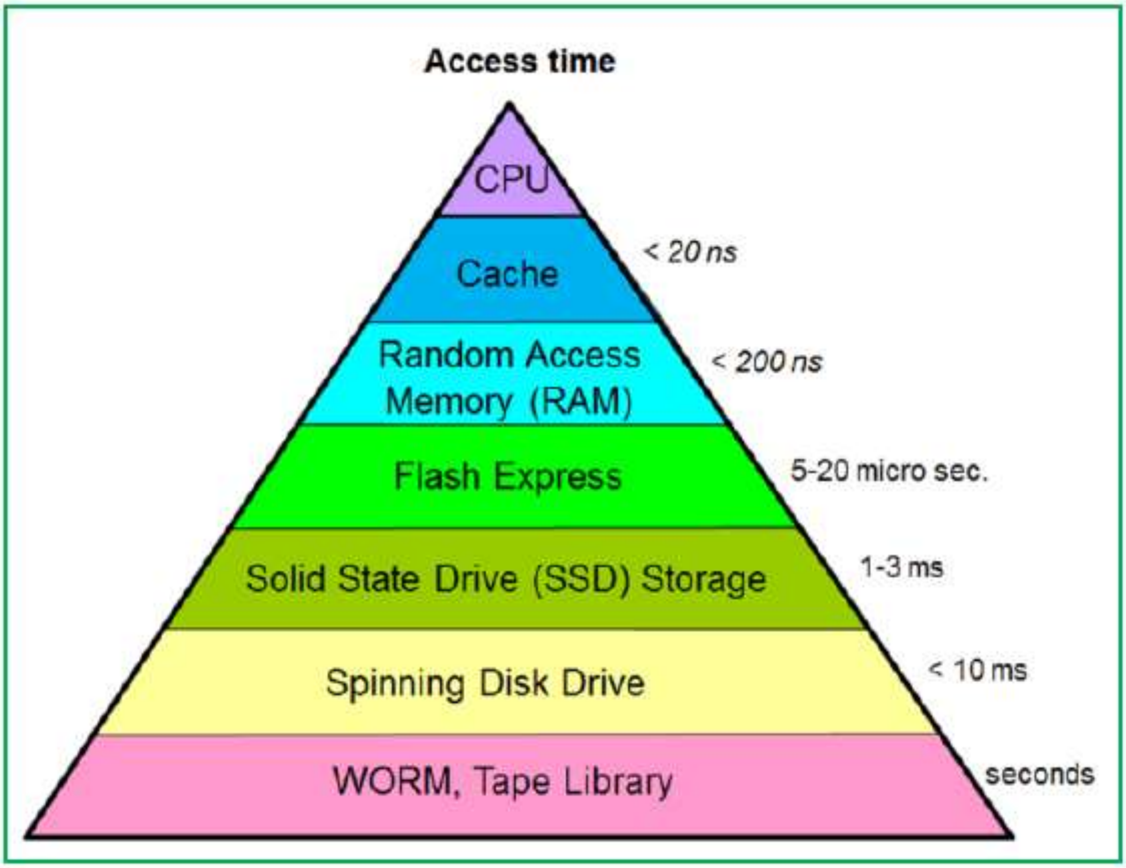
The operating system uses a UCB for device data and an EDT for device allocation.

Console

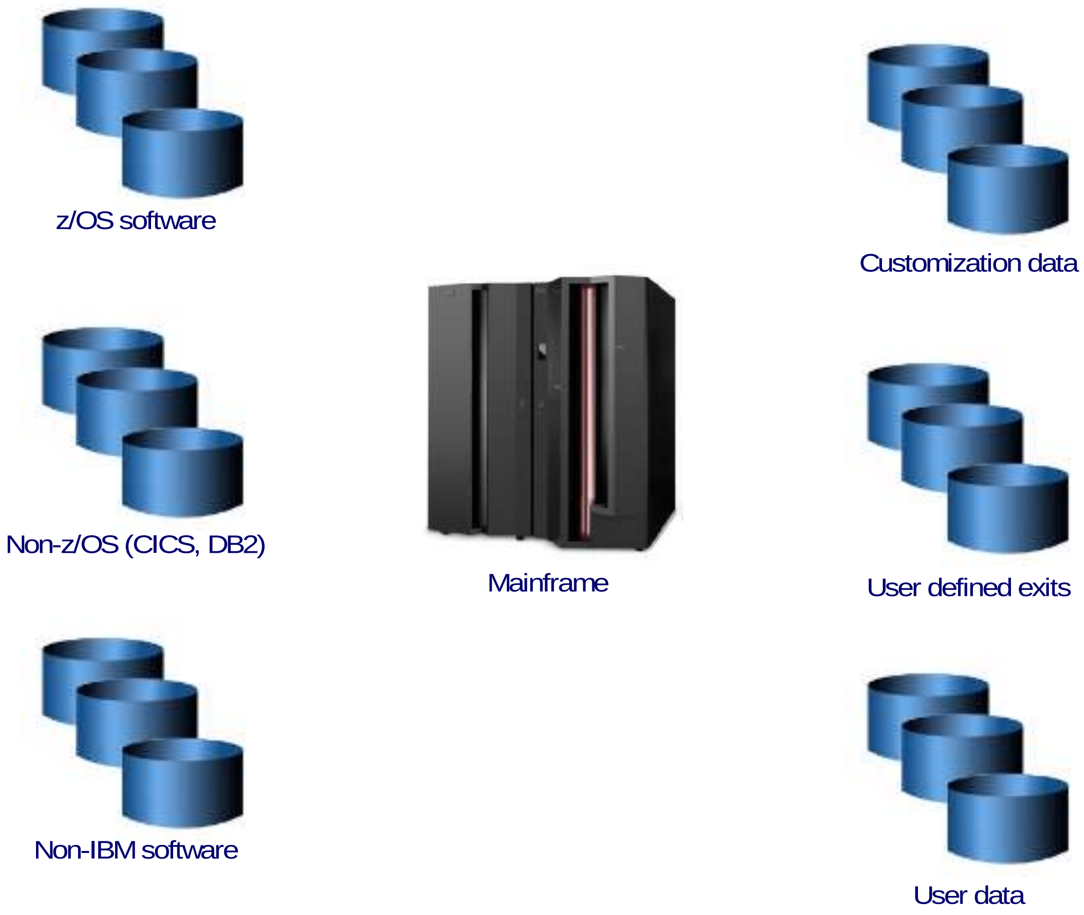
CONSOLxx is an installation-created member of parmlib in which you can define a console configuration to meet the particular needs of your installation. Consoles are used to interact with the system such as issue of commands and taking various actions.


TSO/E session with RMF
TSO/E session with SDSF
SMCS console
System console (attached to the processor controller)
VTAM (SMCS)
TSO/E
Extended MCS console with master authority
MVS
NETVIEW
NetView console
MCS console with master authority
MCS status display console
MCS message stream console

Storage access time



zOS System libraries

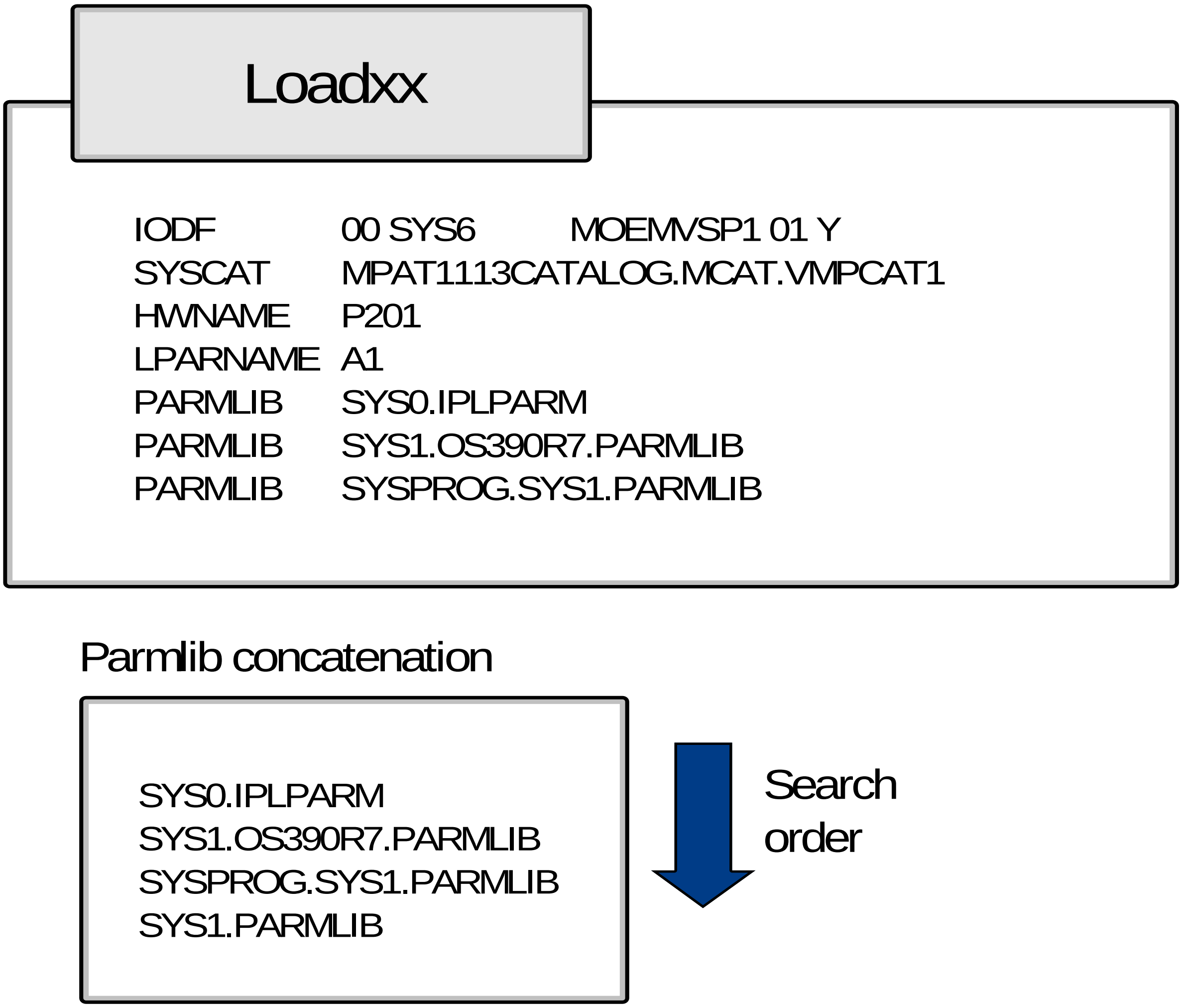


These are the most important system libraries

- SYS1.LINKLIB prime system software library
- SYS1.LPALIB system subroutines
- SYS1.NUCLEUS basic supervisor modules
- SYS1.PROCLIB system procedure JCL
- SYS1.PARMLIB control parameters

Logical Parmlib

The parmlib concatenation is called the logical parmlib.



Loadxx

```
IODF        00 SYS6     MOEMVSP1 01 Y
SYSCAT      MPAT1113CATALOG.MCAT.VMPCAT1
HWNAME      P201
LPARNAME    A1
PARMLIB     SYS0.IPLPARM
PARMLIB     SYS1.OS390R7.PARMLIB
PARMLIB     SYSPROG.SYS1.PARMLIB
```

Parmlib concatenation

```
SYS0.IPLPARM
SYS1.OS390R7.PARMLIB
SYSPROG.SYS1.PARMLIB
SYS1.PARMLIB
```

LLA and VLF

- Library Look Aside (LLA) is an address space which maintains a copy of the directories of selected libraries to improve performance
- By default, LLA always caches the linklist directories
- Virtual Lookaside Facility (VLF) is another address space which caches the most popular modules which reduces fetch time
- VLF can also cache other data objects such as RACF tables to improve response

Parmlib members:

CSVLLAxx for LLA

COFVLFxx for VLF

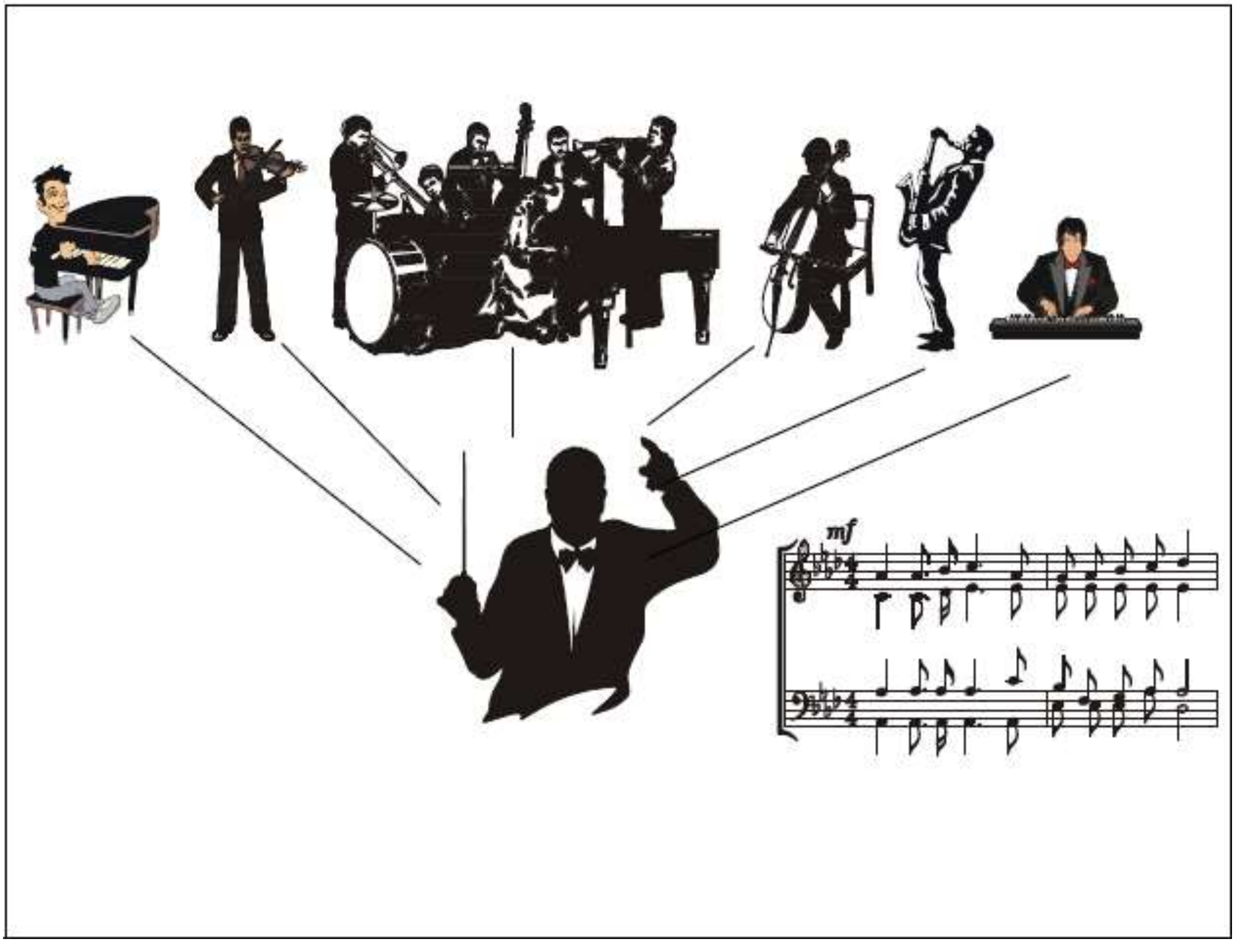
Sysplex

September 1990, IBM debuted the SYSPLEX introducing XCF services allowing authorized applications to communicate with applications on the same or other systems using specialized links.

BASIC Sysplex – A shared couple data set used between all the images holds control information and provides a mechanism for monitoring the status of the images.

Parallel Sysplex – This enhanced sysplex implementation provided the capability to use a specialized LIC operating environment called the coupling facility control code (CFCC) offering speed and integrity to shared data.

A Sysplex is a combination of multiple systems acting as a single logical system and synchronize perfectly like a symphony orchestra.

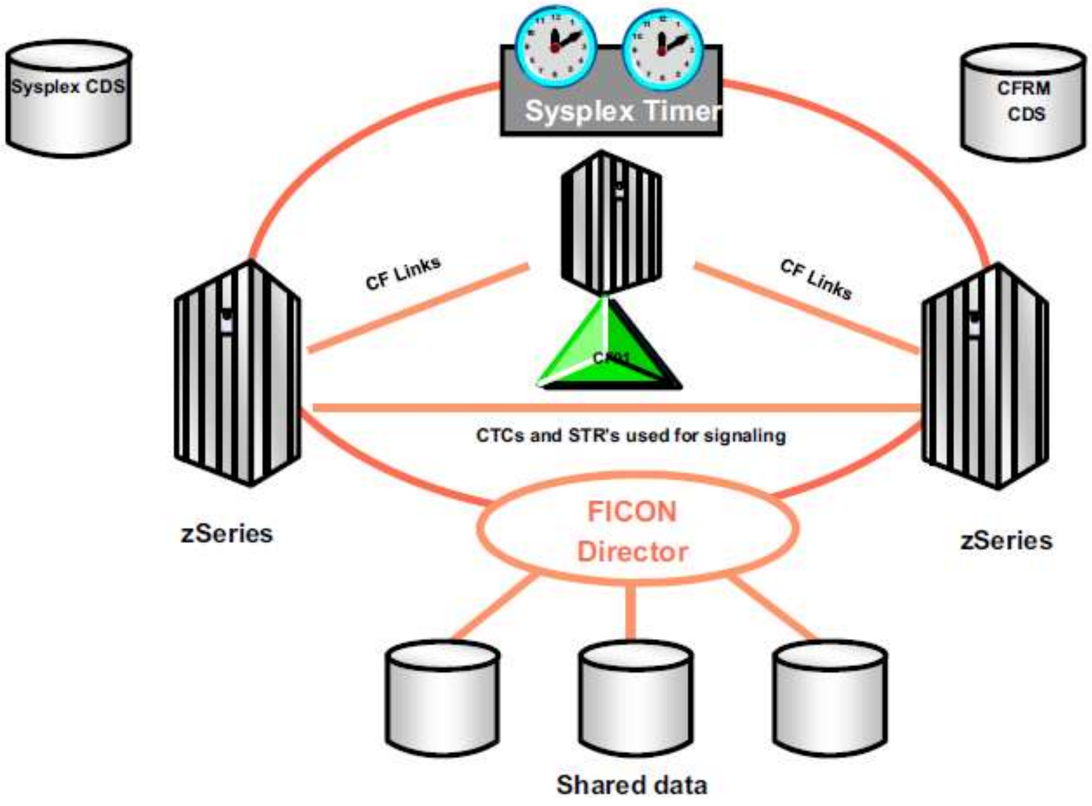


The sysplex symmetry

You can think of a sysplex as a symphony orchestra. The orchestra consists of violins, flutes, oboes, and so on. Think of each instrument as representing a different product (or component) in the sysplex. The fact that you have several of each instrument corresponds to having several images of the same product in the sysplex.

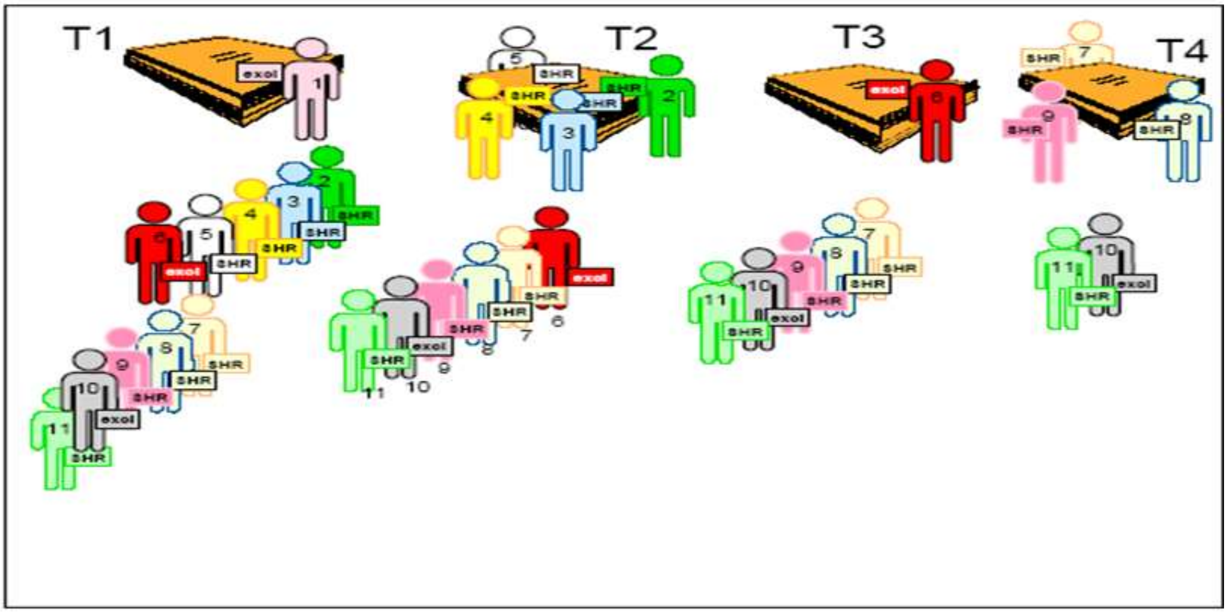
Think of symmetry in the orchestra in the following ways:

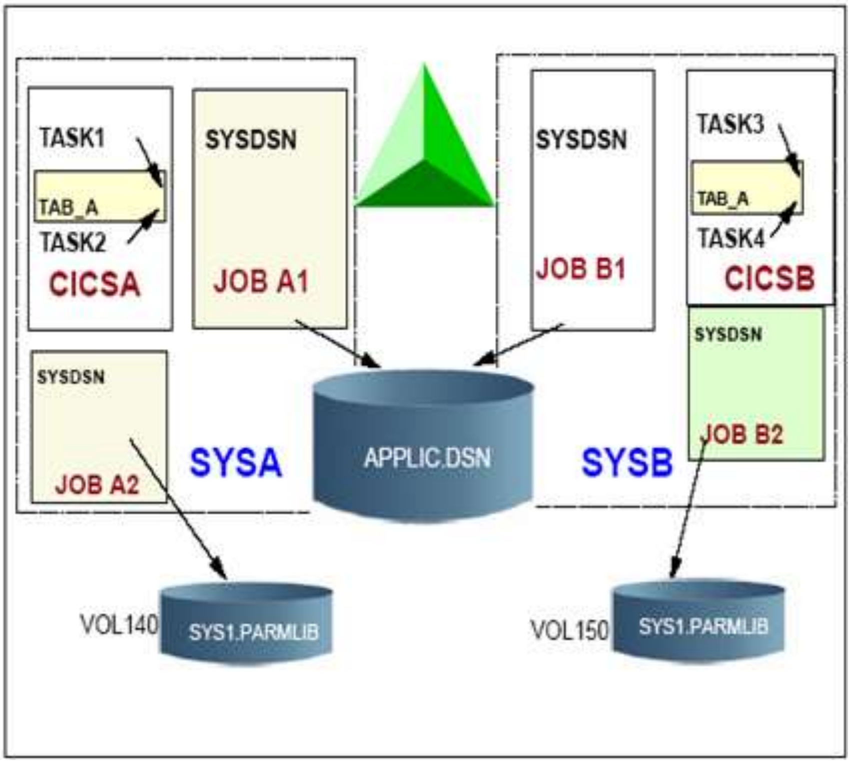
- All the violins (or whatever instrument) sound basically the same, and play the same musical part.
- All the instruments in the orchestra share the same musical score. Each instrument plays the appropriate part for that instrument.



Global Resource Serialization

Four tasks – T1 to T4. Access could be exclusive or shared.





STEP : GRS prevents any other task to access TAB_A in CICSA ASID. If TASK2 requests access, TASK2 must wait until TASK1 releases TAB_A.

SYSTEM : GRS prevents other address spaces in SYSA to use the resource with the same name.

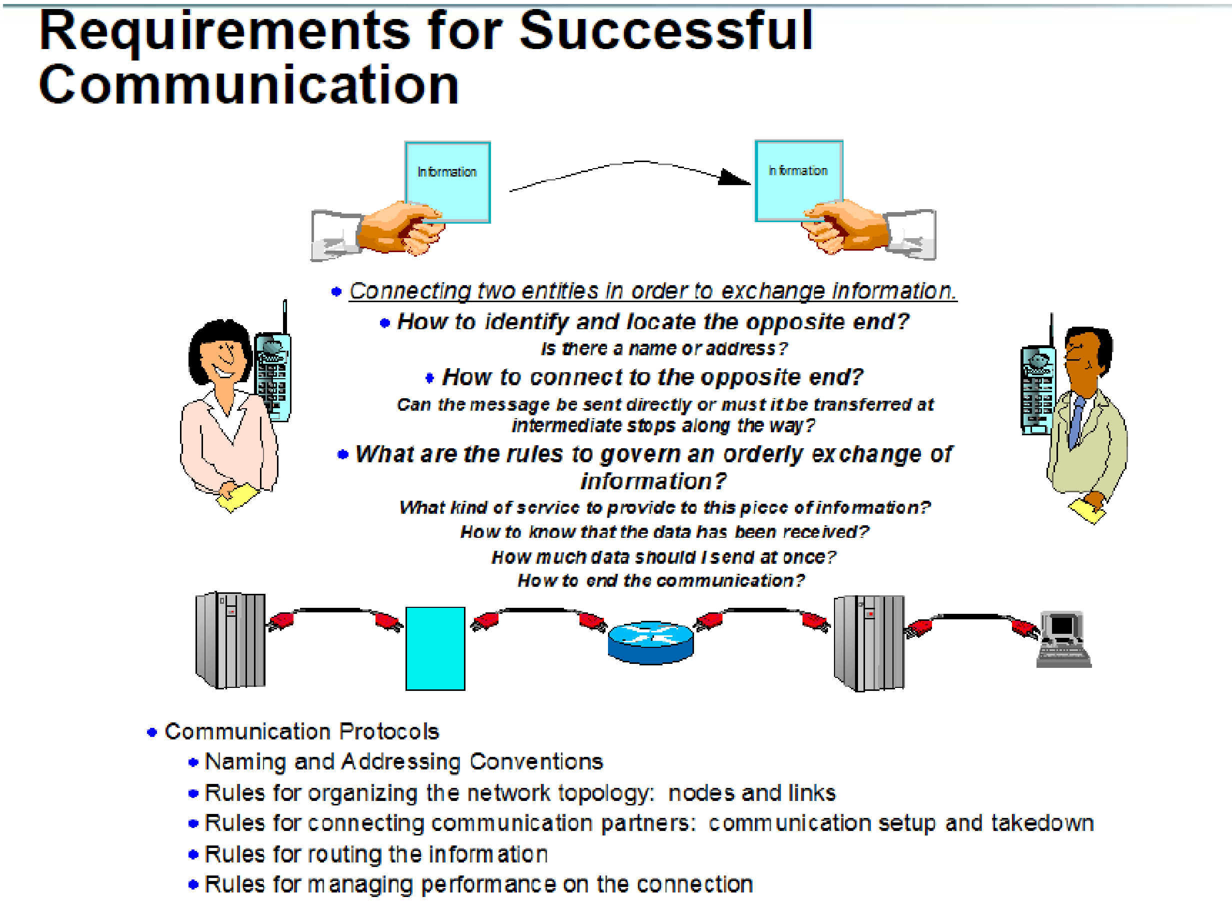
SYSTEMS: GRS prevents other address spaces in the parallel sysplex from using a resource with the name TAB_A

Parmlib members – GRSCNFxx, GRSRNLxx

IBM Communication Server (Network)

I learnt the fundamental of a communications like this.

1. You need minimum two entities (end points) to communicate.
2. They must be connected through a medium such as; air, wire, wireless, satellite etc.
3. Both the entities must be in ‘Active’ state. Not asleep.
4. One end initiates the communication and the other end responds.
5. They agree on the protocol, such as; language, only one end talks at a time etc.
6. Communication continues.
7. Communication ends.



Requirements for Successful Communication

- *Connecting two entities in order to exchange information.*
  - ***How to identify and locate the opposite end?***

    Is there a name or address?
  - ***How to connect to the opposite end?***

    Can the message be sent directly or must it be transferred at intermediate stops along the way?
  - ***What are the rules to govern an orderly exchange of information?***

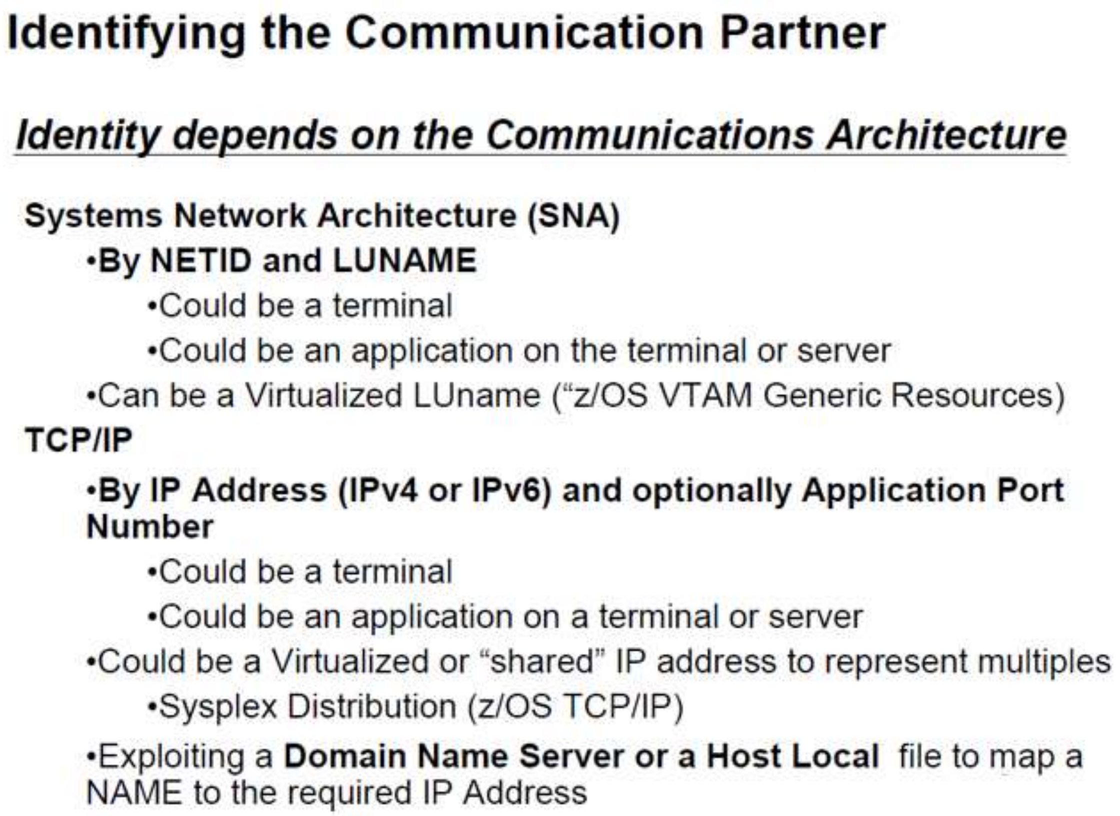
    What kind of service to provide to this piece of information?

    How to know that the data has been received?

    How much data should I send at once?

    How to end the communication?

- Communication Protocols
  - Naming and Addressing Conventions
  - Rules for organizing the network topology: nodes and links
  - Rules for connecting communication partners: communication setup and takedown
  - Rules for routing the information
  - Rules for managing performance on the connection



Identifying the Communication Partner

Identity depends on the Communications Architecture

Systems Network Architecture (SNA)

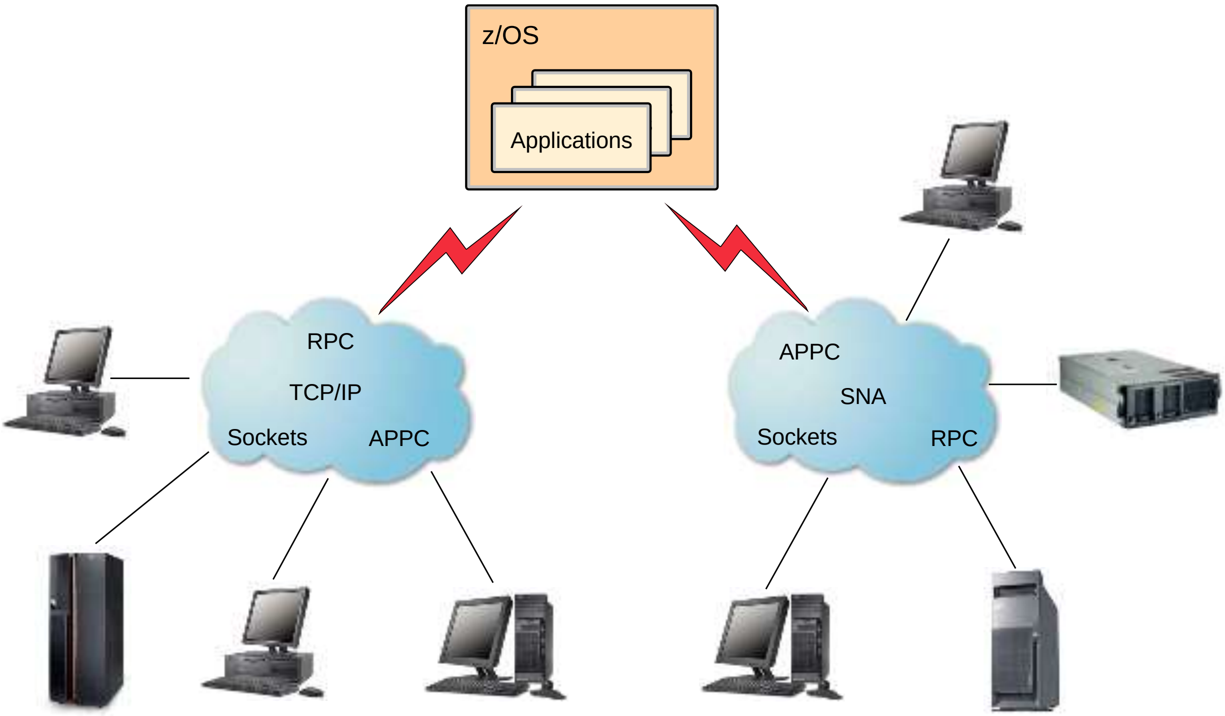
- **By NETID and LUNAME**
  - Could be a terminal
  - Could be an application on the terminal or server
- Can be a Virtualized LUname ("z/OS VTAM Generic Resources)

TCP/IP

- **By IP Address (IPv4 or IPv6) and optionally Application Port Number**
  - Could be a terminal
  - Could be an application on a terminal or server
- Could be a Virtualized or "shared" IP address to represent multiples
  - Sysplex Distribution (z/OS TCP/IP)
- Exploiting a **Domain Name Server or a Host Local** file to map a NAME to the required IP Address

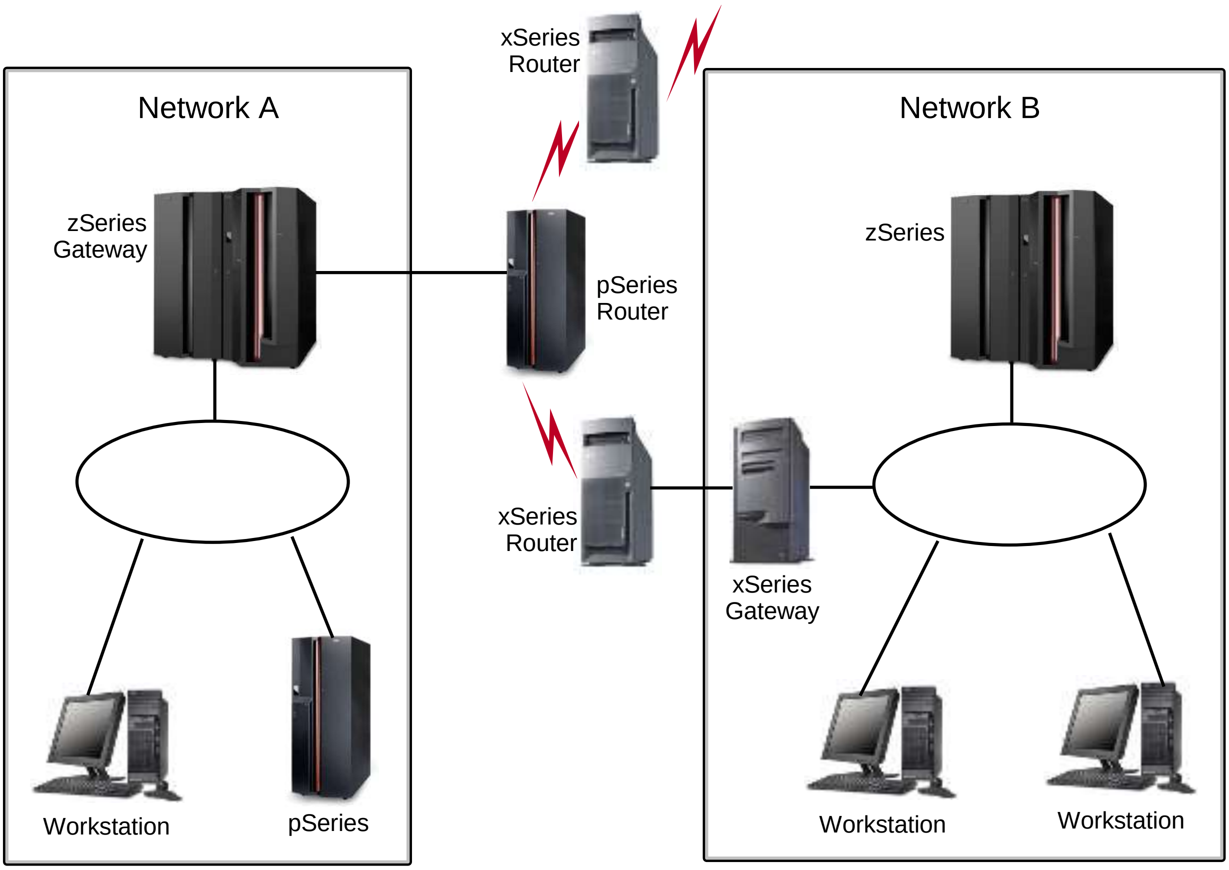
IBM Communication Server

Consists of SNA and TCP/IP.



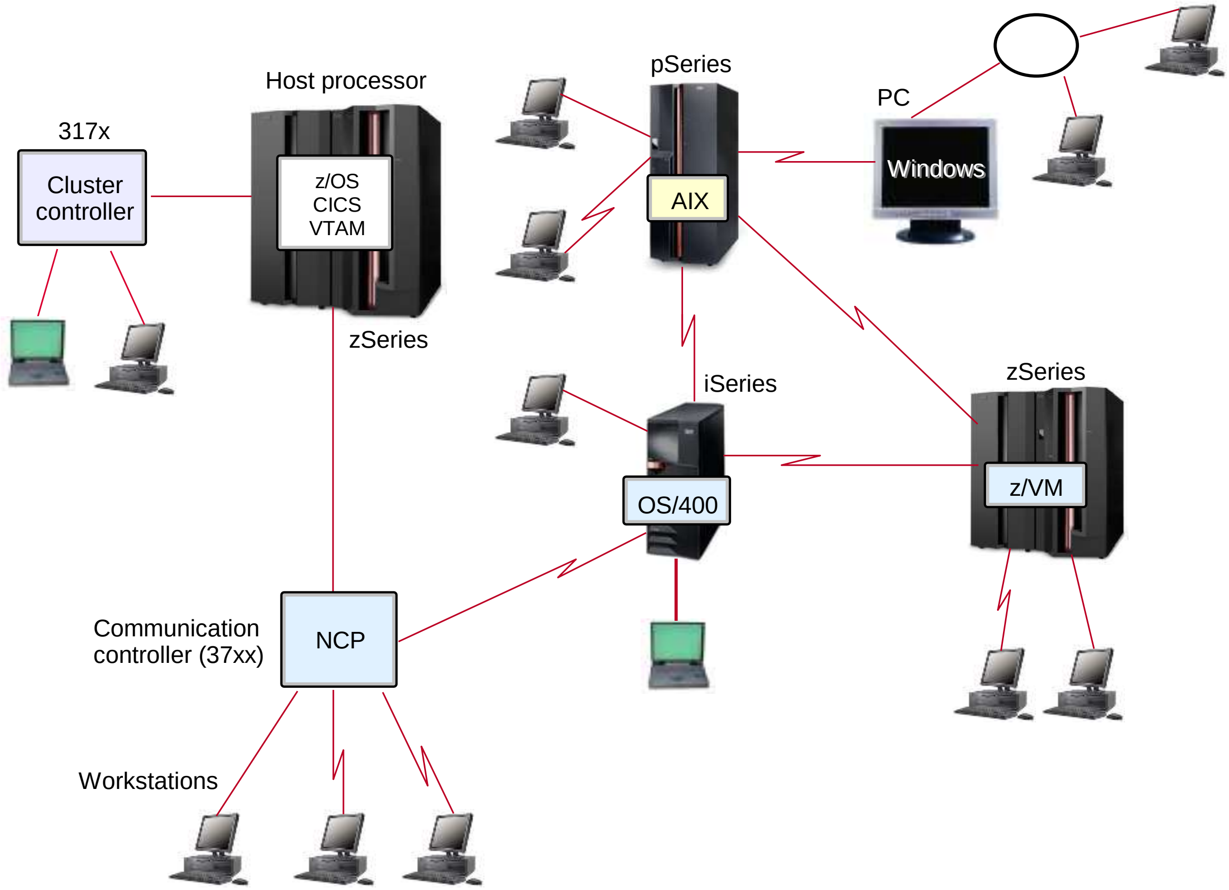
TCPIP

You must be very familiar with TCP/IP.



SNA

It is the element VTAM.



VTAM Parameters (SNA)

ATCSTRxx

Start options provide information about the conditions under which VTAM runs. When you start VTAM, a list of user-defined default values (ATCSTR00) is read from the VTAM definition library. If VTAM cannot find the ATCSTR00 file, it prompts you, giving you a choice of the following:

- Using default values

- Using an alternate start option file (ATCSTRxx, where xx specifies the identifier of the start option file you want to use)
- Cancelling the start attempt

ATCONxx

A configuration list specifies the resources that are to be activated when you start VTAM. Place the member names of the resources you want to have activated when VTAM starts into an ATCCONxx member in the VTAM definition library, where xx is any two alphanumeric characters. The value xx can then be used on the CONFIG operand of the VTAM START command, or on the CONFIG start option in your start option list, to specify which definitions are to be activated at startup.

TCPIP Parms

Two configuration files are used by the TCP/IP stack, PROFILE.TCPIP and TCPIP.DATA. PROFILE.TCPIP is used only for the configuration of the TCP/IP stack. TCPIP.DATA is used during configuration of both the TCP/IP stack and applications.

Files used by the TCP/IP stack

- PROFILE.TCPIP - used only for the stack
  - System operation and configuration parameters
  - A sample data set, hlq.SEZAINST(SAMPPROF), can be copied and modified for use as your default configuration
- TCPIP.DATA - used by stack and applications
  - Configuration information used by TCP/IP clients
  - Create a TCPIP.DATA file by copying the sample provided in SEZAINST(TCPDATA) and modifying it to suit your local conditions

TSO Logon Procedure

A TSO/E logon procedure contains JCL statements that execute the required program and allocate the required data sets to enable a user to acquire the resources needed to use TSO/E.

You must provide at least one logon procedure for your installation's TSO/E users.

Before users can log on to TSO/E, you must give them access to the system. You must review and, if necessary, adjust the variables that limit the number of users who can be logged on at one time. You must also review the size of users' address spaces and, if necessary, change the size. You can specify a default size that applies to all users, a default that varies from one logon procedure to another, or a default that applies to individual users.

You must write and make available the logon procedures that give users access to the resources they need.

Allocation of DD to TSO Logon Proc

- Allocation of Libraries.
  - STEPLIB – Load Libraries
  - ISPMLIB – Message
  - SYSHELP – Help
  - ISPPLIB – Panel
  - ISPSLIB – Skeleton
  - ISPTLIB – Table Input
  - ISPILIB – Image Library
  - SYSPROC – REXX / CLIST Library

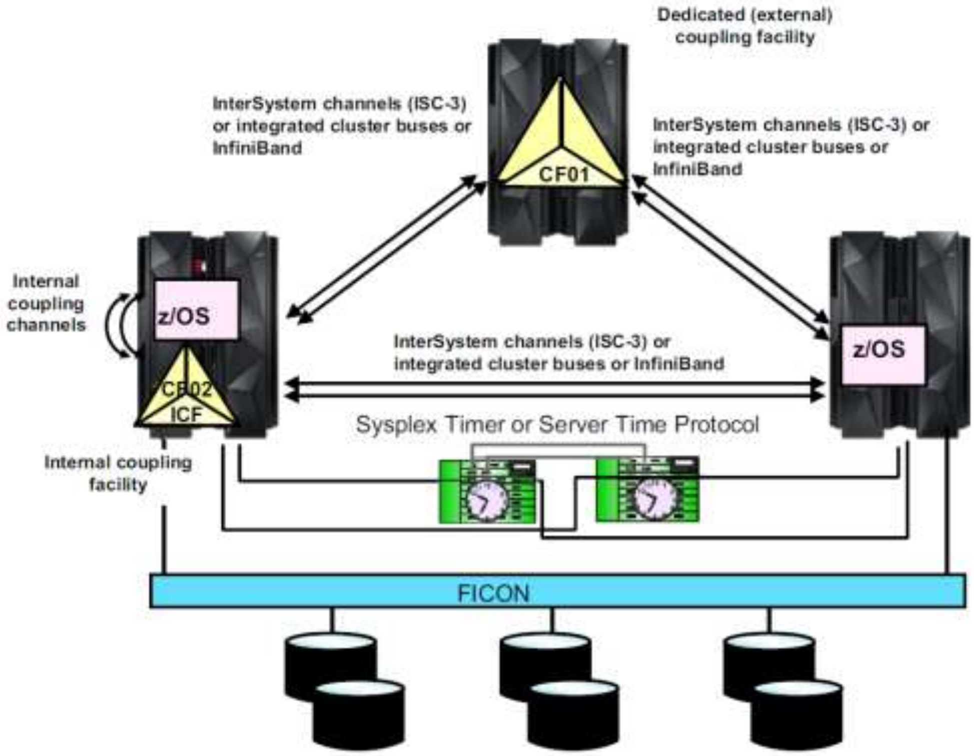
- SYSEXEC – REXX Library
- ISPTABLE – Table output
- ISPPROF – User Profile
- ISPFILE – File Tailoring Output
- ISPLLIB – Load Modules

Coupling Facility (CF)

A **Coupling Facility** or **CF** is a piece of computer hardware or virtual machine that coordinates multiple processors.

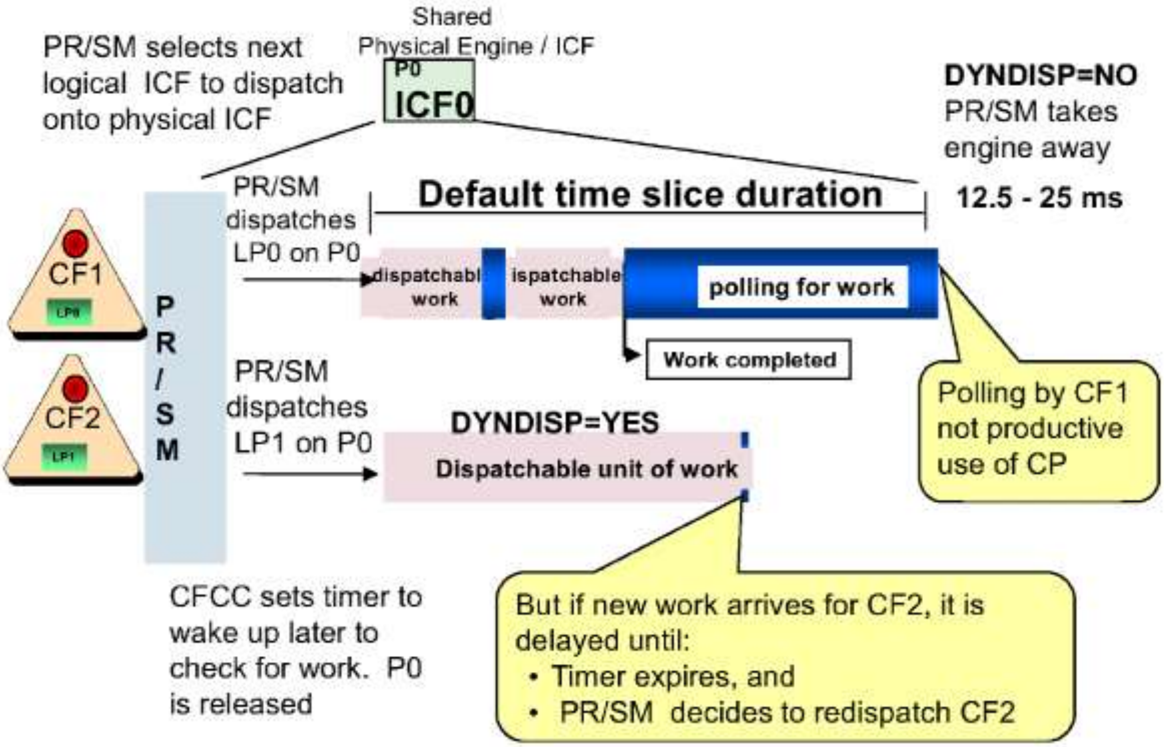
A Parallel Sysplex relies on one or more Coupling Facilities (CFs). A coupling facility is a mainframe processor (runs in an own LPAR, with dedicated or shared physical CP, defined through Hardware Management Console (HMC)), with memory and special channels (CF Links), and a specialized operating system called Coupling Facility Control Code (CFCC). It has no I/O devices, other than the CF links. The information in the CF resides entirely in memory as CFCC is not a virtual memory operating system. A CF typically has a large memory. The CF runs no application software.

The CF could be internal CF which runs in an LPAR along with the other LPARs in the CEC using ICF or GP engines. Or it could be external CF where the CEC is dedicated and configured for CF LPARs only with ICF engines.

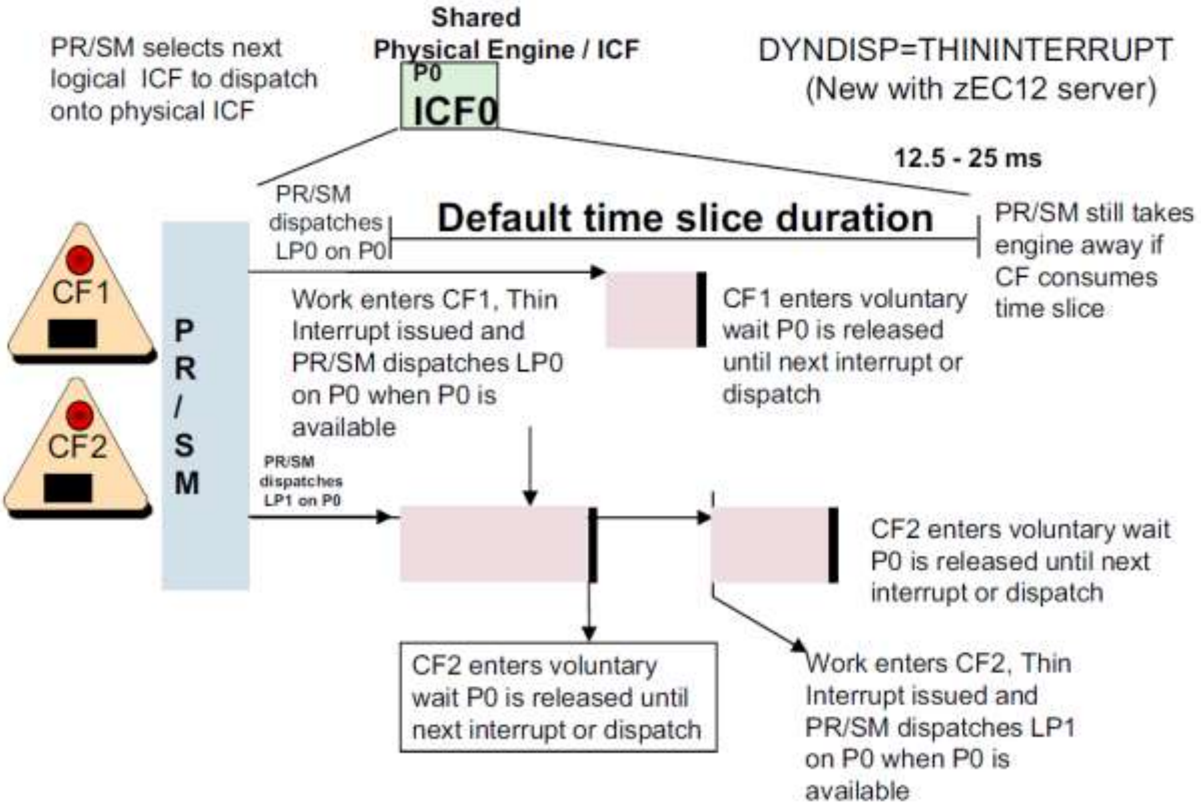

Dedicated (external) coupling facility
InterSystem channels (ISC-3) or integrated cluster buses or InfiniBand
CF01
InterSystem channels (ISC-3) or integrated cluster buses or InfiniBand
Internal coupling channels
z/OS
CF02 ICF
InterSystem channels (ISC-3) or integrated cluster buses or InfiniBand
z/OS
Sysplex Timer or Server Time Protocol
Internal coupling facility
FICON

Dynamic CF dispatching

Dynamic CF Dispatching is a function which allows the installation to limit the impact of CF polling when the CF has a low activity rates and is considered to not be a production CFs. This capability is provided by PR/SM and is available to all IBM CF partitions which share CPs.


Shared
Physical Engine / ICF
P0
ICF0
PR/SM selects next logical ICF to dispatch onto physical ICF
DYNDISP=NO
PR/SM takes engine away
12.5 - 25 ms
Default time slice duration
PR/SM dispatches LP0 on P0
dispatchable work
ispatchable work
polling for work
Work completed
CF1
LP0
CF2
LP1
P R / S M
PR/SM dispatches LP1 on P0
DYNDISP=YES
Dispatchable unit of work
Polling by CF1 not productive use of CP
CFCC sets timer to wake up later to check for work. P0 is released
But if new work arrives for CF2, it is delayed until:
• Timer expires, and
• PR/SM decides to redispatch CF2

Dynamic CF dispatching - Thin interrupt



PR/SM selects next logical ICF to dispatch onto physical ICF

DYNDISP=THININTERRUPT
(New with zEC12 server)

PR/SM dispatches LP0 on P0

Default time slice duration

PR/SM still takes engine away if CF consumes time slice

Work enters CF1, Thin Interrupt issued and PR/SM dispatches LP0 on P0 when P0 is available

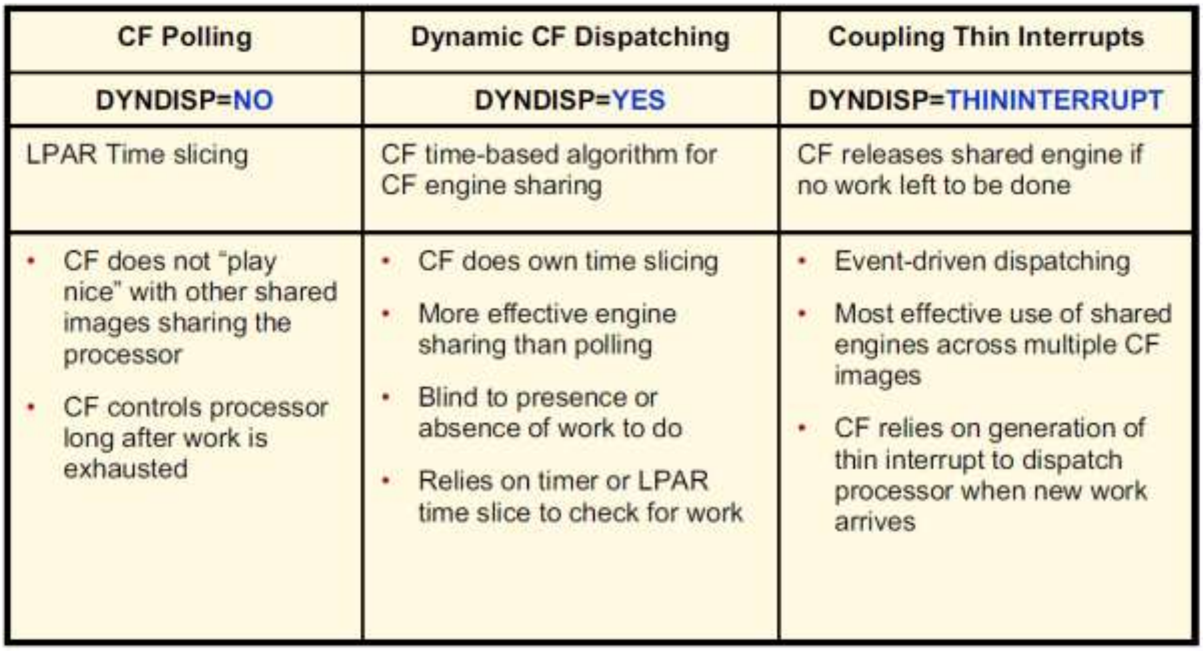
CF1 enters voluntary wait P0 is released until next interrupt or dispatch

CF2 enters voluntary wait P0 is released until next interrupt or dispatch

CF2 enters voluntary wait P0 is released until next interrupt or dispatch

Work enters CF2, Thin Interrupt issued and PR/SM dispatches LP1 on P0 when P0 is available

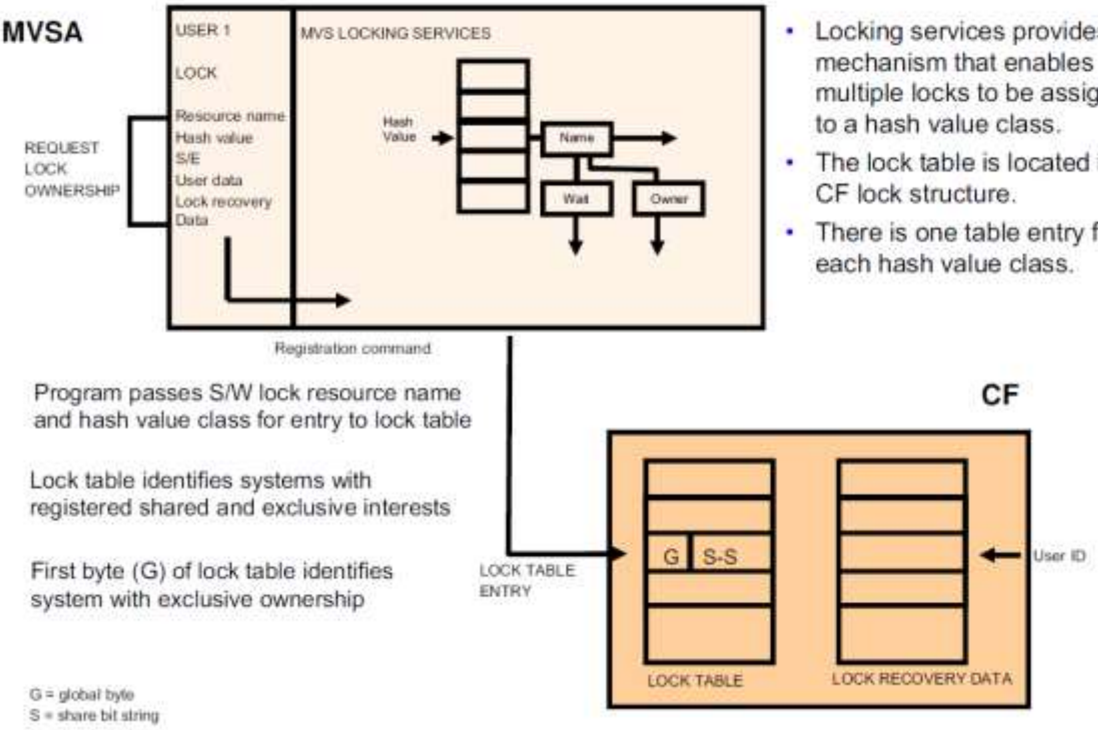
Dynamic CF dispatching – Summary



CF Polling	Dynamic CF Dispatching	Coupling Thin Interrupts
DYNDISP=NO	DYNDISP=YES	DYNDISP=THININTERRUPT
LPAR Time slicing	CF time-based algorithm for CF engine sharing	CF releases shared engine if no work left to be done
• CF does not "play nice" with other shared images sharing the processor • CF controls processor long after work is exhausted	• CF does own time slicing • More effective engine sharing than polling • Blind to presence or absence of work to do • Relies on timer or LPAR time slice to check for work	• Event-driven dispatching • Most effective use of shared engines across multiple CF images • CF relies on generation of thin interrupt to dispatch processor when new work arrives

Lock Structure

Manages shared and exclusive locks that can be used by DB managers



- Locking services provides a mechanism that enables multiple locks to be assigned to a hash value class.
- The lock table is located in the CF lock structure.
- There is one table entry for each hash value class.

Program passes S/W lock resource name and hash value class for entry to lock table

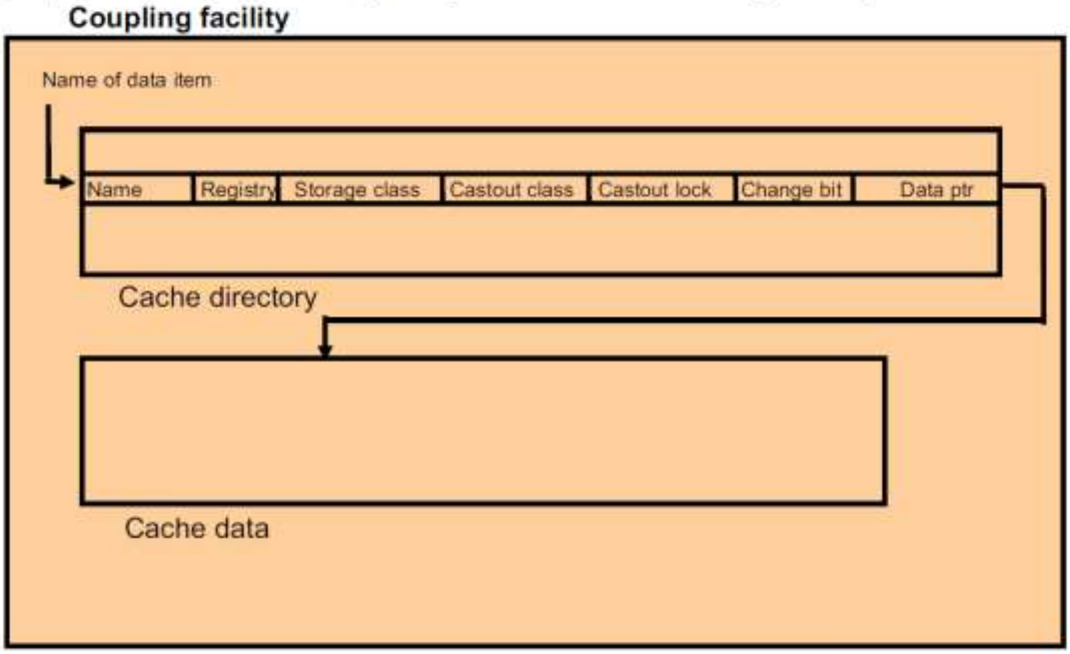
Lock table identifies systems with registered shared and exclusive interests

First byte (G) of lock table identifies system with exclusive ownership

G = global byte
S = share bit string

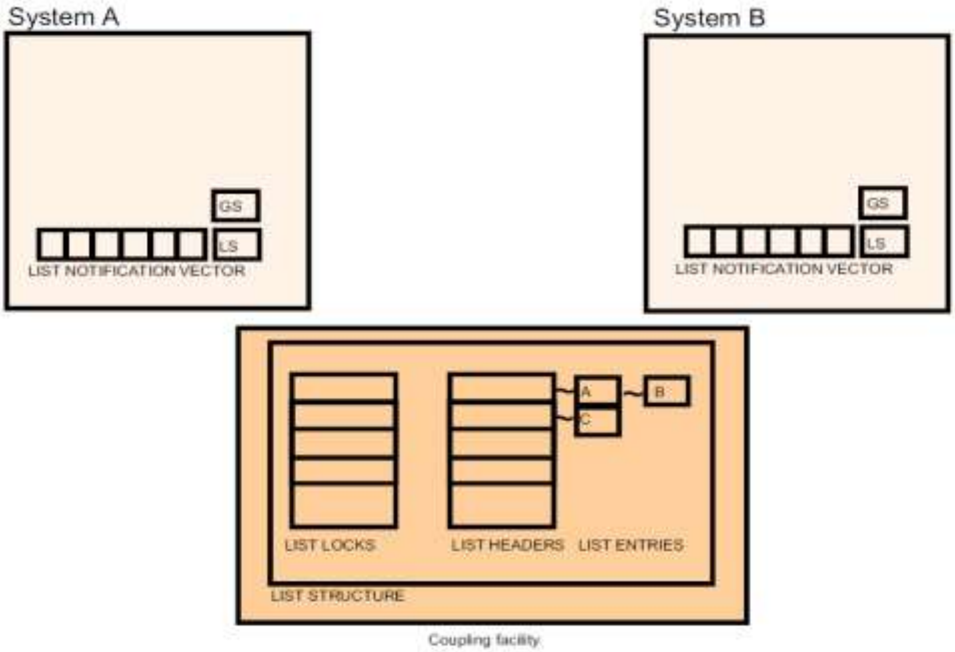
Cache Structure

- High performance for frequently referenced data by multiple users



List Structure

- List structures support general-purpose multisystem queuing constructs.



Mainframe zOS - Datasets classification

Many a times zOS Sysprogs, new to zOS, get confused with different types of the datasets classification. To them, they define the datasets (mostly physical sequential, partitioned, VSAM) using the standard utilities. But, they come across different classifications of the same datasets while working with or talking to other SME's like storage, security, application developers and others. I feel, while building the new zOS Sysprogs, we should familiarize them with these classifications to avoid confusion and have a better work experience.

In different context, all the datasets in the system could be classified as follows. There could be more to this.

1. **Cataloged** or **non-cataloged** datasets - all the datasets defined in the systems are either cataloged or non-cataloged.

2. **VSAM** or **non-VSAM** datasets - all the datasets defined in the systems are either VSAM or non-VSAM.

3. **SMS managed** or **non-SMS managed** datasets - all the datasets in the system are either SMS managed or non-SMS managed. This is a very important view from storage management perspective.

4. **DASD** or **Tape** datasets - this classification is based on the physical location of the dataset.

5. **Temporary** or **Permanent** dataset - this is an old classification and very rarely used by some people.

6. **Protected** or **non-protected** datasets - by security product e.g. RACF - this is a view of datasets by the security administrators.

7. **zOS datasets** or **Unix files** - this classification is viewed while working with USS.

8. **Systems datasets** or **User/Application** datasets - this is a very high level view to differentiate system datasets from others.

IO Processing

We always are told that, the IO (Input and Output) in the system is expensive. But why? Here is a summary of how IO is done in the system and after reading this, you will understand why the IO processing is so expensive. As I remember, each IO in the system execute more than 6000 instructions. That to, in a large installation, when we do around 40000 to 50000 IOs per second, how much time and resources we spend for doing IOs.

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 z 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, etc.).

3. To request the movement of data, either the access method or the user program presents information about the operation to the 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). Media Manager (represented by an MM in the diagram) is the I/O driver of the VSAM access method. NOTE: IMS and DB2 uses the Media Manager under covers.

4. If there are no pending I/O operations for the required device (from this system), 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. One of the System Assist Processors (SAP) processes the SSCH instruction and selects a channel path to initiate the I/O operation. This channel executes the channel program, controlling the movement of data between device, control unit, and processor storage.

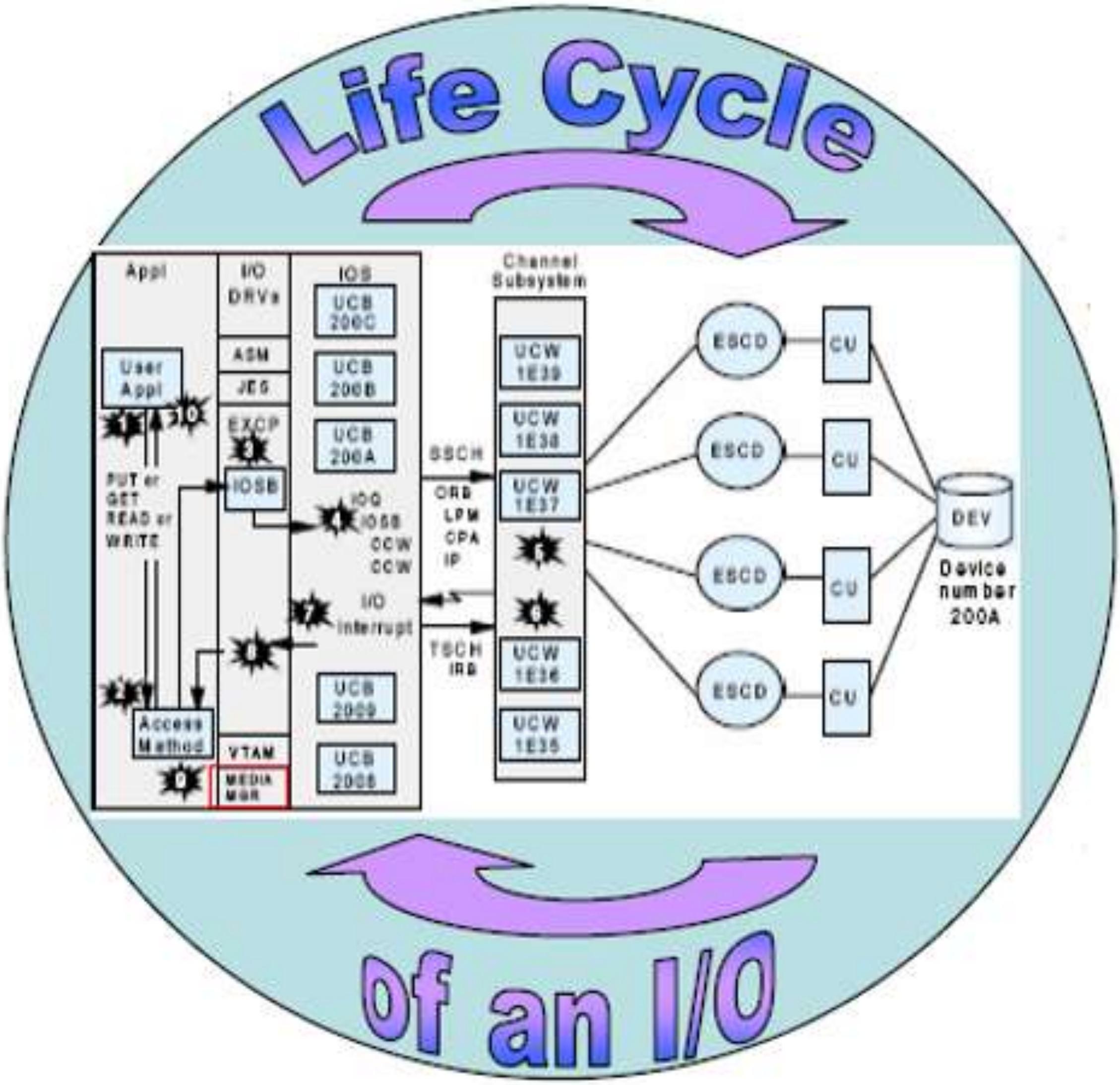
6. When the I/O operation is complete, the channel subsystem signals completion by generating an I/O interrupt.

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 indicates that I/O is complete by posting the access method and calling the dispatcher.

9. When appropriate, the dispatcher reactivates the access method. The access method returns control to the user program (through a branch).

10. The user program continues its processing at its next instruction point.


Life Cycle
Appl
I/O DRVs
IOS
UCB 200C
UCB 200B
UCB 200A
User Appl
ASM
JES
EXCP
IOSB
1
10
3
PUT or GET READ or WRITE
SSCH
ORB
LPM
CPA
IP
4
IOQ
IOSB
CCW
CCW
7
I/O Interrupt
8
TSCH
IRB
2
Access Method
9
VTAM
MEDIA MGR
UCB 2009
UCB 2008
Channel Subsystem
UCW 1E39
UCW 1E38
UCW 1E37
5
6
UCW 1E36
UCW 1E35
ESCD
ESCD
ESCD
ESCD
CU
CU
CU
CU
DEV
Device number 200A
of an I/O

DASD Storage – Cylinder and Track

The DASD under zOS is organized as Cylinder and Tracks.

Cylinder

A disk drive contains *cylinders*. A cylinder is a unit of storage on a count-key-data (CKD) device with 15 number of tracks.

Track

Cylinders contain *tracks*, which are circular paths on the surface of a disk or diskette on which information is magnetically recorded and from which recorded information is read. Tracks are in count-key-data (CKD) format, which means that each track contains fields that indicate the start of the track and the space used, followed by records containing three fields:

- The count field defines the length of the record
- The key field contains optional accounting information
- The data field contains the user data

In 3390 series of DASD, 1 Track = 56664 bytes.

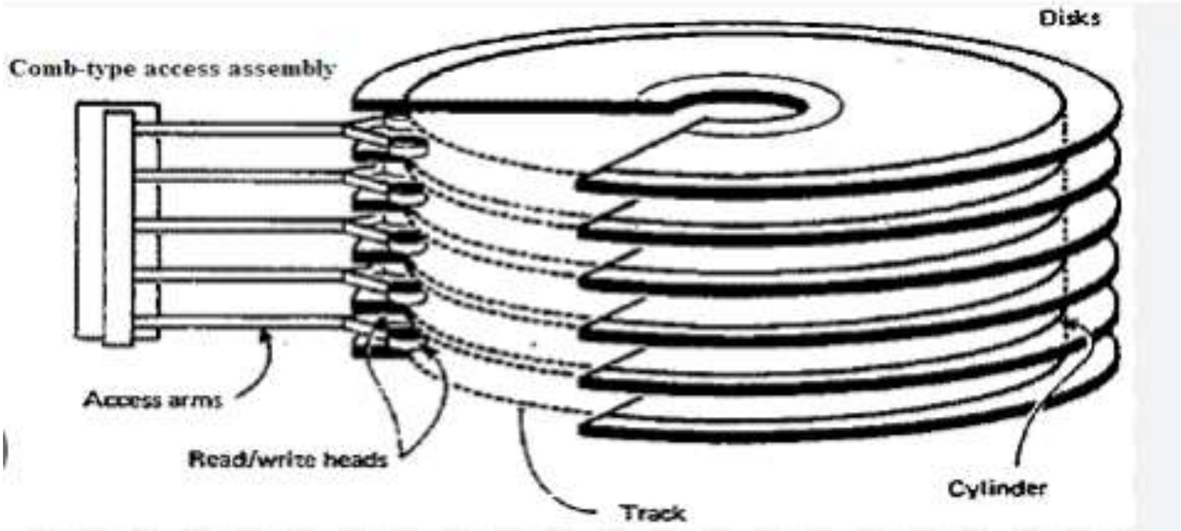
The definition I learnt

I started learning about 3380 DASD storage. It had 9 disk platters mounted in the same axis and rotating together and a read/write head unit with 16 read/write heads moves inside the platters to read/write the data in the disk magnetic surface. Data was being written in both surface of the platter except the top and bottom surface. In this way, 9 platters provided data written in 16 surfaces.

Track – Keeping the read/write head unit fixed, when the platters take one turn, the amount of data read by one read/write head is a track.

Cylinder – Keeping the read/write head unit fixed, when the platters take one turn, the amount of data read by all read/write heads is a cylinder.

Note: There are 16 read/write heads reading 16 tracks of data from 16 surfaces. One track was control information and 15 tracks contained user data. That is how the definition of one cylinder consisting of 15 tracks of data.



3390 DASD Block Size

A nice table which I used to refer for optimizing the allocation of space in 3390 DASD. You will hear the word called ‘Optimum Block Size’ which allows minimal DASD storage usage.

A block size is called the ‘Physical’ record in the DASD storage.

For example, if your logical record size, used in the program, is 500 bytes length and you have defined a sequential file of block size 5000 bytes (10 logical records stored in one physical record), then one track of 3390 DASD will store; 10 physical records (blocks) in one track, 150 physical records in one cylinder and the effective storage usage will be 89.4% (10.6% of storage will be wasted in track/cylinder).

Please note that, when the IO is done to the DASD device, a minimum one block of data is fetched.

Data Length Range		Percent	Maximum Track		Maximum Cylinder	
		Space	Capacity *		Capacity *	
Min	Max	Used *	Records	Bytes	Records	Bytes
27 999	56 664	100.0	1	56 664	15	849 960
18 453	27 998	98.8	2	55 996	30	839 940
13 683	18 452	97.7	3	55 356	45	830 340
10 797	13 682	96.6	4	54 728	60	820 920
8 907	10 796	95.3	5	53 980	75	809 700
7 549	8 906	94.3	6	53 436	90	801 540
6 519	7 548	93.2	7	52 836	105	792 540
5 727	6 518	92.0	8	52 144	120	782 160
5 065	5 726	90.9	9	51 534	135	773 010
4 567	**5 064**	**89.4**	**10**	**50 640**	**150**	**759 600**
4 137	4 566	88.6	11	50 226	165	753 390
3 769	4 136	87.6	12	49 632	180	744 480
3 441	3 768	86.4	13	48 984	195	734 760
3 175	3 440	85.0	14	48 160	210	722 400
2 943	3 174	84.0	15	47 610	225	714 150
2 711	2 942	83.1	16	47 072	240	706 080
2 547	2 710	81.3	17	46 070	255	691 050
2 377	2 546	80.9	18	45 828	270	687 420
2 213	2 376	79.7	19	45 144	285	677 160
2 083	2 212	78.1	20	44 240	300	663 600
1 947	2 082	77.2	21	43 722	315	655 830
1 851	1 946	75.6	22	42 812	330	642 180
1 749	1 850	75.1	23	42 550	345	638 250
1 647	1 748	74.0	24	41 952	360	629 280
1 551	1 646	72.6	25	41 150	375	617 250

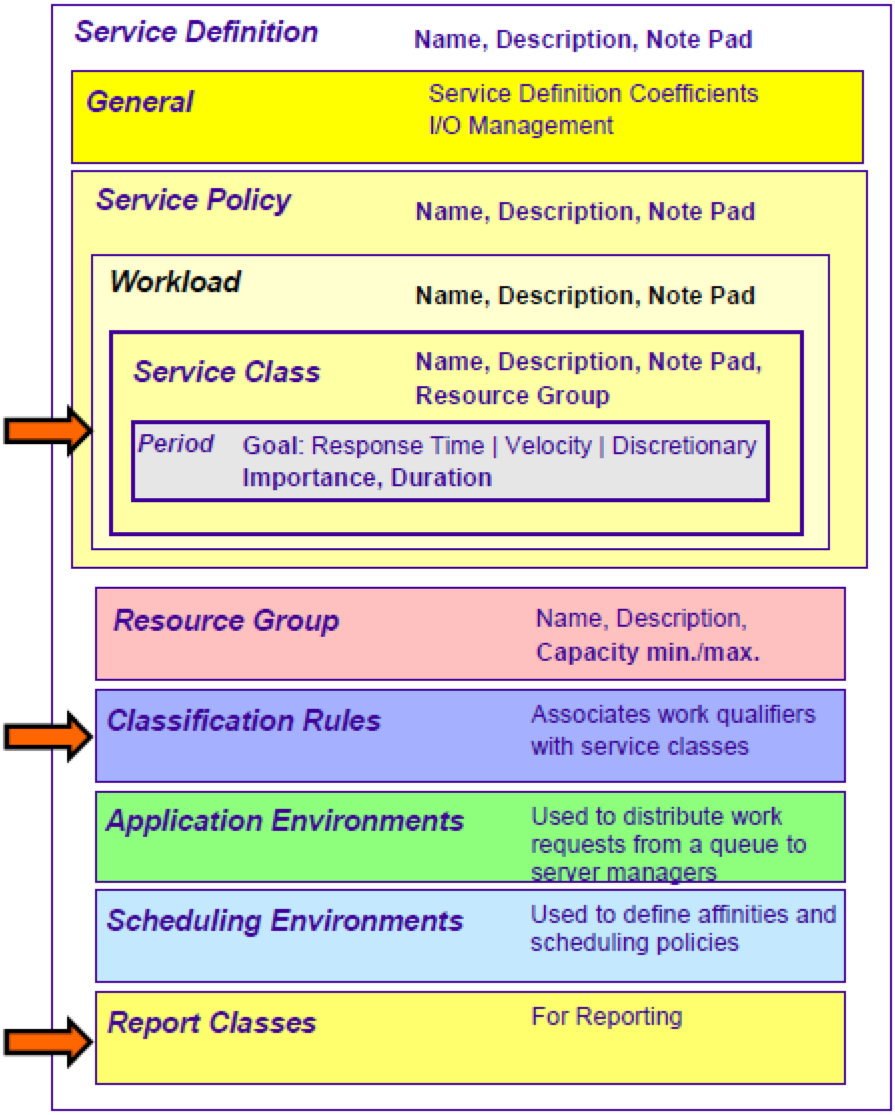
```
|  1 483     1 550  |   71.1   |    26      40 300 |    390   604 500 |
|  1 387     1 482  |   70.6   |    27      40 014 |    405   600 210 |
|  1 319     1 386  |   68.5   |    28      38 808 |    420   582 120 |
|  1 251     1 318  |   67.5   |    29      38 222 |    435   573 330 |
|  1 183     1 250  |   66.2   |    30      37 500 |    450   562 500 |
|  1 155     1 182  |   64.7   |    31      36 642 |    465   549 630 |
|  1 087     1 154  |   65.2   |    32      36 928 |    480   553 920 |
|  1 019     1 086  |   63.2   |    33      35 838 |    495   537 570 |
|    985     1 018  |   61.1   |    34      34 612 |    510   519 180 |
|    951       984  |   60.8   |    35      34 440 |    525   516 600 |
|    889       950  |   60.4   |    36      34 200 |    540   513 000 |
|    855       888  |   58.0   |    37      32 856 |    555   492 840 |
|    821       854  |   57.3   |    38      32 452 |    570   486 780 |
|    787       820  |   56.4   |    39      31 980 |    585   479 700 |
|    753       786  |   55.5   |    40      31 440 |    600   471 600 |
|    719       752  |   54.4   |    41      30 832 |    615   462 480 |
|    691       718  |   53.2   |    42      30 156 |    630   452 340 |
|    657       690  |   52.4   |    43      29 670 |    645   445 050 |
|    623       656  |   50.9   |    44      28 864 |    660   432 960 |
|    589       622  |   49.4   |    45      27 990 |    675   419 850 |
|    555       588  |   47.7   |    46      27 048 |    690   405 720 |
|    521       554  |   46.9   |    48      26 592 |    720   398 880 |
|    487       520  |   45.0   |    49      25 480 |    735   382 200 |
|    459       486  |   42.9   |    50      24 300 |    750   364 500 |
|    425       458  |   42.0   |    52      23 816 |    780   357 240 |
|    391       424  |   40.4   |    54      22 896 |    810   343 440 |
|    357       390  |   37.9   |    55      21 450 |    825   321 750 |
|    323       356  |   35.8   |    57      20 292 |    855   304 380 |
|    289       322  |   33.5   |    59      18 998 |    885   284 970 |
|    255       288  |   31.0   |    61      17 568 |    915   263 520 |
|    227       254  |   28.7   |    64      16 256 |    960   243 840 |
|    193       226  |   26.3   |    66      14 916 |    990   223 740 |
|    159       192  |   23.4   |    69      13 248 |   1035   198 720 |
|    125       158  |   20.1   |    72      11 376 |   1080   170 640 |
|     91       124  |   16.4   |    75       9 300 |   1125   139 500 |
|     57        90  |   12.4   |    78       7 020 |   1170   105 300 |
|     23        56  |    8.1   |    82       4 592 |   1230    68 880 |
|      1        22  |    3.3   |    86       1 892 |   1290    28 380 |
|___________________|__________|__________________|__________________|
* Calculations are made using maximum size record in range.
```

WLM

The Workload Manager component of the z/OS system (referred to as WLM) monitors a system/sysplex and determines how much resource should be given to each item of work in the system/sysplex to meet the goals that you have defined for it.

WLM Definition

This one diagram describes all about the WLM Service definition.


Service Definition
Name, Description, Note Pad
General
Service Definition Coefficients
I/O Management
Service Policy
Name, Description, Note Pad
Workload
Name, Description, Note Pad
Service Class
Name, Description, Note Pad, Resource Group
Period
Goal: Response Time | Velocity | Discretionary
Importance, Duration
Resource Group
Name, Description, Capacity min./max.
Classification Rules
Associates work qualifiers with service classes
Application Environments
Used to distribute work requests from a queue to server managers
Scheduling Environments
Used to define affinities and scheduling policies
Report Classes
For Reporting

WLM Goals

Velocity

- Measure of acceptable delays while work is capable of running
  - Delays measured for CPU and storage, optionally for I/O and JES Queue time
- Used for long running and non-interactive workloads
- High velocity can be interpreted as:
  - "When this work is ready to run, make sure it runs without delay"
- Low velocity can be interpreted as:
  - "When this work is ready to run, keep it plodding along to ensure it will eventually finish"
- Velocity is not equal to dispatch priority

$$\text{ExVel} = \frac{\text{Using Samples}}{\text{Using Samples} + \text{Delay Samples}} \bullet 100$$

Percentile Response Time

- Percentile of work to be completed in the specified amount of time
- Best goal type able to handle variability in transaction response times
- Boundaries are from 1 to 99
- Response times can range from 15 ms to 24 hours
- WLM will not delay work, or limit it, to achieve the response time goal when extra processing time exists

Average Response Time

- Heavily influenced by the extremes of transaction distribution
- WLM will manage work based on worst behaving transactions
- When to use:
  - Current SLA's are defined in terms of averages
  - As a starting point when transaction information is unavailable

  - Once average response time goal is established, RMF will show on Workload Activity Report distribution of response times around the average

Note: A few long running transactions may very badly impact the overall goal of the workload.

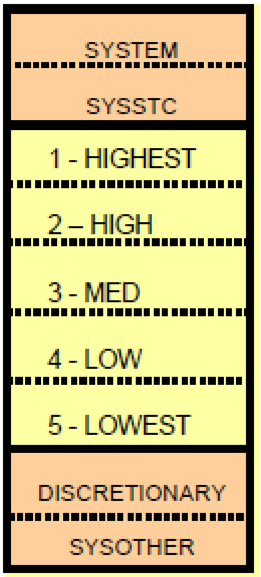
Discretionary Goal

- Runs when all work with non-discretionary goals are absent
  - OS/390 R2V6 will introduce changes to discretionary workloads. Work with goals which is over-achieving is capped, freed CPU now can be used by discretionary work. PI of capped work kept between .7 and .81
  - Only work with a R/T goal > 1 minute, or velocity <= 30
- Equivalent to work in lowest mean time to wait (MTTW) group in the IPS whose MPL fluctuates based on available capacity Only WLM goal type managed by SRM with the MTTW algorithm
- Work with this goal is a candidate for individual storage control via working set management

System Goals

- Used to handle some work by default
  - SYSTEM - Dispatch Priority of 255
  - SYSSTC - Dispatch Priority of 254
  - TSO - RT goal of 80% between 0 and 2 seconds ????
  - SYSOTHER - WLM Discretionary Goal

Service Class - Importance



Workload Importance

- **Most important factor for WLM how work is treated**
  - This means
    - Not all work is equally important
    - WLM works best if it can distinguish work by its importance
- **It's a good idea**
  - To understand how many transactions and/or CPU service are consumed by importance level
    - You may find out that certain importance levels are barely used
- **How to distinguish work**
  - System Importance: lock manager and monitors
    - This is work which should always run
  - Importance 1: most important started tasks

        - For example: DB2 regions
        - IT might be a good idea not to place the production work at this level
    - Importance 2: most important production work
        - CICS, IMS transactions, 1st period of short running transactions (TSO, DDF, …) etc…
    - Importance 3: production applications
    - Importance 4,5: TSO, DDF lower periods, batch, test, etc …
    - Discretionary: It's a good idea to use it but not mandatory

Note: There are significant changes and improvements to WLM today. Please refer to WLM manuals for details.

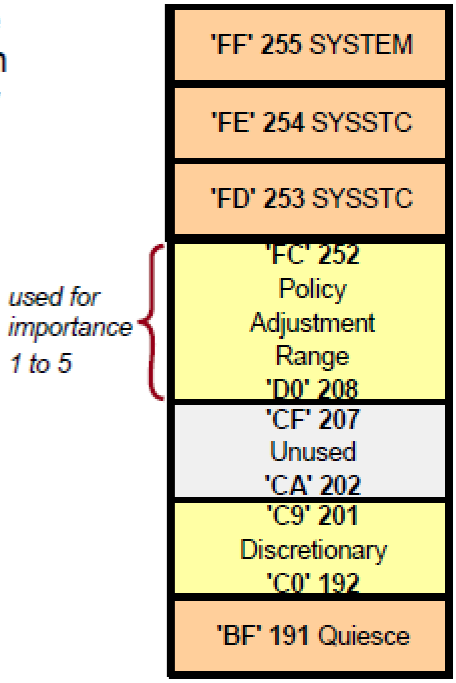
CPU Critical

- Protects work against lower important service classes
    - Lower important service classes will never get an equal or higher DP than CPU Critical service class
    - There are still WLM adjustments with equal or higher important work
- The positive
    - It is sometimes useful to protect importance 1 work (special regions) against lower important service classes and thus reducing the amount of assessments WLM considers
- The negative
    - If it is done too much and for too many importance levels the service policy gets static and WLM does not do many adjustments anymore
        - Can result in less efficient resource utilization

System Goals and Dispatching Priority

WLM-defined Goal Types that are automatically assigned to certain types of workload recognized by WLM.

- SYSTEM
  - z/OS system address spaces created at IPL.
  - Highest dispatching priority.
- SYSSTC
  - You assign important STCs
  - Second highest dispatching priority.
- SYSOTHER
  - Catcher for forgotten subsystem definitions.
  - Same as discretionary. Lowest dispatching priority.



'FF' 255 SYSTEM

'FE' 254 SYSSTC

'FD' 253 SYSSTC

'FC' 252
Policy
Adjustment
Range
'D0' 208

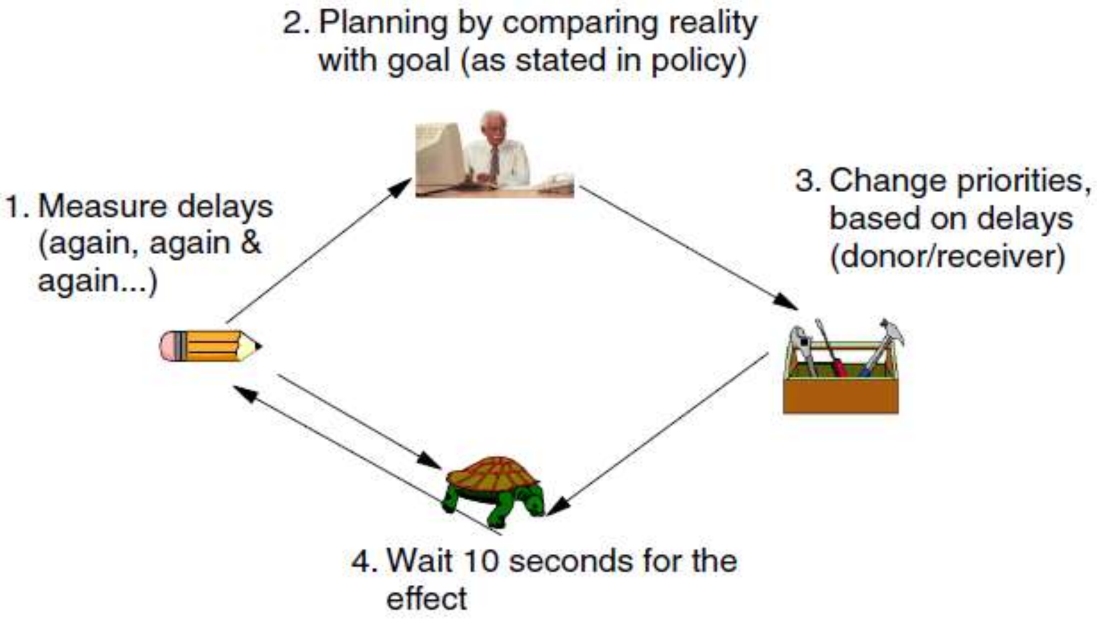
'CF' 207
Unused
'CA' 202

'C9' 201
Discretionary
'C0' 192

'BF' 191 Quiesce

used for importance 1 to 5

WLM takes decision every 10 sec



2. Planning by comparing reality with goal (as stated in policy)

1. Measure delays (again, again & again...)

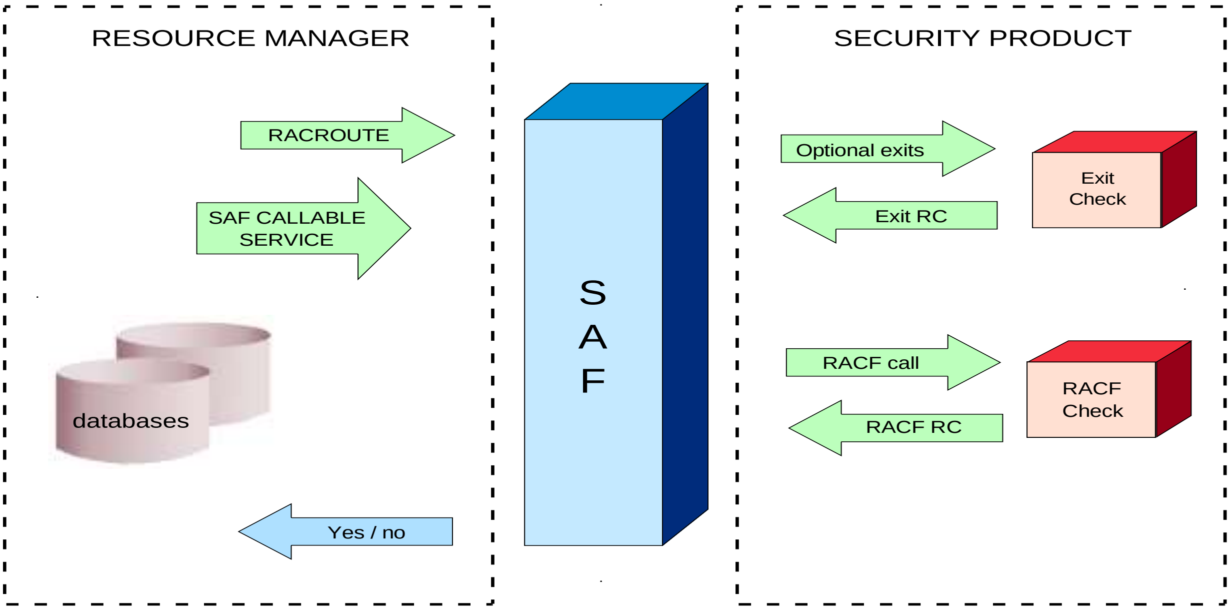
3. Change priorities, based on delays (donor/receiver)

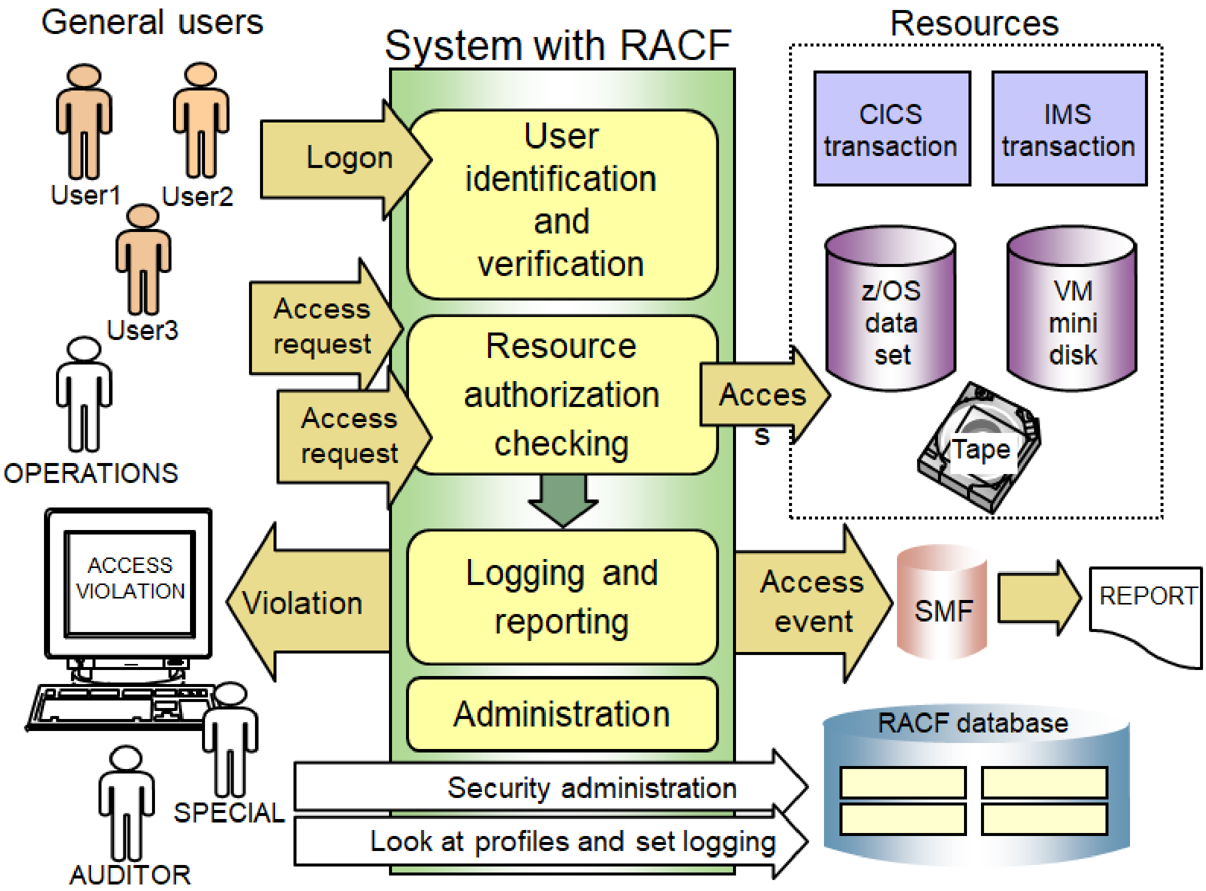
4. Wait 10 seconds for the effect

RACF

SAF provides an installation with centralized control over system security processing by using a system service called the SAF router. The SAF router provides a focal point and a common system interface for all products providing resource control. The resource-managing components and subsystems call the SAF router as part of certain decision-making functions in their processing, such as access control checking and authorization-related checking. These functions are called control points. This single SAF interface encourages the use of common control functions shared across products and across systems.

The SAF router is always present on a z/OS system whether or not RACF is present. If RACF is available in the system, the SAF router might pass control to the RACF router (ICHRFR00). The RACF router in turn invokes the appropriate RACF function, based on parameter information and the RACF router table. The RACF router table, consisting of modules ICHRFR0X and ICHRFR01, associates router invocations with RACF function. If your installation decides not to call RACF, you must code the SAF router exits appropriately. For more information, see External Security Interface (RACROUTE) Macro Reference for z/OS.





RACF has four major functions.

The first of these is the **identification and verification of users**.

- Is user defined to RACF?
- Is the password valid?
- Is the user a member of a RACF group?
- Has the user been suspended (revoked)?
- Is user authorized for that terminal?
- Is user authorized for that application?
- Is user authorized to log on at this time of day or day of week?
- Can this terminal be logged on to at this time of day or day of week?

If user identification and verification is successful, RACF specifies (in a control block called the *accessor environment element* or ACEE) the scope of the user's authorization for the current terminal session or batch job.

The second of RACF's prime tasks is **resource authorization checking**.

- When a user attempts to access a specific resource, RACF is called to determine whether this user should be allowed to access the resource.
- RACF can be used to control access to many types of resources, such as z/OS data sets, CICS and IMS transactions, and VM minidisks.
- When a user attempts to access a resource, RACF is called to determine whether this user should be allowed to access the resource.
- RACF refers to a resource profile to determine access.

The third major function RACF performs is **logging and reporting** of attempts to access resources.

- After RACF has made the decision to allow or deny access to the requested resource, it checks to see whether any logging was requested. Auditing can be specified by the security administrator or the auditor.
- If logging was requested, then the access event is written to SMF.
- Additionally, a message is sent to the system console when a violation occurs. Optionally, a notification can be sent to a specific user.
- The SMF data can be processed to produce reports.

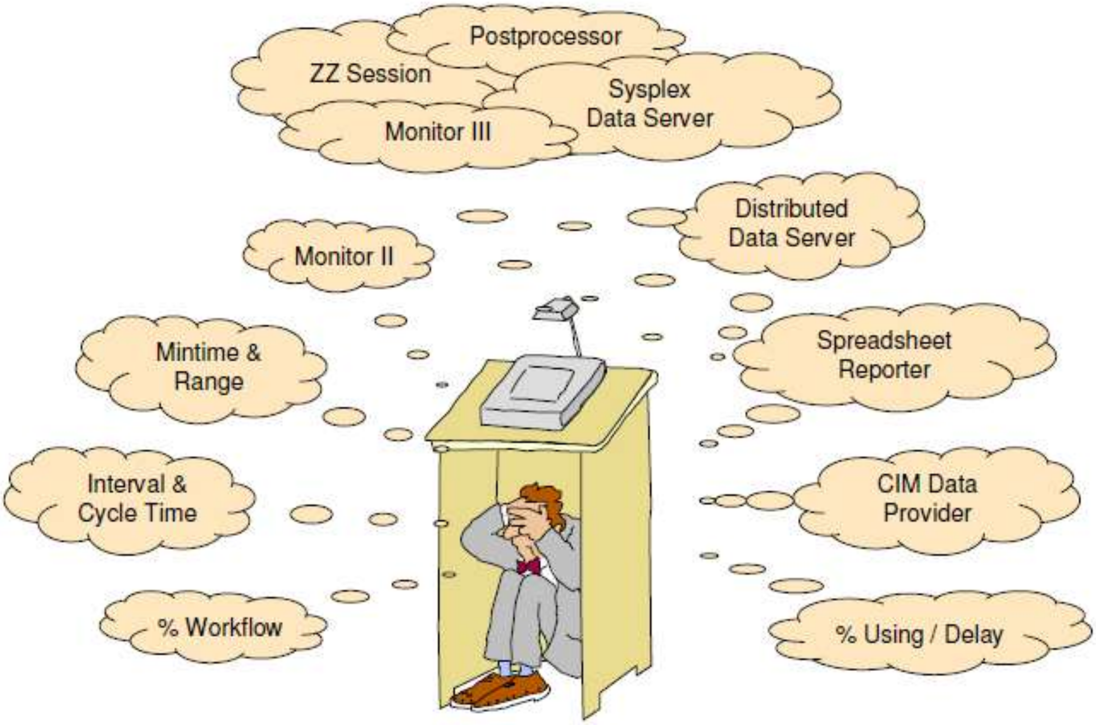
The fourth function of RACF is **security administration**.

- The user with the RACF SPECIAL attribute is able to administer security for the system. This is done by defining, changing, listing, and deleting the various profiles in the RACF database.

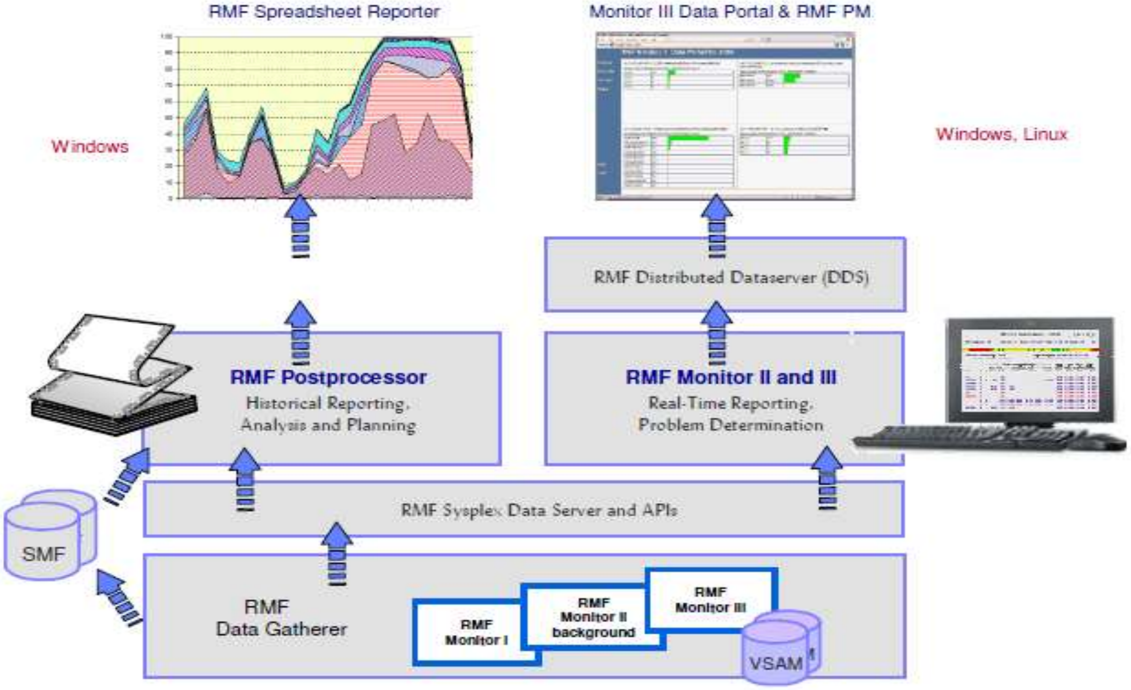
- The SPECIAL attribute does not give access to the resources. Rather, the SPECIAL attribute allows the user to administer the profiles in the RACF database.

RMF

The RMF product consists of several components for all kinds of performance related disciplines and the following picture shows the main RMF components.



RMF Product



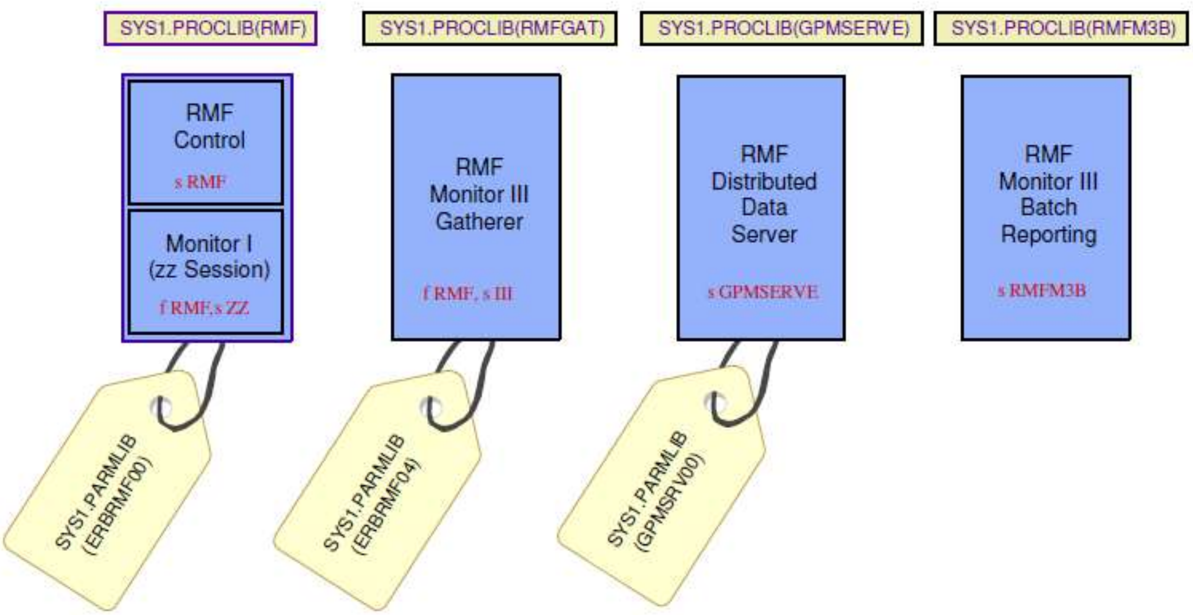
z/OS Resource Measurement Facility (RMF) is an optional priced feature of z/OS. It supports installations in performance analysis, capacity planning, and problem determination. For these disciplines, different kinds of data collectors are needed:

- **Monitor I** long term data collector for all types of resources and workloads. The SMF data collected by Monitor I is mostly used for capacity planning and performance analysis
- **Monitor II** snap shot data collector for address space states and resource usage. A subset of Monitor II data is also displayed by the IBM SDSF product
- **Monitor III** short-term data collector for problem determination, workflow delay monitoring and goal attainment supervision. This data is also used by the RMF PM Java Client and the RMF Monitor III Data Portal

Data collected by all three gatherers can be saved persistently for later reporting (SMF records or Monitor III VSAM datasets)

While Monitor II and Monitor III are real-time reporters, the RMF Postprocessor is the historical reporting function for Monitor I data

RMF Address Spaces

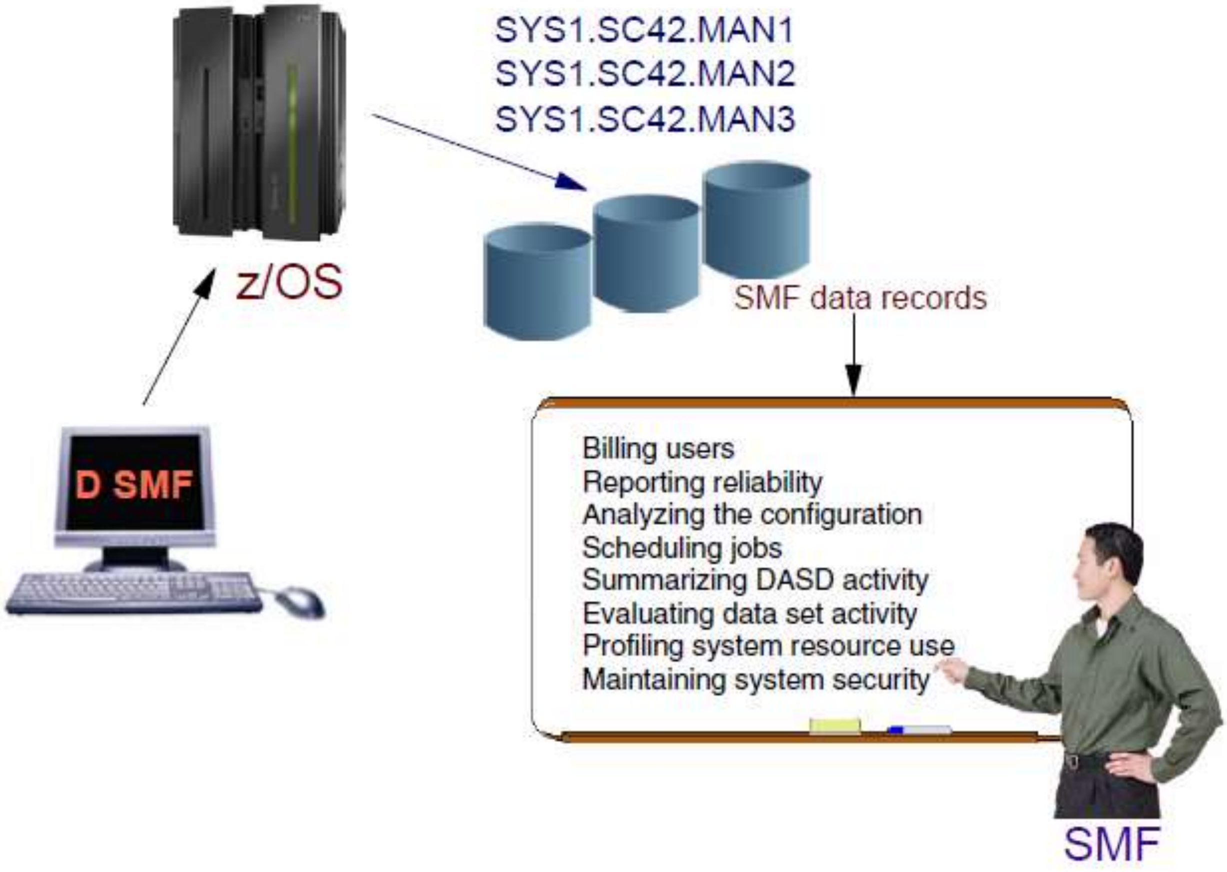


In order to make all RMF functions working, a couple of address spaces have to be activated:

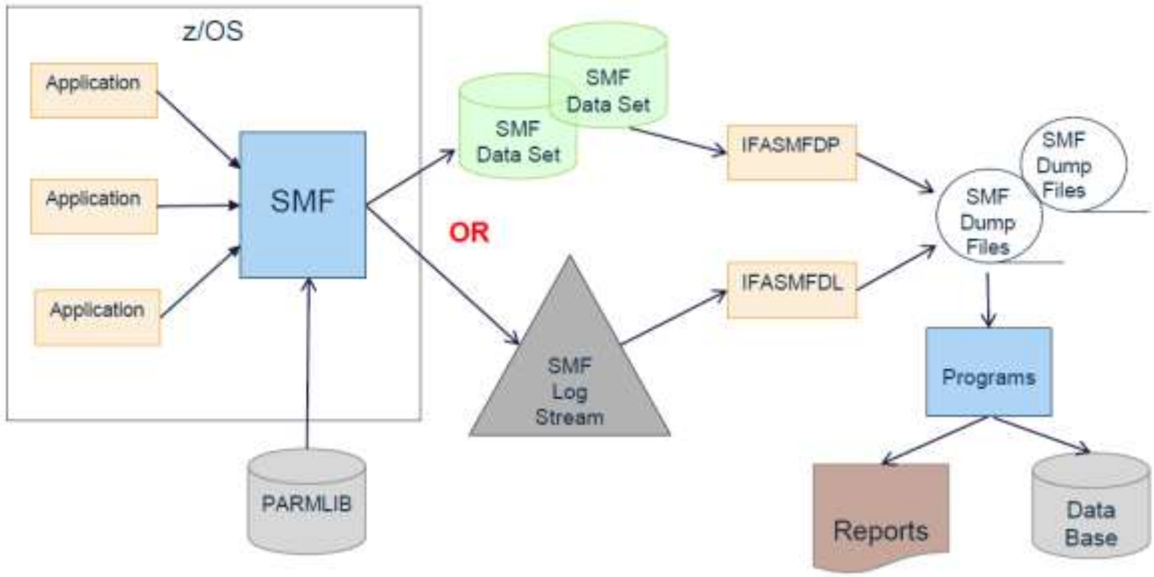
- RMF is the root address space. It is required for any further activities
- The Monitor I gatherer is a subtask in the RMF address space. It is started immediately together with RMF (by default) or later by means of the modify command
- RMFGAT is the Monitor III data gatherer. It cannot be started as standalone address space. Like Monitor I, it can only be activated with the modify command. Within a sysplex, one instance per system is needed for RMFGAT
- The RMF Distributed Data Server is the data source for the RMF Performance Monitoring java client and the Monitor III Data Portal. Only one instance is needed per sysplex
- RMFM3B is needed for the generation of console messages in case a certain threshold is exceeded

SMF

The z/OS system collects statistical data for each task when certain events occur in the life of the task. The System Management Facility (SMF) formats the information that it gathers into system-related (or job-related) records. System-related SMF records include information about the configuration, paging activity, and workload. Job-related records include information about the CPU time, SYSOUT activity, and data set activity of each job step, job, APPC/MVS transaction program, and TSO/E session. SMF data is written to the SYS1.MANx data sets.

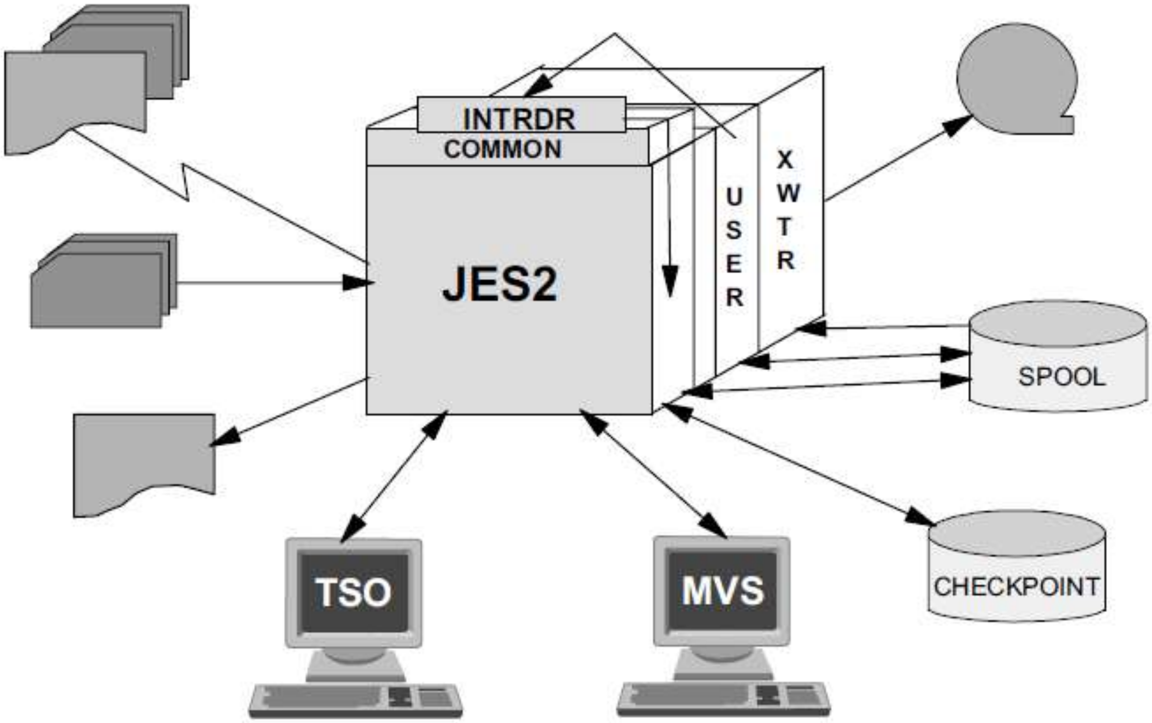

SYS1.SC42.MAN1
SYS1.SC42.MAN2
SYS1.SC42.MAN3
z/OS
SMF data records
D SMF
Billing users
Reporting reliability
Analyzing the configuration
Scheduling jobs
Summarizing DASD activity
Evaluating data set activity
Profiling system resource use
Maintaining system security
SMF

SMF Recording



JES2

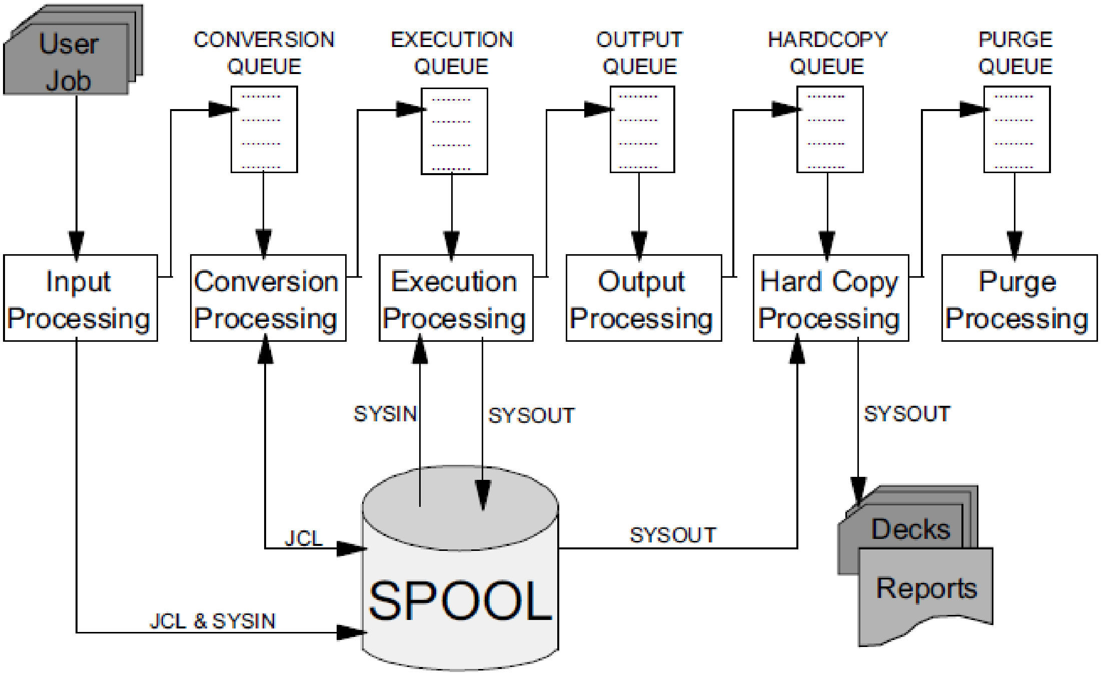
JES2 components



- The primary function of JES2 is to manage batch work.
- Jobs enter the system by JES readers. These readers may be local (channel attached) or remote (teleprocessing attached).
- JES provides support for punches, line printers, and page printers (3800, 3820). Punches and printers may be local or remote.
- TSO SUBMIT and OUTPUT commands are supported by JES. SUBMIT support uses the internal reader facility (INTRDR). The OUTPUT command uses the JES SYSOUT interface to select output data sets from SPOOL.
- JES has a job entry command structure beyond the MVS set of commands. JES commands are primarily related to managing job queues (input, execution, output).

- JES has Checkpoint data sets to checkpoint the status of job processing and SPOOL data.
- JES SPOOL contains JCL, SYSIN data, SYSOUT data, and job related JES control blocks.
- The external writer provides a facility to process SYSOUT data without using a JES writer. This facility runs as a started task in it's own address space and will retrieve SYSOUT data from SPOOL using a JES application programming interface (API).
- Typical uses of external writers are:
    - Writing SYSLOG data to tape.
    - Archiving special classes of SYSOUT data.

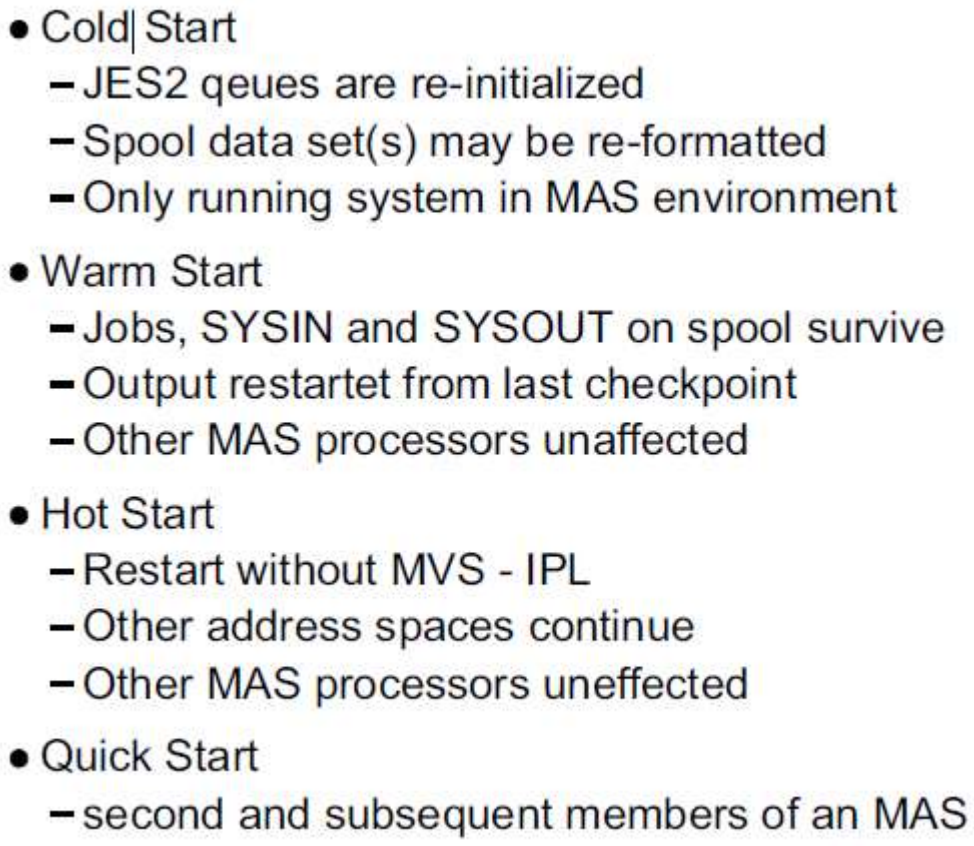
Job Processing



- The input phase spools the job's JCL and SYSIN data to SPOOL, processes JES user control statements, assigns a JES job number, and provides a default job class and job priority when not provided by the user.

- The conversion phase merges user JCL with JCL from the procedure libraries, syntax checks the JCL, converts the JCL to converter/interpreter (C/I) text, and queues the job for execution.
- JES execution phase supplies an initiator with a job based on the type of initiator request. If a batch initiator, JES will consider job class and JES job priority. For TSO LOGON or started tasks, the request is for a specific job.
- The output phase determines SYSOUT data set characteristics and groups a job's output data sets into groups with identical output characteristics.
- JES processing at the hardcopy phase matches the SYSOUT data set requirements with hardcopy devices (printers and punches) and prints or punches the data.
- Purge processing deletes the job from the system and creates SMF records to reflect the job's system resource usage.

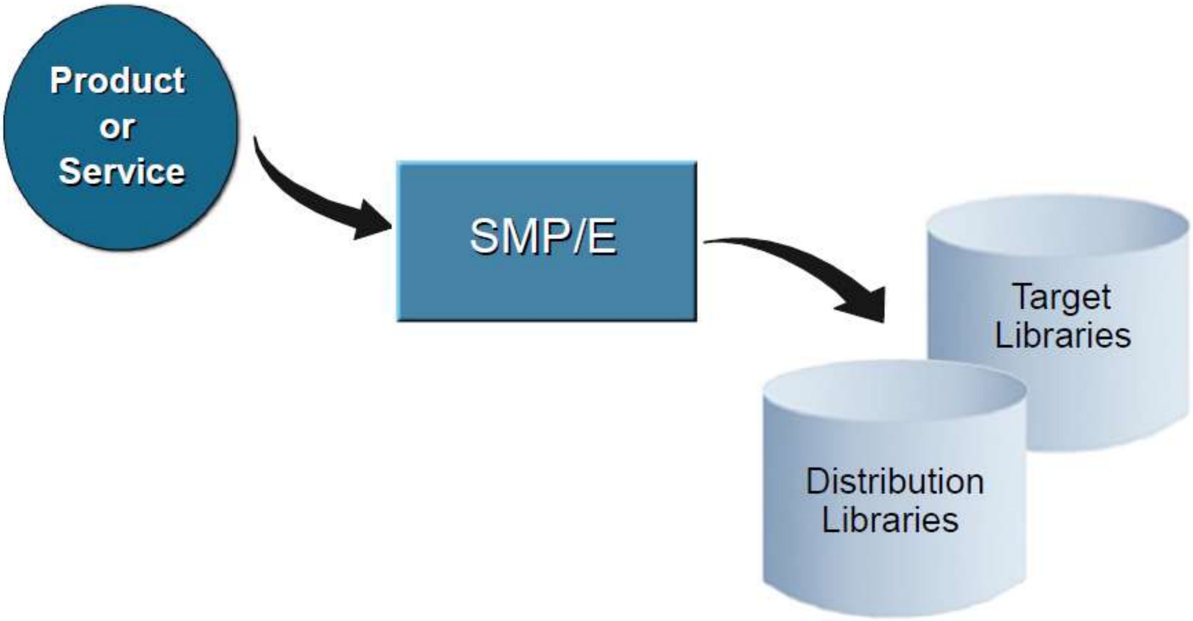
JES2 start types

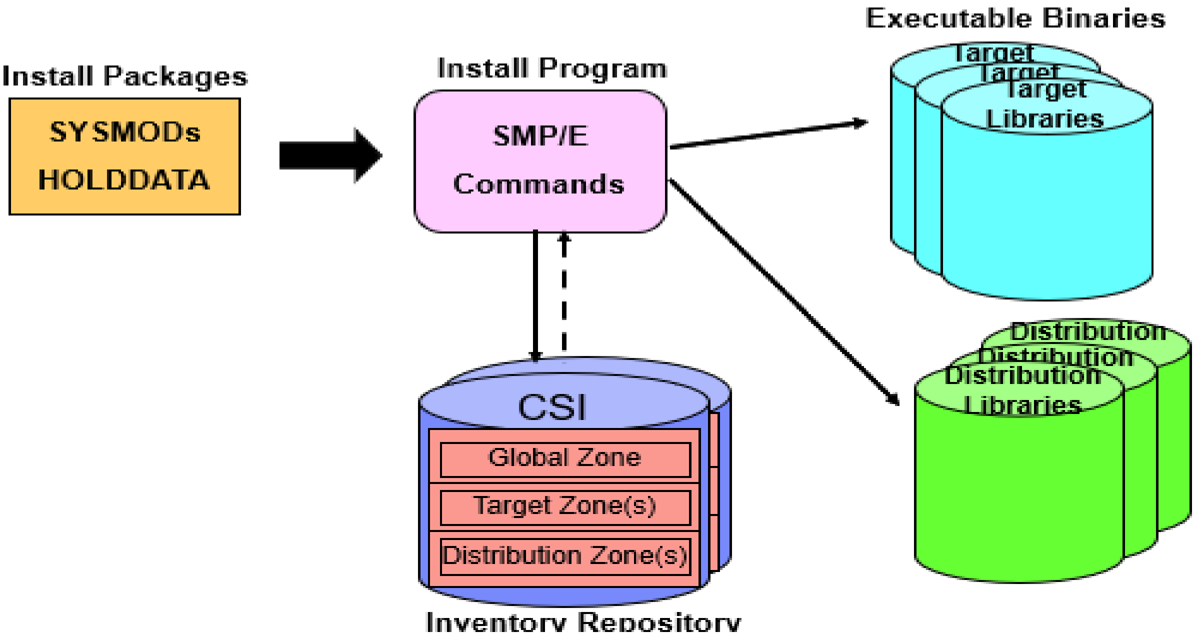


- Cold Start
  - JES2 qeues are re-initialized
  - Spool data set(s) may be re-formatted
  - Only running system in MAS environment
- Warm Start
  - Jobs, SYSIN and SYSOUT on spool survive
  - Output restartet from last checkpoint
  - Other MAS processors unaffected
- Hot Start
  - Restart without MVS - IPL
  - Other address spaces continue
  - Other MAS processors uneffected
- Quick Start
  - second and subsequent members of an MAS

SMPE

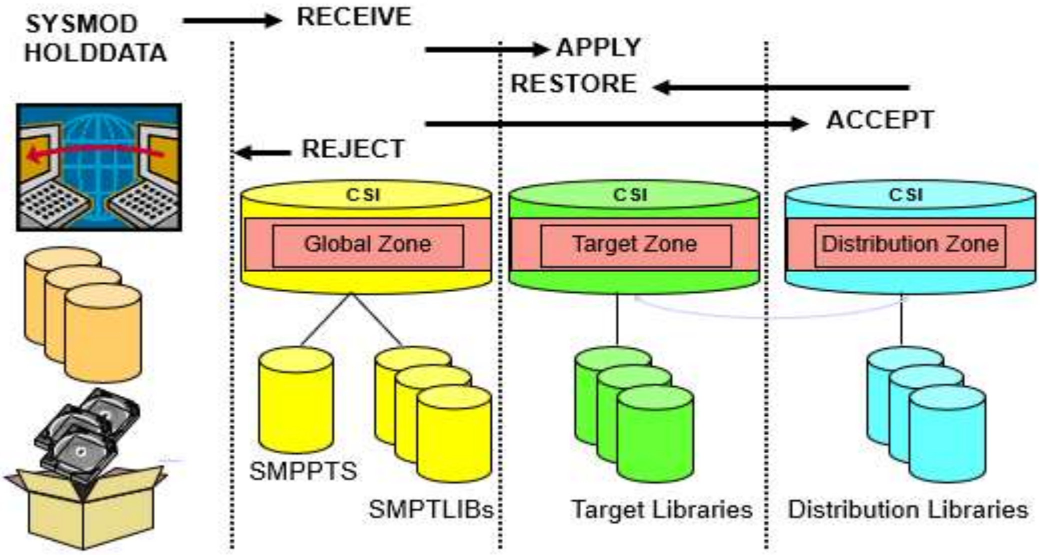
The following two pictures shows the SMPE activity.

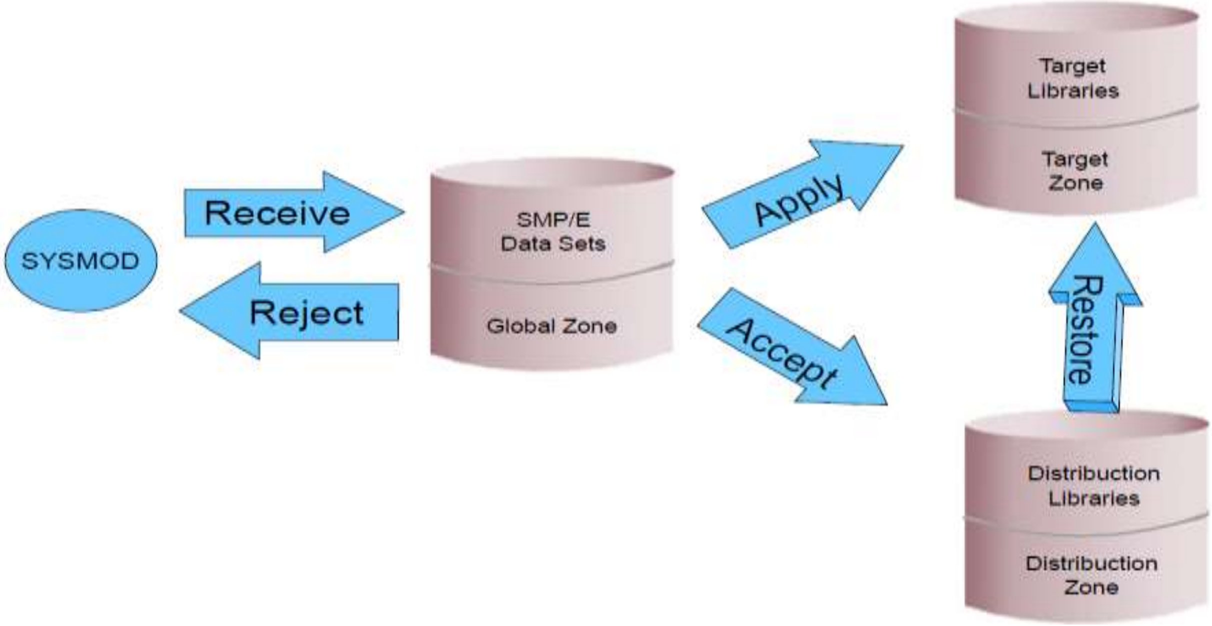




SMPE Processing

All these two pictures explain the same processing activities. The pictures are self-explanatory.



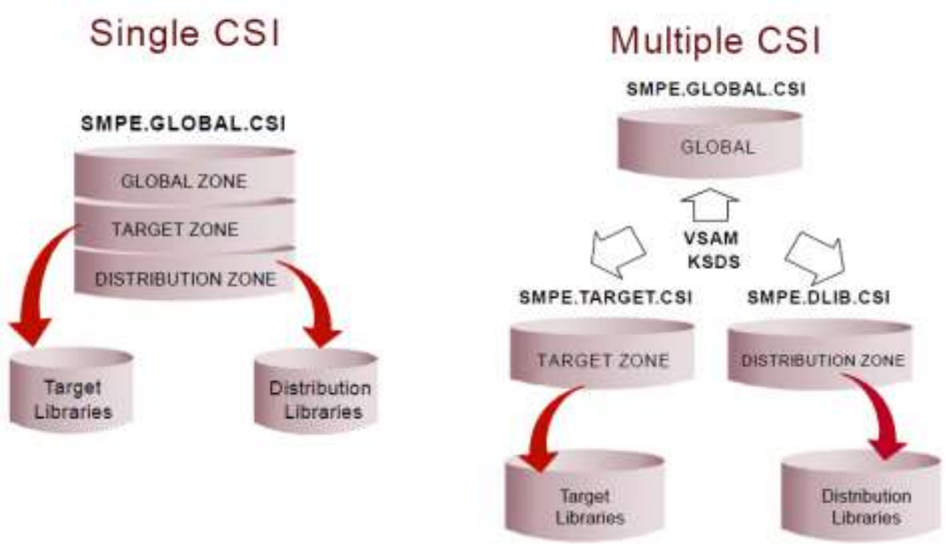


SYSMODs

There are four types of SYSMODs:

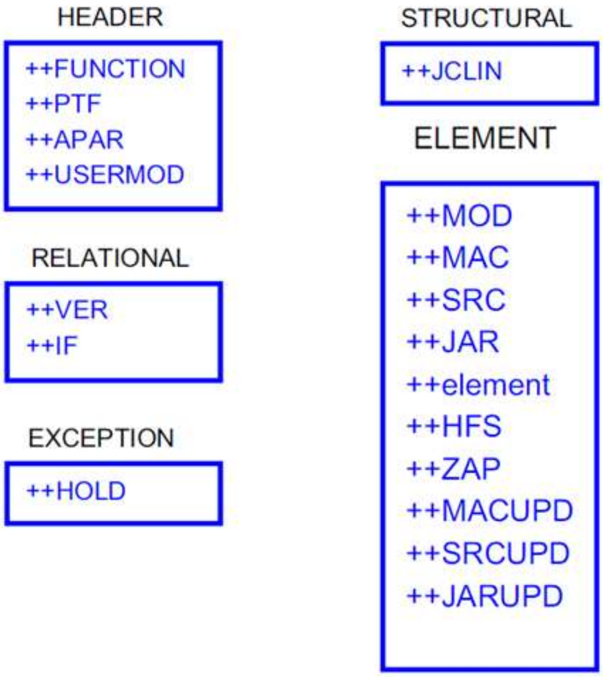
- FUNCTION – Adds a new product or function
- PTF – Provides an IBM correction to all z/OS installations
- APAR – Provides a temporary fix to a specific z/OS installation
- USERMOD – Adds an installation-provided modification.

CSI configuration



SYSMOD packaging

SYSMOD
MCS
Modification Text



HEADER

++FUNCTION
++PTF
++APAR
++USERMOD

RELATIONAL

++VER
++IF

EXCEPTION

++HOLD

STRUCTURAL

++JCLIN

ELEMENT

++MOD
++MAC
++SRC
++JAR
++element
++HFS
++ZAP
++MACUPD
++SRCUPD
++JARUPD

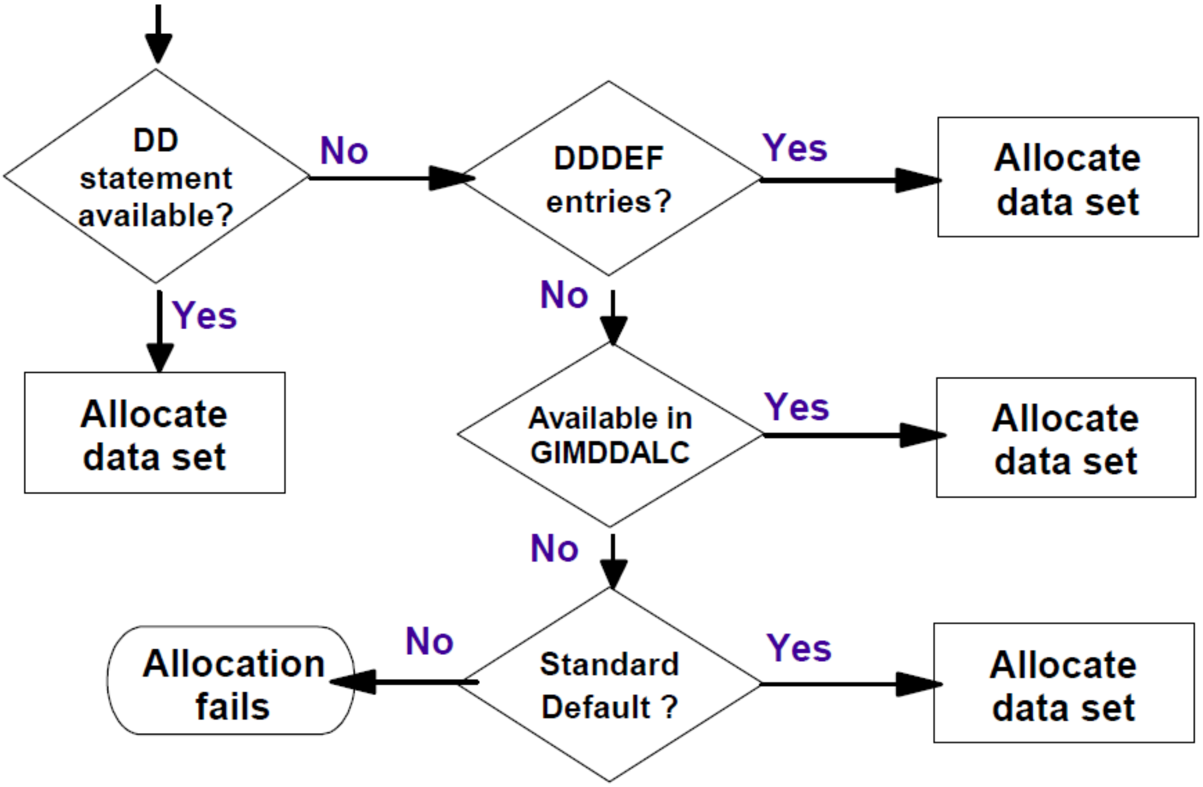
Packaging techniques

- Relative file
- Inline
- Indirect library
- GIMZIP archives

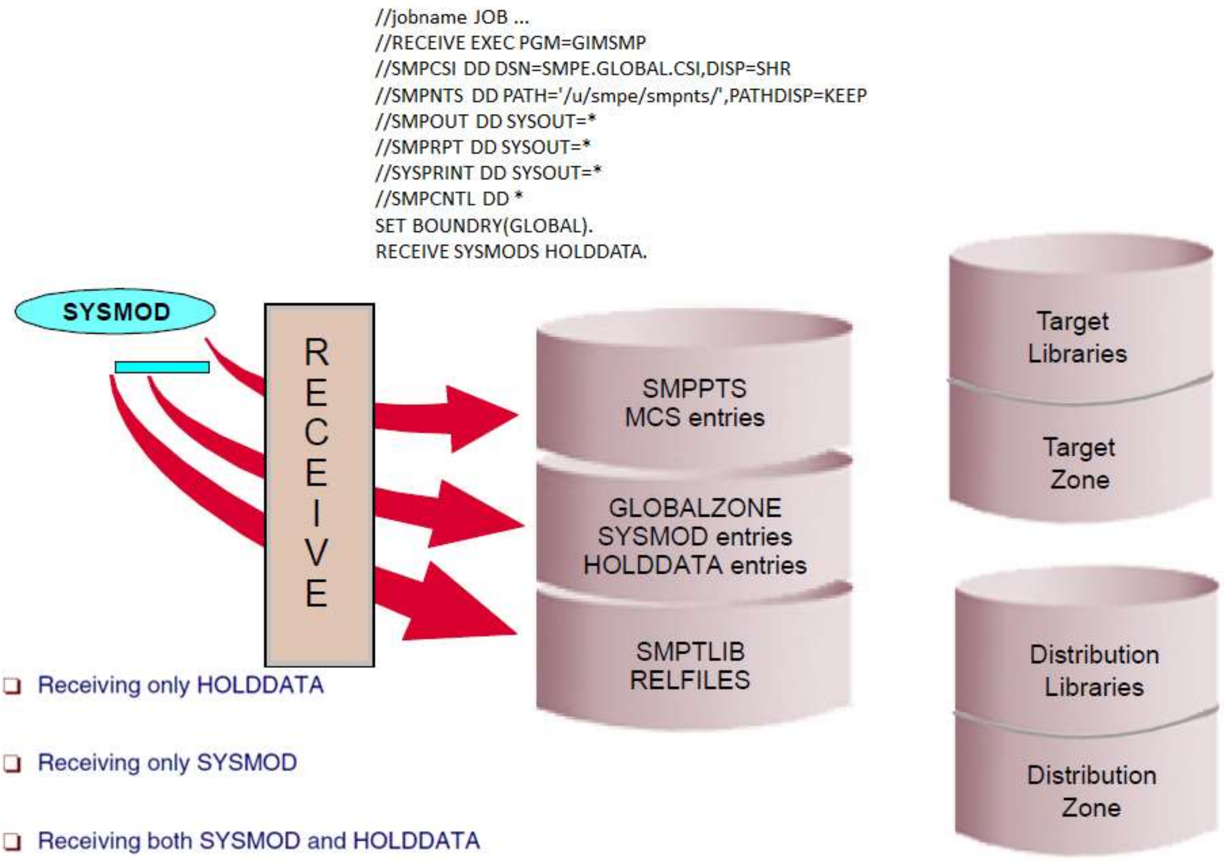
++CLIST
++DATA
++EXEC
++HELPxxx
++USERS
++PROC......

++SAMPxxx
++TEXTxxx
++USER1
++MSGxxx
++PARM

Dynamic allocation of datasets



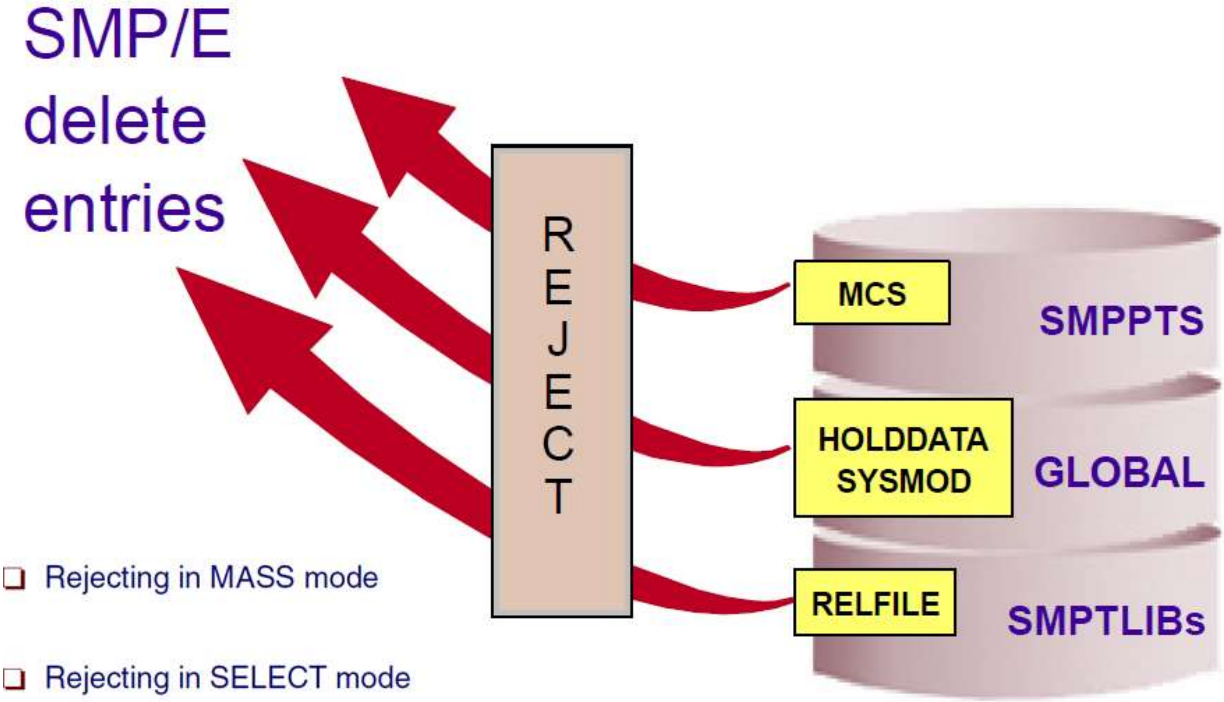
Receive



```
//jobname JOB ...
//RECEIVE EXEC PGM=GIMSMP
//SMPCSI DD DSN=SMPE.GLOBAL.CSI,DISP=SHR
//SMPNTS DD PATH='/u/smpe/smpnts/',PATHDISP=KEEP
//SMPOUT DD SYSOUT=*
//SMPRPT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SMPCNTL DD *
SET BOUNDRY(GLOBAL).
RECEIVE SYSMODS HOLDDATA.
```

- ❑ Receiving only HOLDDATA
- ❑ Receiving only SYSMOD
- ❑ Receiving both SYSMOD and HOLDDATA
- ❑ Receiving selected SYSMOD and HOLDDATA

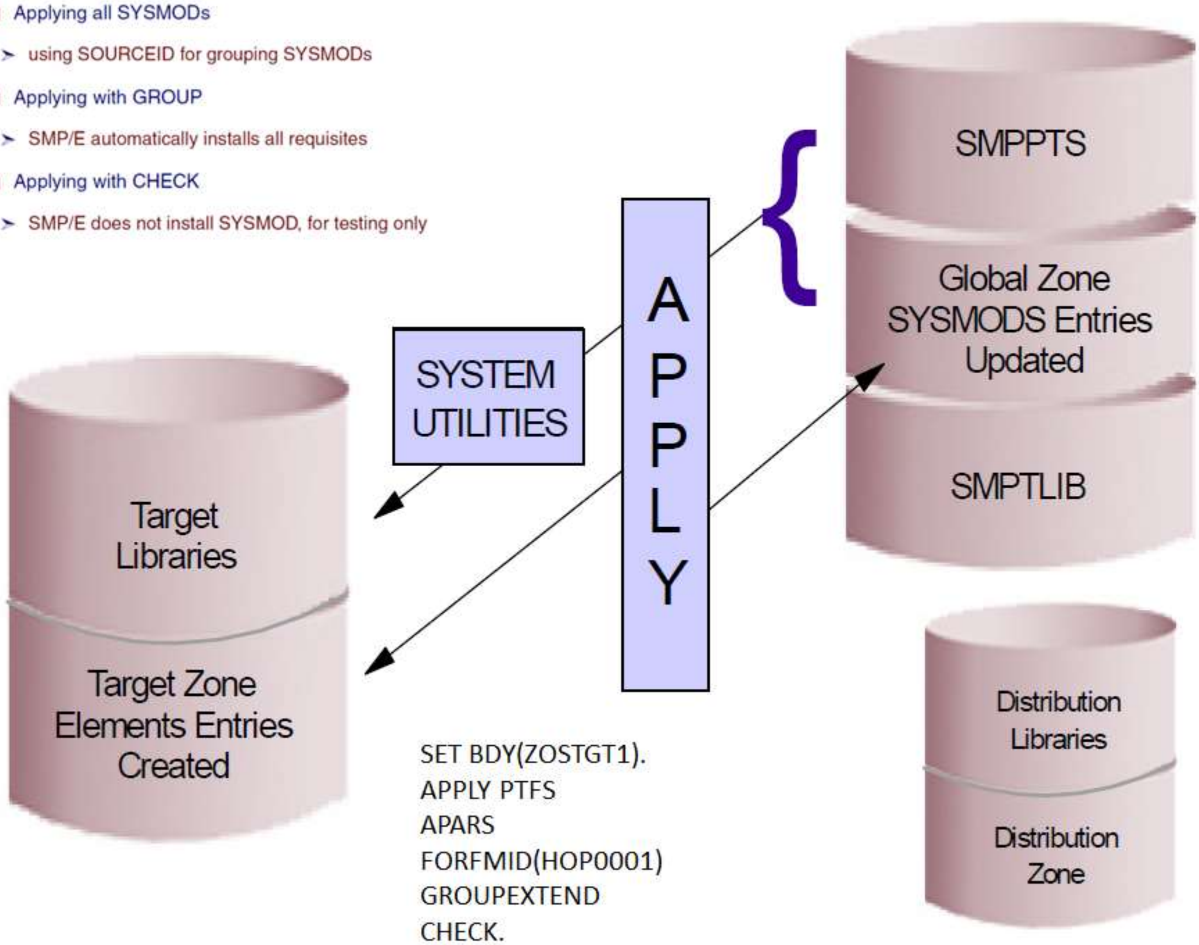
Reject



- ❑ Rejecting in MASS mode
- ❑ Rejecting in SELECT mode
- ❑ Rejecting in PURGE mode
- ❑ Rejecting in NOFMID mode

Apply

- ❑ Applying all SYSMODs
  - ➢ using SOURCEID for grouping SYSMODs
- ❑ Applying with GROUP
  - ➢ SMP/E automatically installs all requisites
- ❑ Applying with CHECK
  - ➢ SMP/E does not install SYSMOD, for testing only

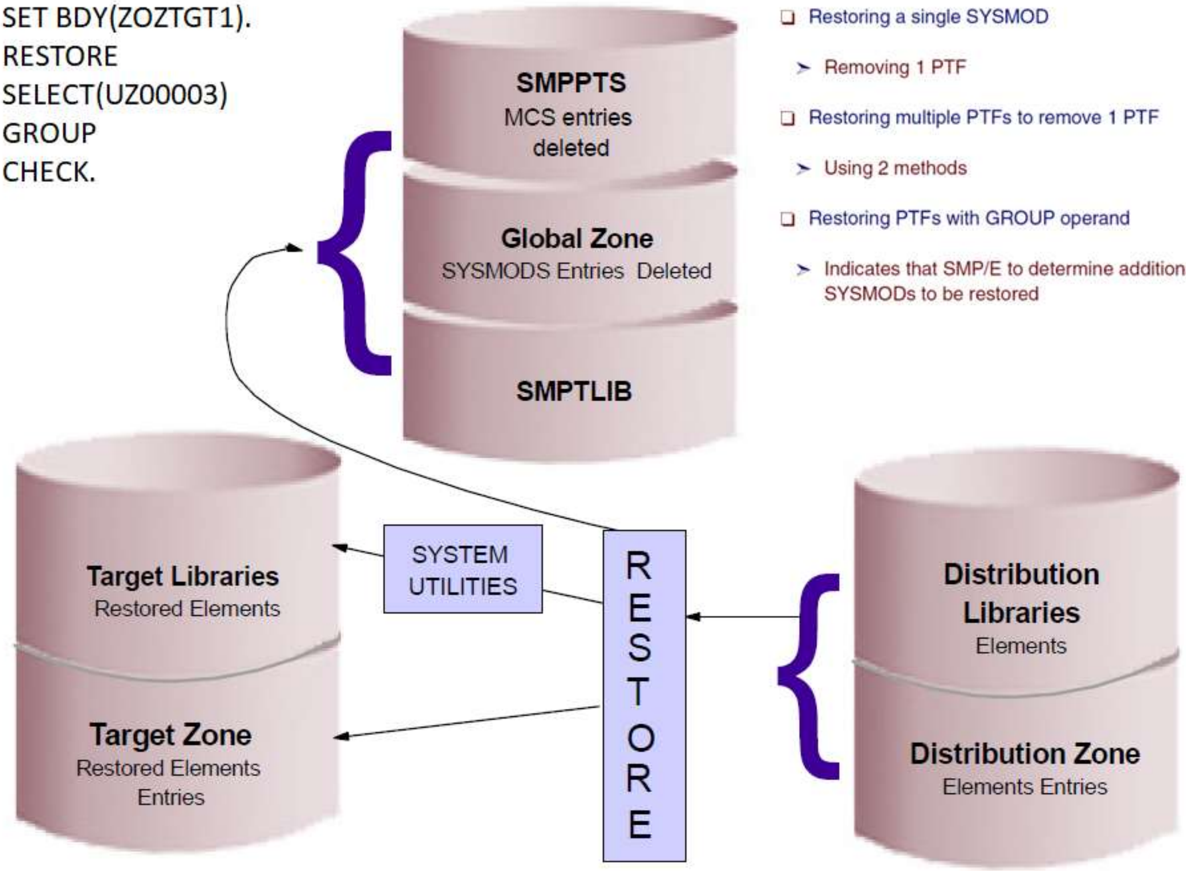


```
SET BDY(ZOSTGT1).
APPLY PTFS
APARS
FORFMID(HOP0001)
GROUPEXTEND
CHECK.
```

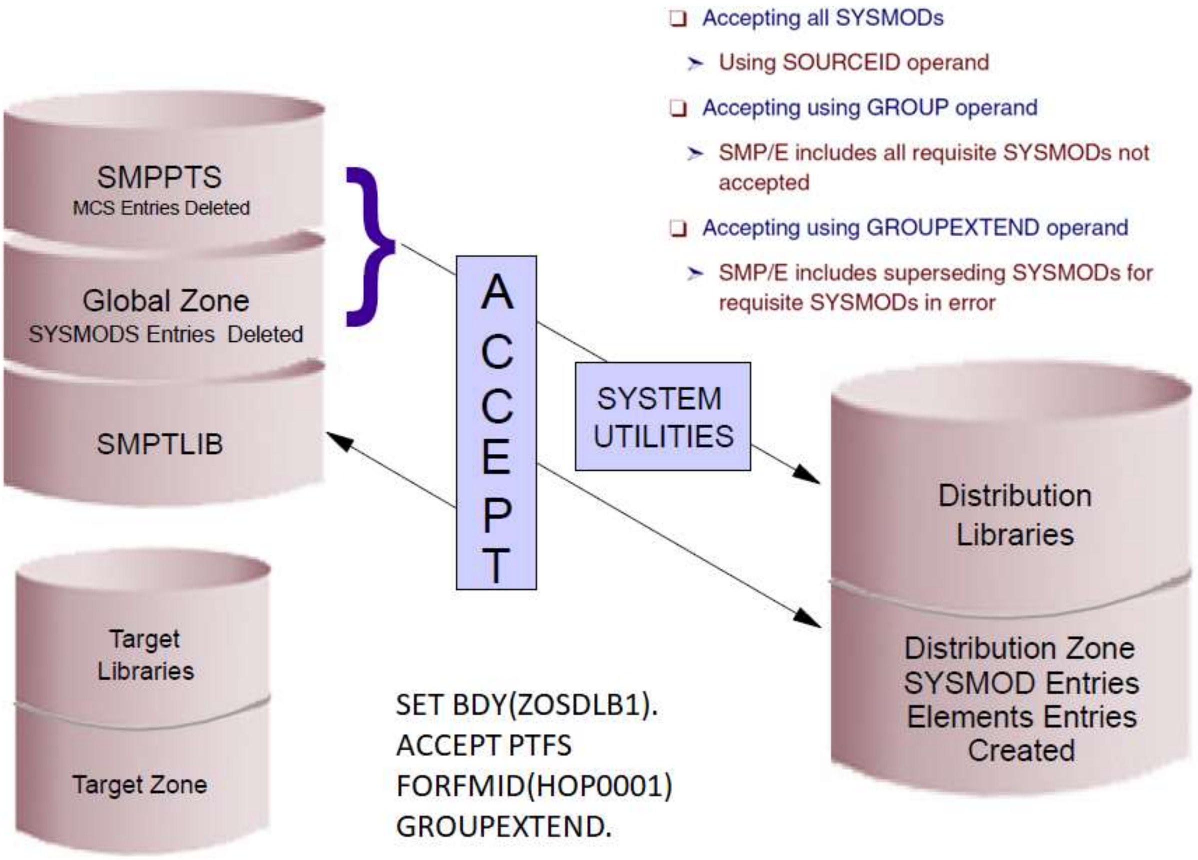
Restore

```
SET BDY(ZOZTGT1).
RESTORE
SELECT(UZ00003)
GROUP
CHECK.
```

- ❑ Restoring a single SYSMOD
  - ➢ Removing 1 PTF
- ❑ Restoring multiple PTFs to remove 1 PTF
  - ➢ Using 2 methods
- ❑ Restoring PTFs with GROUP operand
  - ➢ Indicates that SMP/E to determine additional SYSMODs to be restored



Accept



- ❑ Accepting all SYSMODs
  - ➢ Using SOURCEID operand
- ❑ Accepting using GROUP operand
  - ➢ SMP/E includes all requisite SYSMODs not accepted
- ❑ Accepting using GROUPEXTEND operand
  - ➢ SMP/E includes superseding SYSMODs for requisite SYSMODs in error

```
SET BDY(ZOSDLB1).
ACCEPT PTFS
FORFMID(HOP0001)
GROUPEXTEND.
```

Commands

- ACCEPT
- APPLY
- BUILDMCS
- CLEANUP
- DEBUG
- GENERATE
- GZONEMERGE
- JCLIN
- LINK LMODS
- LINK MODULE
- LIST
- LOG
- RECEIVE
- REJECT
- REPORT CROSSZONE
- REPORT ERRSYSMOD
- REPORT MISSINGFIX
- REPORT SOURCEID
- REPORT SYSMODS
- RESETRC
- RESTORE
- SET
- UCLIN
- UNLOAD
- UPGRADE
- ZONECOPY
- ZONEDELETE
- ZONEEDIT
- ZONEEXPORT
- ZONEIMPORT
- ZONEMERGE
- ZONERENAME

Pervasive Encryption

Pervasive Encryption for IBM Z is a consumable approach to enable extensive encryption of data in-flight and data at-rest to substantially simplify encryption, and reduce costs associated with protecting data.

Pervasive Encryption for IBM Z can be used in conjunction with full disk and tape encryption, database encryption, and application encryption.

Full disk and tape encryption

Full disk and tape encryption protects against intrusion, tamper or removal of physical infrastructure.

File or data set encryption

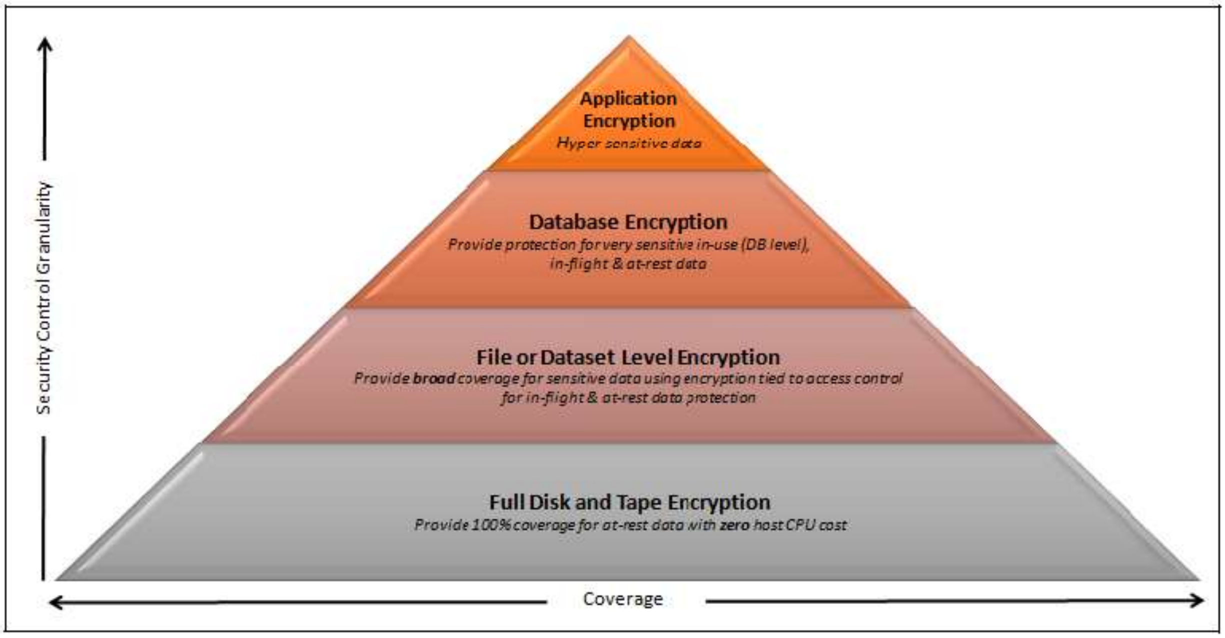
File or data set encryption is managed through z/OS, and provides simple policy controls that allow clients to protect data in mission critical databases including DB2, IMS and VSAM. Additionally, z/OS data set encryption gives clients the ability to eliminate storage administrators from the compliance scope.

Database encryption

Database encryption provides selective encryption and granular key management control of sensitive data.

Application encryption

Application encryption is used to encrypt sensitive data when lower levels of encryption are not available or suitable.


Application Encryption
Hyper sensitive data
Database Encryption
Provide protection for very sensitive in-use (DB level), in-flight & at-rest data
File or Dataset Level Encryption
Provide broad coverage for sensitive data using encryption tied to access control for in-flight & at-rest data protection
Full Disk and Tape Encryption
Provide 100% coverage for at-rest data with zero host CPU cost
Security Control Granularity
Coverage

zOS Container Extension

New function in z/OS 2.4 that enables clients to:

- Deploy Linux on Z software components as Docker Containers in a z/OS system, in direct support of z/OS workloads
- Without requiring a separately provisioned Linux server
- While significantly improving network transaction rates and reducing latency
- While maintaining overall solution operational control within z/OS and with z/OS Qualities of Service
- Requires IBM z14 (or later) based server with Container Hosting Foundation (feature code 0104)
    - zCX Trial - Try and Buy capability let's you optionally kick the tires for 90 days

What is a Docker

A Packaging standard for software

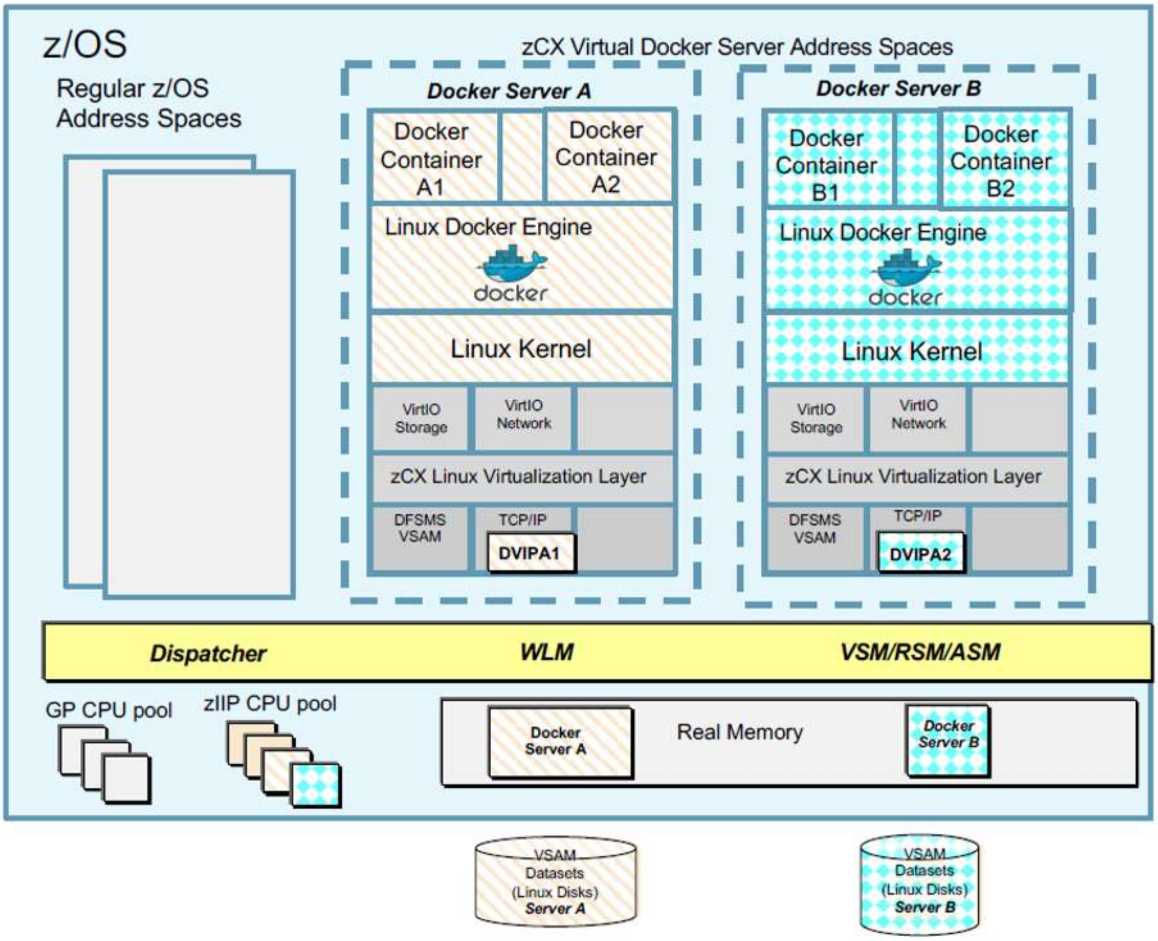
- Think of it like a shipping container
- Makes moving, stacking, unstacking of compliant software easier
- Common in the application world on Linux and cloud

Dockerhub

- Contains many popular docker packages
- s390x packages support Linux on z
- https://hub.docker.com/search?q=&type=image&architecture=s390x

By focusing on Docker

- We reduce the complexity of installation and configuration for the user
- We reduce the service footprint on Linux to what Docker supports
- We gain access to a large number of packages out of the box


z/OS
Regular z/OS Address Spaces
zCX Virtual Docker Server Address Spaces
Docker Server A
Docker Container A1
Docker Container A2
Linux Docker Engine
docker
Linux Kernel
VirtIO Storage
VirtIO Network
zCX Linux Virtualization Layer
DFSMS VSAM
TCP/IP
DVIPA1
Docker Server B
Docker Container B1
Docker Container B2
Linux Docker Engine
docker
Linux Kernel
VirtIO Storage
VirtIO Network
zCX Linux Virtualization Layer
DFSMS VSAM
TCP/IP
DVIPA2
Dispatcher
WLM
VSM/RSM/ASM
GP CPU pool
zIIP CPU pool
Docker Server A
Real Memory
Docker Server B
VSAM Datasets (Linux Disks) Server A
VSAM Datasets (Linux Disks) Server B

USS and zLinux

USS

- UNIX System Services is crucial to the e-business run time environment on z/OS®
- It is a key element in supporting middleware and applications, and is an integral part of some strategic solutions such as WebSphere
- It provides a sound and robust UNIX environment, bringing with it the qualities of service (availability, security, workload balancing) expected and delivered with z/OS
- USS will continue to be enhanced to support the e-business infrastructure, as well as vendor and customer applications running on z/OS

Linux on z

- Linux on zSeries is a new environment with rapidly growing acceptance in the marketplace
- It brings a large number of applications and faster porting of existing UNIX applications that do not need to exploit z/OS features or qualities of service
- It enables the consolidation of a large number of Linux servers onto a single hardware platform

When use USS

- When you need the full range of z/OS Qualities of Service
- To take full advantage of the platform's strengths in availability, handling mixed workloads, security, transaction processing and performance

- To optimize database access performance when running an application in the same address space as DB2
- For access to the full set of advanced capabilities of WorkLoad Manager (WLM)
- To combine existing OS/390 applications with newer components like files and datasets that run in a UNIX environment
- To provide the highest degree of availability
- To utilize Parallel Sysplex capabilities
- To utilize Recoverable Resource Management Services (RRMS) functionality
- When you need advanced security features such as those provided by Remote Access Control Facility (RACF)

When to use Linux on z

- Porting a C/C++ application to Linux on zSeries is easier and faster than a port to USS
- Using the System Administration Facility, Linux on zSeries provides an environment that supports hundreds of images
- Consider porting applications that need 64-bit addressing to Linux on zSeries
- Applications that need full ASCII capabilities or that need ASCII double byte character sets are supported in Linux on zSeries
- When speed to market is crucial, utilizing LPARs to simultaneously develop, produce and test on the same machine can give a crucial edge

- Linux on zSeries provides a rich development environment for application programmers

Benefits of Linux on z

1. Scalability (ten to thousands of images depending on workload)

2. z/Architectecture exploitation: IRD & WLM

3. Integrated business solutions: data richness of zSeries and applications

4. Flexibility and openness of Linux with the Quality of Service (QOS) of zSeries

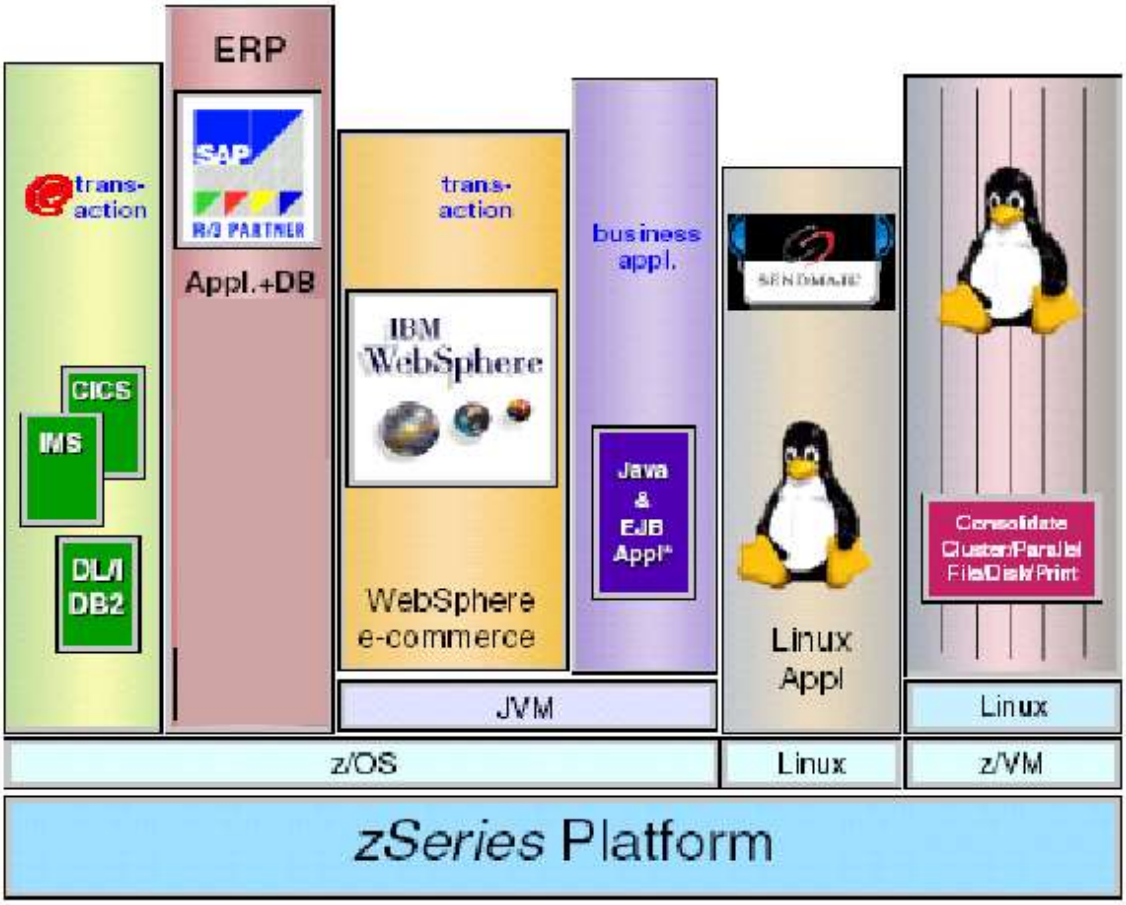
5. zSeries customers are offered additional opportunities to leverage their investments through Linux

6. Consolidation benefits while maintaining one server per application

7. Reduced Total Cost of Ownership (TCO)

8. Improved service level: zSeries QOS

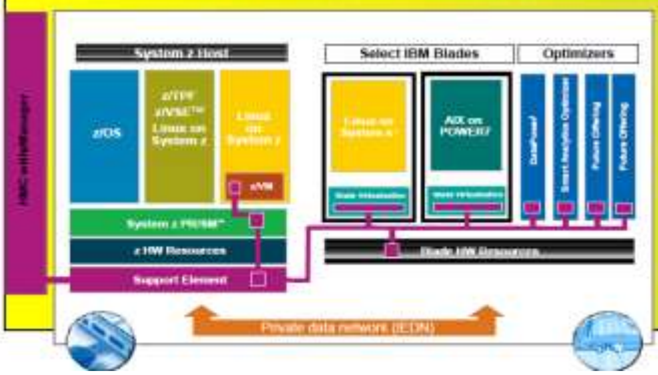
9. Speed to market time increased


ERP
trans-action
SAP R/3 PARTNER
Appl.+DB
CICS
IMS
DL/I
DB2
trans-action
IBM WebSphere
WebSphere e-commerce
business appl.
Java & EJB Appl*
SENDMAIL
Linux Appl
Consolidate Cluster/Parallel File/Disk/Print
JVM
Linux
z/OS
Linux
z/VM
zSeries Platform

zOS Automation

System Automation for z/OS - Components

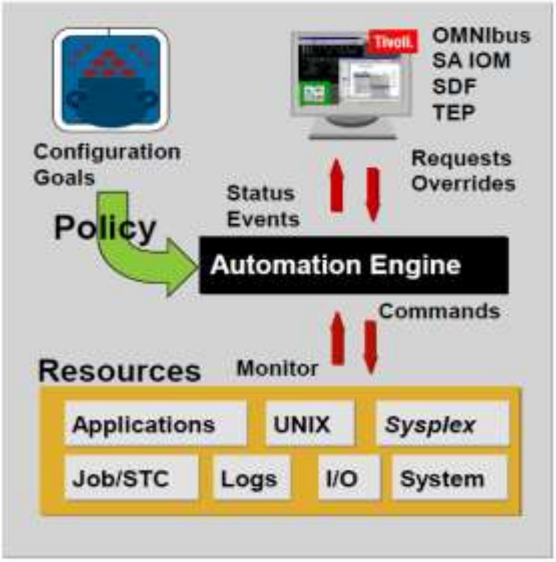
- Automate repetitive and complex tasks in **z/OS systems**
  - ✓ Helps operators to perform their tasks in a more robust way
  - ✓ Reduces z/OS specific skill requirements
- **Automated operations** through monitoring of applications, messages, and alerts:
  - Increase availability and performance through pro-active automation



Processor Operations

- ✓ Automate and control **hardware operations**
- ✓ Power on/off and reset
- ✓ Perform system IPL for z/OS, Linux, z/VM, blades
- ✓ Automate **LPAR settings** like weights and capping

Component System Operations

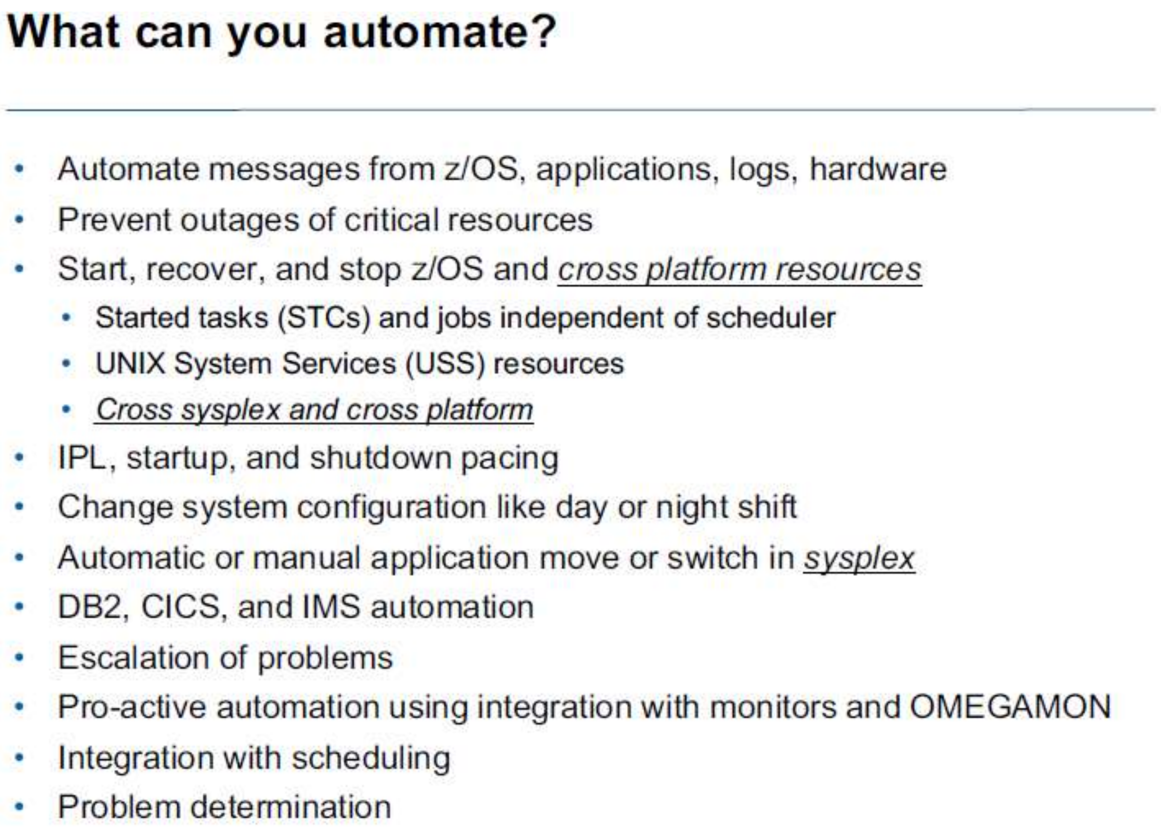


Policy-based automation

- Start, recover, and shutdown applications, system and *sysplex* using:
  - Timers, events, triggers, service periods...
  - Dependencies and groups
  - Thresholds, active and external monitors
- JES, CICS, IMS, DB2, TWS, WebSphere, OMEGAMON, UNIX, *SAP, GDPS*...
- Message monitoring & response
  - WTO, WTOR, joblog, UNIX, NetView, hardware
- Escalation to SA IOM, OMNIbus...
- Prevent outages of critical resources:
  - WTO and AMRF buffers, spool, *sysplex*
  - SYSLOG, LOGREC, SMF, dump data sets

Easier operations at the application level

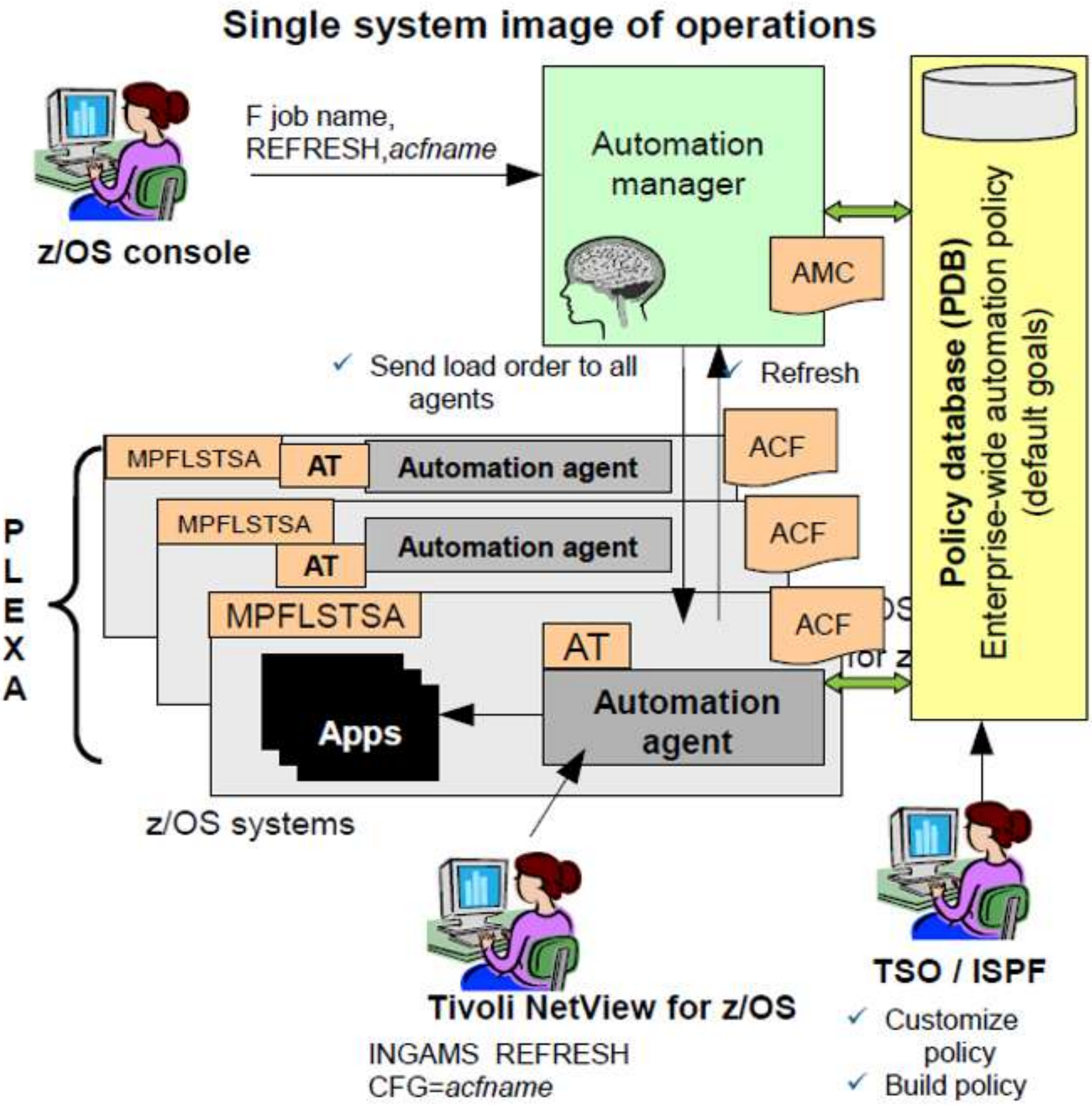
- 3270 or Tivoli Enterprise Portal
- Single point of control, *single system image*
- Goal-driven automation



What can you automate?

- Automate messages from z/OS, applications, logs, hardware
- Prevent outages of critical resources
- Start, recover, and stop z/OS and *cross platform resources*
  - Started tasks (STCs) and jobs independent of scheduler
  - UNIX System Services (USS) resources
  - *Cross sysplex and cross platform*
- IPL, startup, and shutdown pacing
- Change system configuration like day or night shift
- Automatic or manual application move or switch in *sysplex*
- DB2, CICS, and IMS automation
- Escalation of problems
- Pro-active automation using integration with monitors and OMEGAMON
- Integration with scheduling
- Problem determination

zOS Automation Policy



zOS Data Sets

Here is list of most commonly used datasets under zOS.

Direct Data Sets

In this dataset, the records are arranged in any sequence your program indicates, and retrieves records by actual or relative address. If you do not know the exact location of a record, you can specify a point in the data set where a search for the record is to begin.

Basic Direct Access Method (BDAM) access method manages this data set.

Partitioned Data Set (PDS)

A PDS contains multiple members of sequential datasets and includes a directory that relates member names to locations within the data set. PDSs can have any type of sequential records.

Basic partitioned access method (BPAM) access methos manages this data set.

Partitioned Data Set Extended (PDSE)

A PDSE is an extension of PDS. A PDSE has a different internal storage format than a PDS, which gives PDSEs improved usability characteristics. You can use a PDSE in place of most PDSs, but you cannot use a PDSE for certain system data sets.

Basic partitioned access method (BPAM) access methos manages this data set.

z/OS UNIX files

UNIX files are byte streams and do not contain records. BPAM converts the bytes in UNIX files to records.

Basic partitioned access method (BPAM) access methos manages this data set.

Sequential Data Set

This data set contains the records in the order they are entered.

Basic Sequential Access Method (BSAM) access method manages this dataset.

VSAM Data Sets

VSAM data sets arrange records by an index key, relative record number, or relative byte addressing. VSAM is used for direct or sequential processing of fixed-length and variable-length records on DASD. Data that is organized by VSAM is cataloged for easy retrieval and is stored in one of five types of data sets.

Entry-sequenced data set (ESDS). Contains records in the order in which they were entered. Records are added to the end of the data set and can be accessed.

Key-sequenced data set (KSDS). Contains records in ascending collating sequence. Records can be accessed by a field, called a key, or by a relative byte address.

Linear data set (LDS). Contains data that has no record boundaries. Linear data sets contain none of the control information that other VSAM data sets do. Linear data sets must be cataloged in a catalog.

Relative record data set (RRDS). Contains records in relative record number order, and the records can be accessed only by this number. There are two types of relative record data sets.

Fixed-length RRDS: The records must be of fixed length.

Variable-length RRDS: The records can vary in length.

Catalogs

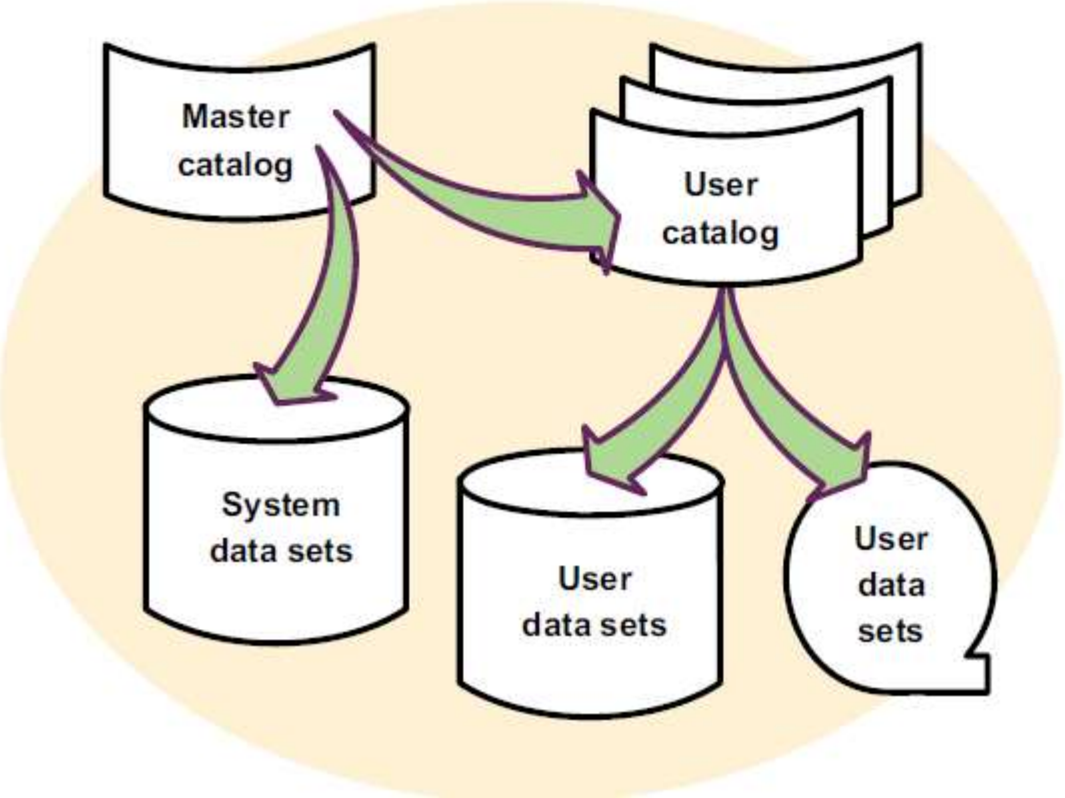
There are two types of catalogs to manage all the datsets under zOS.

Master Catalog

User Catalog.

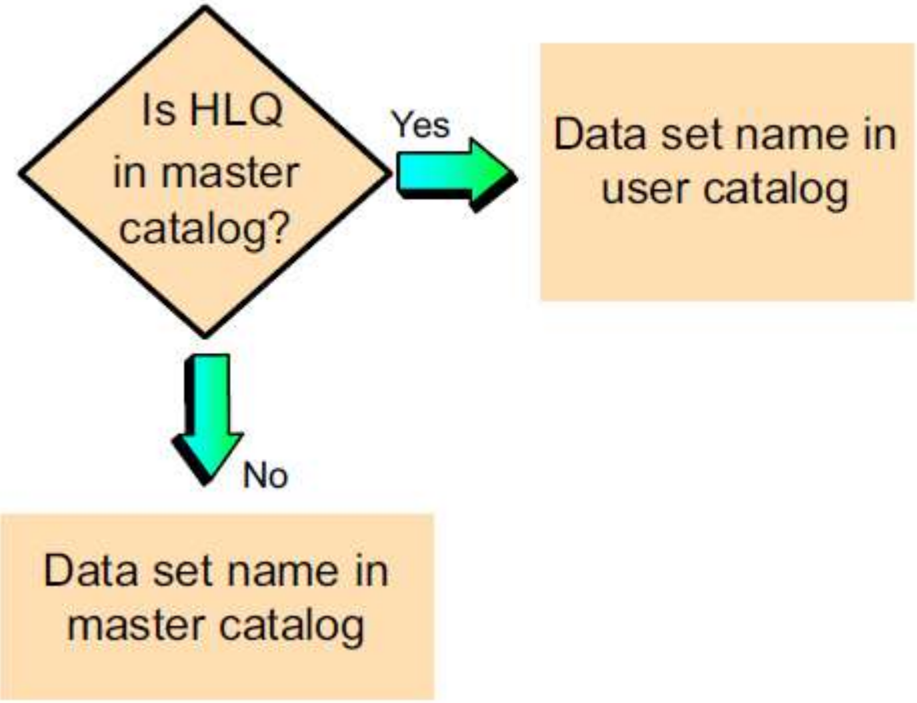
In general, the master catalog should contain only system-related entries. The master catalog contains pointers to system data sets and user catalogs.

User catalogs contain pointers to user data sets.

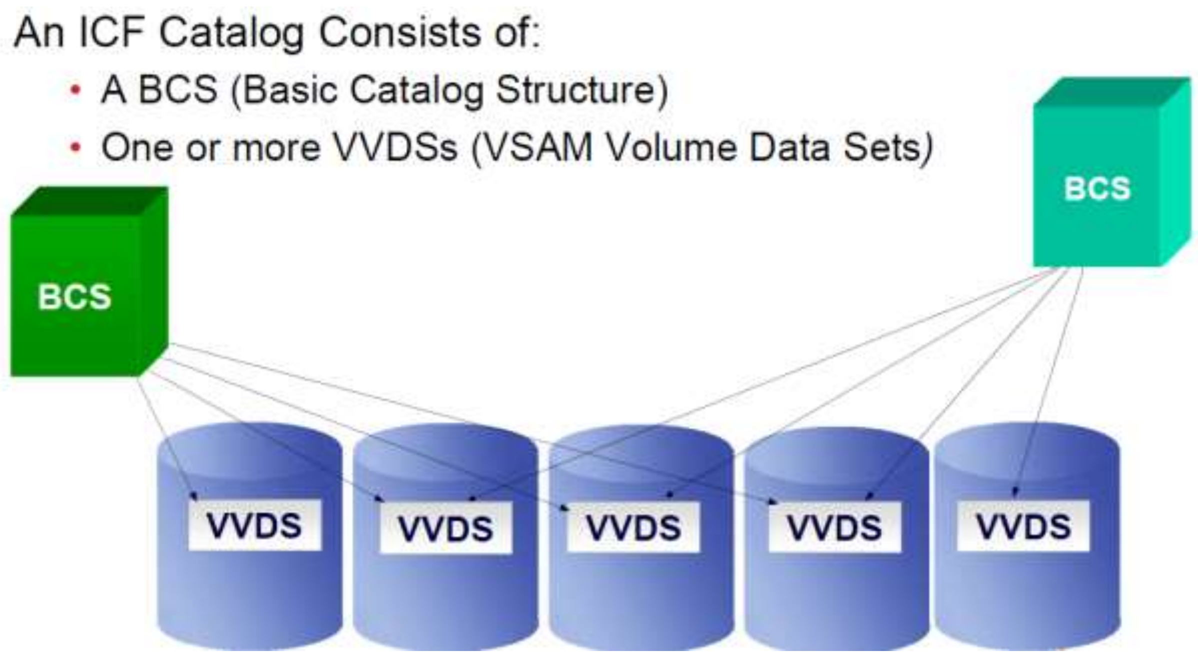


Catalog search order

The system searches the master catalog for an alias. If one is found, it will go to the user catalog specified and search for the fully qualified data set name. If one is not located, the system will search the master catalog for the fully qualified data set name.


Is HLQ in master catalog?
Yes
Data set name in user catalog
No
Data set name in master catalog

Catalog structure



An ICF Catalog Consists of:

- A BCS (Basic Catalog Structure)
- One or more VVDSs (VSAM Volume Data Sets)

The VVDS is a VSAM entry-sequenced data set that has a 4KB control interval size. It contains information about the VSAM and SMS-managed non-VSAM data sets residing on the DASD volume with the VVDS.

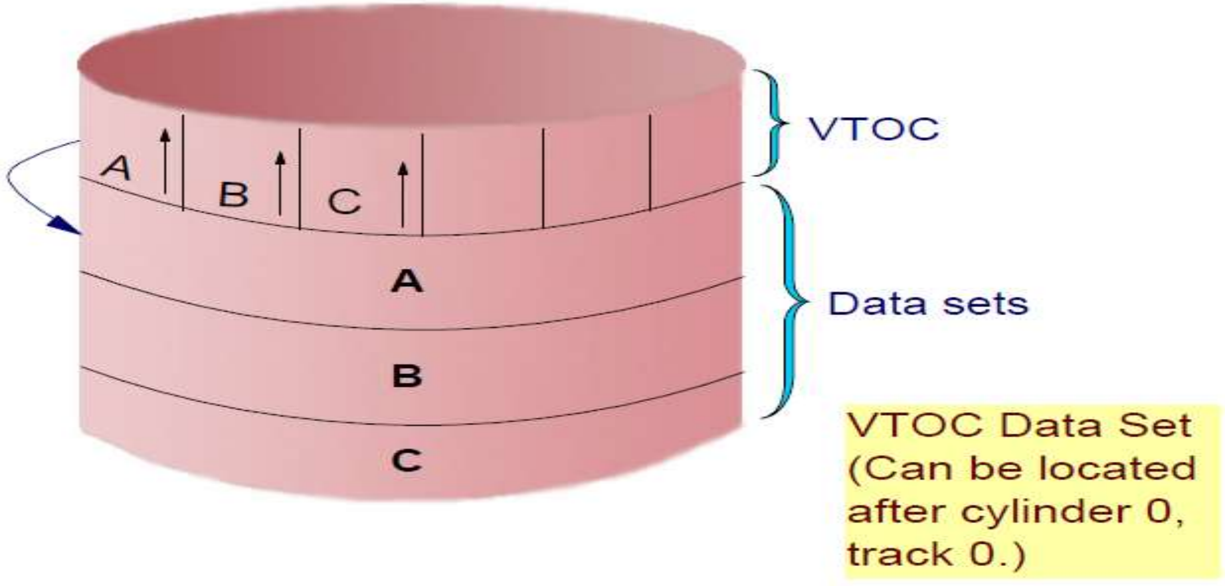
VTOC

The VTOC is a data set that describes the contents of the direct access volume on which it resides. It is a contiguous data set; that is, it resides in a single extent on the volume and starts after cylinder 0, track 0, and before track 65,535. A VTOC's address is located in the VOLVTOC field of the standard volume label. The volume label is described in z/OS DFSMS Using Data Sets. A VTOC consists of complete tracks.

The VTOC lists the data sets that reside on its volume, along with information about the location and size of each data set, and other data set attributes. It is created when the volume is initialized through the ICKDSF utility program.

The VTOC locates data sets on that volume. The VTOC is composed of 140-byte data set control blocks (DSCBs). That correspond either to a data set currently residing on the volume, or to contiguous, unassigned tracks on the volume. A set of assembler macros is used to allow a program or z/OS to access VTOC information.

The IEHLIST utility can be used to list, partially or completely, entries in a specified volume table of contents (VTOC), whether indexed or non-indexed. The program lists the contents of selected data set control blocks (DSCBs) in edited or unedited form.



Mainframe zOS - Roles and responsibilities of a Systems Programmer

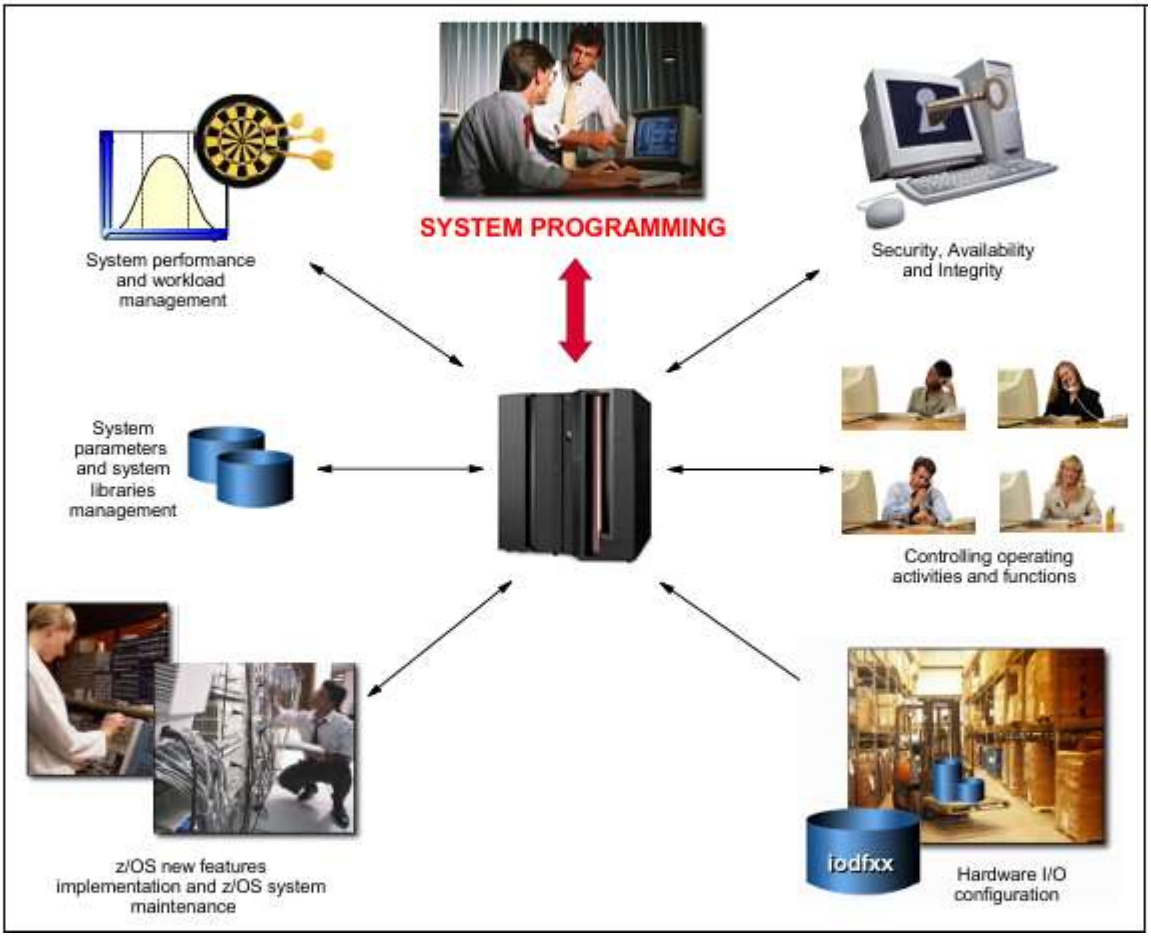
I have received this question from many, starting/planning their career as a ‘zOS Systems Programmer’.

My definition: A zOS Systems Programmer (Sysprog), has four specific responsibilities. INSTALL, CUSTOMIZE, MAINTAIN, and SUPPORT zOS systems, sub-systems and software products under zOS.

Is it not simple to learn and be a zOS Sysprog ?

Let me describe it in more detail.

The job of the z/OS system programmer is very complex and requires skills in many aspects of the system. In a mainframe IT organization, the system programmer (or system**s** programmer) plays a central role. The system programmer installs, customizes, and maintains the operating system, and also installs or upgrades products that run on the system. The system programmer might be presented with the latest version of the operating system to upgrade the existing systems. Or, the installation might be as simple as upgrading a single program, such as a sort application. In most large z/OS installations, system programmers become specialists for only specific tasks. Generally speaking, the system programmer is responsible for managing the mainframe hardware configuration, and installing, customizing, and maintaining the mainframe operating system. System programmers ensure that their installation's system and its services are available and operating to meet service level agreements. Installations with 24-hour, 7-day operations need to plan for minimal disruption of their operation activities.


SYSTEM PROGRAMMING
System performance and workload management
Security, Availability and Integrity
System parameters and system libraries management
Controlling operating activities and functions
z/OS new features implementation and z/OS system maintenance
iodfxx
Hardware I/O configuration

The system programmer performs such tasks as the following:

- Planning hardware and software system upgrades and changes in configuration
- Training system operators and application programmers
- Automating operations
- Capacity planning
- Running installation jobs and scripts
- Performing installation-specific customization tasks
- Integration-testing the new products with existing applications and user procedures
- System-wide performance tuning to meet required levels of service

The ‘Programmer’ word in ‘Systems Programmer’ is very interesting. To me, it is a legacy, and don’t get confused that you have to do a lots of programming as a Sysprog. In the past, Sysprogs were doing a lots of programming in systems area to meet the installation’s need. But today, it is very much limited to maintaining the EXITs, USERMODs and old system tools, developed for the installation.

As a Sysprog, whether everyone does everything ? The answer in general is NO.

In many organizations, the ‘installation, customization and maintenance’ activities are grouped under a framework called ‘BUILD the systems’ where less than 25% of the time is spent. And, ‘support’ activity is called ‘RUN the systems’ where more than 75% of the time is spent.

A small group of Sysprogs, install and maintain the softwares, who need the SMPE skill to perform this. This is mostly done centrally because of deployment complexity in rolling the build to different systems and keeping all the systems in synch. However, there could be an exception for emergency maintenance which can be done by any SME.

Customization is a decentralized activity as it needs the specific skills (CICS, DB2, Storage…) to customize and verify it in the respective areas.

Support activity is performed by all. Here you handle all day-to-day issues and problems in the system and make sure that, the systems, subsystems and products run fine without any issues. Here, you work with all relevant parties inside and outside your organization.

The system programmer must be skilled at debugging problems with system software. These problems are often captured in a copy of the computer's memory contents called a dump , which the system produces in response to a failing software product, user job, or transaction. Armed with a dump and specialized debugging tools, the system programmer can determine where the components have failed. When the error has occurred in a software product, the system programmer works directly with the software vendor's support representatives to discover whether the problem's cause is known and whether a patch is available.

Under zOS, competencies are built over different areas: zOS, CICS, DB2, IMS, MQ, Network, Security, Storage, CPO, Automation, etc. who are called SMEs. So we say, ‘I am a zOS or CICS or DB2 Sysprog’. And very often we use the word ‘Administrator (Admin)’ instead of ‘Sysprog’ such as ‘zOS or CICS or DB2 Admin’.

In a large z/OS installation, there is usually a separation of duties both among members of the system programming staff, and between the system programming department and other departments in the IT organization. A typical z/OS installation includes the following roles and more:

- z/OS system programmer
- CICS system programmer
- Database system programmer
- Database administrator
- Network system programmer
- Automation specialist
- Security manager
- Hardware management
- Production control analyst
- System operator – Console and Batch
- Network operator
- Security administrator
- Service manager
- Capacity, Performance and Optimization
- Finance

Once a SME gets some mastery (3 to 5 years) in one area, can grow into other areas. It is a general practice to allow SMEs to grow horizontally in their technical skills. So, depending on the size of the zOS installation, the total number of Sysprogs could be anywhere between less than 10 to more than 100.

Not to forget the critical activities spanning over to hardware areas; IODF definition and HMC configuration. Few Sysprogs have the special skill to perform this in collaboration with the hardware engineers.

Also the zOS architect(s) taking care of the end-to-end zOS design and engineering activities and prepare a long term strategy/roadmap for the zOS in the organization. They also collaborate on zOS costing and chargeback.

Mainframe zOS - Technical references

zOS Sysprogs, look at the SYSLOG for anything and everything throughout the day. But, when I have asked the new Sysprogs to explain the syslog messages, interpret the fields in it and in which manual the message format is described, I have seen a '??' in more than 80% of the faces. So, I tell them to read 'chapter-1 -Introduction' of the zOS system messages volume-1 where to find a wealth of information. Here you find a table which describes the prefix of systems messages and where the messages are documented. No doubt, you can search any message online, but it is important to know, which zOS component has issued the message and in which document you can find its explanation.

Hardcopy manuals are becoming history. In 1990's we used to joke at customer site, 'IBM' means' I Bring Manuals'. Today, most information is available in soft copy or online and is extremely easy and fast to search and refer to any page in the document.

I suggest the zOS Sysprogs to familiarize with various manuals to know what information they can find where. Of course, you can search online to find everything available in the internet.

When I work with any component or product, which I don't know, I always look for the introduction chapter in one of the related manuals. Then I move to more in-depth reading related to my work.

Here are few sources which I refer to:

Manuals - You will find various manuals such as …. Introduction, Installation, Customization, Migration, Planning, Operation, Guide, Reference, Messages, Diagnosis, Codes, Commands, Data Areas etc. You can find all zOS related manuals in web page 'zOS Internet Library'. You can find most of the other manuals in the related vendor web sites.

Redbooks - I suggest to have the Redbooks as a very good source of reference. For new zOS Sysprog, 'ABCs of zOS Systems Programming' (13 volumes) Redbooks are a great source of reference. I co-authored two Redbooks for zOS 1.5 and 1.6 and it was a great experience for me. Especially, when I tested and wrote on 'Restartable PDSE address space (SMSPDSE1)' which was introduced for first time in zOS.

Internet - There are plenty of presentations and other references available in the internet.

zTidbitz - You will find a lots of good information in the internet.

When I teach new zOS Sysprogs or prepare a learning roadmap for them, I make sure to include 2 hours session to give them an overview on ‘documentation’ and explain them the value of it and most importantly how to find out the right information quickly.

Lastly, you know that, zOS is full of abbreviations and terminologies. When you have time, please read ‘Glossary of zOS terms and abbreviations’. You will find similar information for other components such as DB2.

Here is a list of manuals for zOS components (may not be complete).

Encryption Facility for z/OS

EREP

GDDM

HLASM

IBM HTTP Server

IBM Tivoli Directory Server for z/OS

IBM Z Deep Neural Network Library (zDNN)

IBM Z Multi-Factor Authentication

IBM Z Platform for Apache Spark

IBM Z System Automation

IBM z/OS Change Tracker

IBM z/OS Management Facility

ICKDSF

Infoprint Server, Infoprint Transforms PSF for z/OS, DCF, fonts

ISPF

REXX Alternate Library

Runtime Library Extensions

SDSF

z/OS Authorized Code Scanner

z/OS Common Information Model

z/OS Communications Server

z/OS Cryptographic Services

z/OS Data Gatherer

z/OS Data Set File System

z/OS DFSMS

z/OS DFSORT

z/OS File System (zFS)

z/OS HCD

z/OS HCM

z/OS Integrated Security Services

z/OS JES2

z/OS Language Environment

z/OS MVS

z/OS NFS

z/OS OpenSSH

z/OS RMF

z/OS Security Server RACF

z/OS SMP/E

z/OS System-Level

z/OS TSO/E

z/OS UNIX System Services

z/OS XL C/C++

z/OS Xvfb

IT Skill Roadmap

The IT skill requirement in the market is numerous and based on demand by the industry.

As I see, less than 20% of the new comer to IT plan their initial career before starting to work in areas like Windows, Java, Unix, Linux, Mainframe, and numerous other IT skill sets. It is mostly because, they studied certain skills in their universities, executed certain IT projects, underwent some training, got some career coaching etc.

The rest 80% or so joined some organization to work and then put into an area where there is a demand. As I saw in a company, a group hired to work on Unix got trained on zOS and other group promised to work on DB2 got trained on IMS. Few got frustrated due to this, but most of them accepted the reality and moved forward.

So I say, while choosing an IT skill, do some research, decide, fall in love and then get married. Otherwise, marry first before falling in love. But, once you fall in love, love with all sincerely and with right attitude, mindset, and adaptability which will help you to create your skill roadmap in the future.

If you come to mainframe, you either work in development area where you develop and support applications or work in systems area where you work as a systems programmer.

If you work in application area, you have a career path in applications development (using COBOL, PL1, Java etc.), support, performance management and finally become an architect and manage specific business areas.

If you work as Sysprog, you specialize in zOS, CICS, DB2, MQ, Automation etc.

Under mainframe, you have a chance to gain skill on multiple areas like zOS, zVSE, zVM, zTPF, Linux, Unix and many products.

Whatever path you may choose, I will suggest to expand your skill horizontally, then only you will survive and meet the market demand, volatility and dynamism.

Always look for ‘opportunity’ and ‘ask for what you want’. This may take time but you will get it.

You have a chance to choose generic skills which helps you to move from platform to platform easily. Some of them are Database, Networking, Security, Storage, programming languages like Java etc.

There are other areas like Audit and Compliance where it goes beyond IT.

In IT industry, please get ready to face many challenges like losing your job without any notice or being forced to learn new skill or work in areas not of your interest. Here, I suggest you to remember ‘IF-THEN-ELSE’ logic and ‘Think for the best and prepare for the worst’ statement.

Job switching in IT industry is extremely common today. I have seen people doing the same repetitive work for 25/30 years without getting bored and in other hand, people complaining ‘I am working in this area for 2/3/5 years and need a change’. I have seen resumes of people changing multiple jobs in a single year. Good or bad, it is a debate. But, if you want to expand your skill horizontally, you will achieve by working relatively long time in the same organization.

Building zOS Sysprog skill pool

This is my view to address the shortage/crisis/gap of zOS Sysprog skill pool.

All mainframe organizations talk about it. No doubt some actions are being taken to address this. But, I have not seen a long term sustainable approach by organizations to build the Sysprogs systematically. Rather, when there is a need, they always look at the open market to hire one. But, I feel, the organizations can equip themselves internally to create and maintain their skill pool.

In my experience, to build a zOS Sysprog from Zero level, you need minimum 1 year of dedicated time investment. 3 months of classroom training with lab on zOS basics, 3 months of guided hands-on in the 'Test systems' and finally 6 months of mentoring with hands-on on specific areas (zOS, CICS, DB2, MQ etc.). Now they will be ready to perform tasks in 'Prod systems' under supervision followed by performing the same independently.

I feel, creation of a skill pool should be a virtual factory where we have a well-developed process to build a talent pipeline to meet the demand at any time. My suggestion follows.

1. Mentor & Mentee program (Horizontal skill development) in zOS Support team - This is the best approach to face any unforeseen situation. A zOS Sysprog in an installation should acquire at least 3 or more competencies e.g. zOS, CICS and DB2, at different level. They have a prime responsibility in one area and secondary responsibility in others. So, when a person, on primary role, leaves the organization, we have others to close the gap.

   For this, we have to create a Mentor & Mentee framework where each Sysprog in the organization is a mentor and a mentee, irrespective of their years of experience. Everyone should be ready to learn different skills from others. In this way, we have a diversified skill pool.

   When we are in a school or college, we learn many different subjects in parallel and do well in all. Why not here ?

2. Extend the mentorship program to application developers - Identify developers, interested on the systems area and spare some time to mentor them. They will be the silent reserve pool to meet your demand in time.

3. Prepare a roadmap to develop Sysprog skills in the Operations and Monitoring teams - I find them to be an excellent source of talent pool to take the role of Sysprogs with least effort.

4. Aging team members - We must target to tap their expertise as much as possible. I will suggest to make use of 40% of their time in last 2 years before retirement to mentor identified team members to enrich their expertise. Learning from experience is much faster than exploring yourself.

5. Build and hire from the universities - Talk to universities, identify talents, and start the learning process in their last year where you may have special coaching sessions for the group, provide necessary reading materials to prepare them to be ready and be a part of your team as soon as they graduate.

New zOS Sysprogs build from L0

When you start building zOS Sysprogs from Zero level, I will suggest to start with a 12 weeks class room training with labs. With my experience, include at least the following topics in the curriculum (not in order), which should have most of the topics, but may not be exhaustive though. You can always add on to it.

All these topics are the parts of a large jigsaw puzzle, which is the complex zOS system. So, focus on What, Why, Where, When, How etc. against each topic.

I always compare complexity of zOS with a human being which is made up of so many components, integrated together as a single entity and collaborate so harmoniously to run an extremely complex human machine so beautifully.

You can decide the number of hours to allocate against each topic so that you can cover all the end-to-end basics without going in-depth. Share your practical experiences and tips to understand the topics easily and create a beautiful picture of zOS in their mind. Provide sufficient reading materials (manuals, redbook, presentations etc.) for their reading and reference.

Basic hands-on labs, as applicable, should be provided during the sessions.

Make the sessions as interactive as possible. Face to face classroom training is the best.

Conduct written exams to assess their understanding, find the common gaps and organize additional sessions, if necessary.

You can also build yourself as a Sysprog thru self-study.

Recommended topics:

1. Introduction
2. zOS Sysprog roles and responsibilities
3. HW
4. IODF
5. HMC
6. LPAR configuration and resource allocation

7. Address space
8. Common areas
9. Systems address spaces
10. Virtual storage management
11. PSW
12. POR
13. IPL
14. TSO
15. ISPF/PDF
16. SDSF
17. Technical references
18. Console
19. System and sub-system messages and codes
20. Mainframe history
21. Latest z servers
22. zOS overview
23. zOS components
24. Latest zOS version
25. Sub-systems and non-IBM products
26. JCL & procedures
27. System Utilities
28. Parallel sysplex and GDPS
29. Coupling Facility
30. zOS Networking - VTAM and TCPIP and network security
31. Parmlib, Proclib members
32. TSO Logon Procedure
33. JES2
34. System IO
35. Storage (DASD and Tape), SMS & VSAM
36. ICF Catalog
37. zOS and JES2 commands
38. Automation
39. USS
40. zOSMF
41. Basic REXX and develop sample REXX programs
42. Clist and develop sample Clists
43. WLM concept
44. RMF concept

45. SMF concept
46. Problem diagnosis
47. Control blocks
48. Disaster Recovery
49. SMPE
50. Installation, Customization & Maintenance
51. RACF
52. Capacity, performance & optimization
53. Pervasive Encryption
54. Container Extension
55. IPCS
56. LLA and VLF
57. Logs and Logger
58. Workload running in the system (batch and online)
59. Linux on z
60. Cloud, AI and other modernizations under zOS
61. Systems operation and monitoring
62. Batch scheduling
63. Process and procedures
64. Prepare documentation
65. Other related topics, as necessary

zOS Modernizations

Automation, AI, Machine Learning, Cloud, Robotics, DevOps, Hybrid model etc. are the new buzzwords and you find a lot of discussions, developments, marketing, education, certifications etc. surrounding this in any mainframe forum and LinkedIn.

If you talk to the old Mainframer's, they don't get excited with all these cosmetics as the zOS system and its operation remains ASIS. But, it is the demand of time and must move on.

If mainframe is running so smoothly for decades, why this buzz in the last decade or so? When I think deeply, I find two root cause to this.

1. SKILL GAP
2. COST

This has resulted a silent war between two groups.

1. One group is trying hard to get rid of mainframe by any means. Some of the top decision makers, in the organization, get allergy in the name of mainframe.

2. And the other group is trying hard to preserve and continue with the Legacy by doing anything and everything in the name of 'Modernization'.

We all know the outcome of a war. But the IT evolution must continue.

It reminds me the 'Client/Server' revolution in 1990's when everyone thought, that was the future of IT. But we all know the outcome. Rather, many of us have nightmares in resolving many disasters caused by a small unknown and forgotten client sitting somewhere in a dark corner.

Do you remember 'Msys for operation' in early 2000 ?

SKILL GAP - The truth is, we created this by not taking timely action to build our next generation to continue with the Legacy seamlessly. In the past, we have built many in-house tools and customizations to make our mainframe operation efficient and effective. But today, most of them are ‘Black Boxes’ with no knowledge, documentation and even source code. We call them ‘Technical Debts’ and unable to get rid of most because they are integrated so well that it may cause disaster to the current operations. These are in thousands in old installations.

COST - Majority of the ‘Mainframe Cost’ is due to computing resources (MIPS or MSU or ???) and human resources (development and support). Has anyone tracked the historical trend of the growth of resources over years especially starting early 2000 ? Anyway, it is what it is today. But again, the top decision makers, in the organizations,’ are thinking to get rid of or pass on this ‘Headache of mainframe’ to someone else, in the name of ‘Modernization’.

Mainframe is one platform where programs developed before many decades still run today without any change.

And, zOS system never says ‘NO’ to any workload coming in. It may crawl in executing the workload when overloaded, but hardly breaks.

Mainframe exists since last six decades and will exist for many decades to come. So, we must create a long term roadmap to move forward from ‘Today’ to ‘Tomorrow’ seamlessly embracing the ‘Modernization’ and ‘Legacy’ together. And we must avoid building many more layers of ‘Technical Debts’, over what we have today, due to new technologies.

zOS Operational challenges

I will suggest here some of my experiences (good practices) to overcome many operational challenges.

1. Know your installation very well; Each and every system components, sub-systems and products running in your installation, the detailed customizations and all installation specific set ups such as USERMODs, EXITs etc. and documented up-to-date.

2. Have a detailed software inventory including installation developed tools, their current versions, EoM/EoS/EoL dates. Each and every item should have an owner. Must have a detailed up-to-date documentation.

3. Have a very well defined maintenance process; periodic, hipers, red alerts, emergency.

4. Design a very robust, efficient and optimized LPARs configuration, placement in CECs, resource allocation etc.

5. Have a detailed documentation of list of batch jobs, online transactions, MQ channels, network nodes and all other setups. Most importantly, each and every item should have an owner and support hierarchy.

6. There should be nothing called ‘I don’t know’ or ‘this is a history’ or ‘I am new here’ etc. The functional managers and team leads should have the relevant/related information available to them and others in most user friendly manner.

7. Define SLAs at all level and very strictly adhere to them. Update as necessary to meet the current business dynamics.

8. Have a robust business continuity and disaster recovery setup to make all the services available continuously as per SLA.

9. Plan your support resources. Have relevant talents in place and plan at least 2 to 3 years ahead. Should not have any talent gap in supporting your installation.

10. Bring up the Console and Batch monitoring skill level to ‘0.5 Sysprog’ and you will see the magical improvements in your system’s operation and resolving most of the L1 issues timely and fast.

11. Create well defined processes covering all areas and keep it reviewed/update at least yearly.

12. Create a user friendly problem database containing all history of at least last 5 to 10 years. One should be able to find the repetitive and installation specific recursive issues easily.

13. Have documentation at every level. An expert may leave the organization, but the daily mundane work should not suffer. Just follow the document to perform and deliver.

14. Must have a clear roles and responsibilities throughout the operational structure.

15. If you are in-house installation, make the Chargeback process very simple. You will save couple of resources here and lots of CPU cycles in generating unnecessary reports to address queries. A fixed charging model based on history will be the best.

16. Have regular Internal audits conducted very sincerely to identify any gaps anywhere and take actions to close them.

17. Have a strong CPO team to focus on optimization. Every CPU cycle counts.

18. Automate anything and everything possible without having any impact to your systems uptime and SLAs.

zOS Problem Diagnosis

As a Sysprog, PDPSI (Problem Diagnosis and Source Code Identification), the term I learnt in 1990s, has been the most enjoyable activity during my career, especially when I was in the support role at IBM Singapore supporting all MVS/OS390/zOS customers in Asia pacific. Solving many of the problems have been like solving murder mysteries.

To me, problems diagnosis is an art and you have to mature this skill over time. There is always a systematic approach to handle every problem.

I will not go to the technical details. You can refer to 'zOS Problem Management' and other component manuals for this.

My tips are not in order and you can apply it in any combination. They may appear discreet activities, but many go in parallel from the time of the problem occurrence till closing.

General:
1. Thorough installation knowledge - HW, SW, system's configuration (Prod, Test, Recovery, DR..), integration, business applications (online and batch), service levels, installation processes, escalation matrix, software vendors and their support contacts etc.
2. Know your team and their skill and expertise. This will equip you to reach to right person for help, if needed.

Problem diagnosis:

1. Handle all problems with 'Urgency'.
2. Understand the severity of the problem and act accordingly.
3. Collect a detailed symptom of problem, both internal and external. You need a list of right questions to ask to understand the details.
4. Never get panic. Stay cool and move forward following the process.
5. Don't think big, stay in context, you will reach the end very fast.
6. Engage your manager immediately on P2 and P1 problems.
7. Broadcast a message in your sysprog team. May be someone already have a solution or advice for you.
8. Do a quick search in your problem database and vendor support site. May be

a solution already exist there.
9. Do timely escalation, to management and customers (internal and external), audit, compliance and many other teams/organizations, as applicable.
10. Engage the vendor support (HW/SW), if needed.
11. Work on to normalize the service first before moving to perform detailed investigation.
12. Make sure to collect all documents for further investigation, if the root cause is not found/known - what, when, where, symptoms, error messages, logs, SMF records, logrec, dumps(if any), screen shots, RMFMON III datasets backup...
13. Issue relevant commands and capture their output.
14. Capture a console dump before you recycle an address space.
15. If need to IPL a LPAR, remember about stand alone dump, if needed.
16. Create problem record with the vendor support and provide all detail documentation for faster resolution.
17. Once root cause is found, do a proper documentation in your problem database for future reference by you and others.
18. Do share your learnings with other team members.
19. Implement preventive measures to avoid such problem recurring in the future.

Typical z/OS problems are classified by the following symptoms:

- **Abend** – an error or abnormal end of a program or job.
- **Wait or Hang** – a coded wait state is loaded or the system or a job appears hung or does not complete.
- **Loop** – the system or program executes infinitely typically using large or higher amounts of processor resource.
- **Incorrout** – there is incorrect or missing output from a program or job.
- **Performance** – processing is using too much system resource and impacting other parts or users of the system, or processes are taking too long.
- **Message** – an error is reported through a message to the operator or in a log.

Disaster Recovery

- Disaster recovery (DR) is the process of reacting to a disaster by being able to provide computing services from another location. In most cases, the counter measures you employ to recover from a disaster are entirely different from the solution you use to achieve high availability. In a disaster situation, users are aware that an outage occurred at the central computer facility, and the duration of the outage is dependent on the recovery solution. Usually we measure this duration in two ways: the time until computing services are once again available to the user and the period of time prior to the disaster event for which data is lost.

- Disaster Recovery (DR) is the ability to recover from unplanned outages at a different site, usually on different hardware. There are different levels of errors that result in some kind of recovery. Not all errors need to involve DR, but general IT recovery does not always include a site failover.

- A DR solution involves a whole company, not only the IT environment, but also human resources, production facilities, physical location, communication lines and more. IT environment involves the infrastructure such as LAN, SAN, WAN, data, servers, and applications.

- Many implement an IT environment with a DR solution based on a two-site solution, where data is mirrored between the two sites and where the application can move from one site to the other, in case of a total site failure. If a total site failure occurs, caused by a fire, flood, earthquake, or similar, it does not only affect your IT environment, but also all other facilities at the site. This could be offices, production machines, labs, communications lines and more. So a DR solution where only the IT is protected, might not be enough to protect your company. You may have your data and applications up and running, but you may not have any place where your employees can work with the data or keep production running.

RTO

- Recovery time objective (RTO)

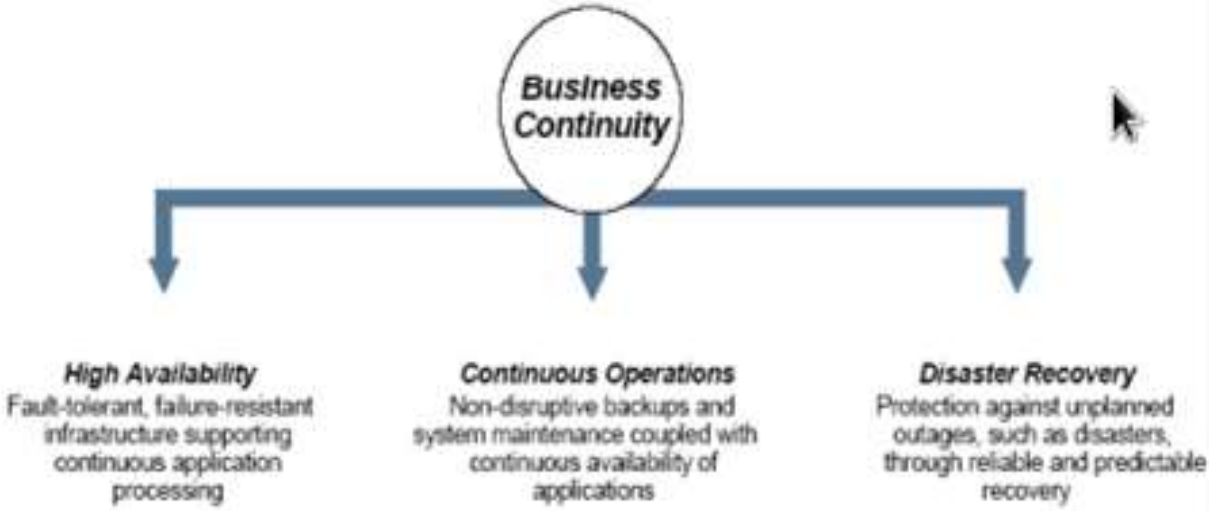
  - The time as measured from the time of application unavailability to the time of recovery (resuming business operations).
  - The Recovery Time Objective (RTO) is the duration of time following a disaster for which you are willing to accept the loss of computing services. The period starts from the moment of the disaster until the moment when systems are recovered. You can consider the RTO as a measure of how long a business can afford to have systems and applications down after a disaster. For example, a business that believes that it could afford to be without systems for eight hours has an RTO of eight hours.

RPO

- Recovery point objective (RPO)
  - The last data point to which production is recovered upon a failure. Ideally, customers want the RPO to be zero lost data. Practically speaking, we tend to accept a recovery point associated with a particular application state.
  - The Recovery Point Objective (RPO) represents the extent of data loss you are willing to accept due to a disaster. It is measured as the duration of time prior to the disaster event for which you must re-execute your work (or accept its loss) after your system is recovered.

Outages

Group	Possible outage types that an HADR solution might cover
Group 1	CPC (hardware: CPU and memory)
Group 2	Network or storage adapter failures, cable disconnects and so on External errors: storage errors, switch errors and so on
Group 3	Critical operating system resource: volume, file system, IP and so on
Group 4	Application, middleware, and operator actions
Group 5	Site outages



Business Continuity

High Availability
Fault-tolerant, failure-resistant infrastructure supporting continuous application processing

Continuous Operations
Non-disruptive backups and system maintenance coupled with continuous availability of applications

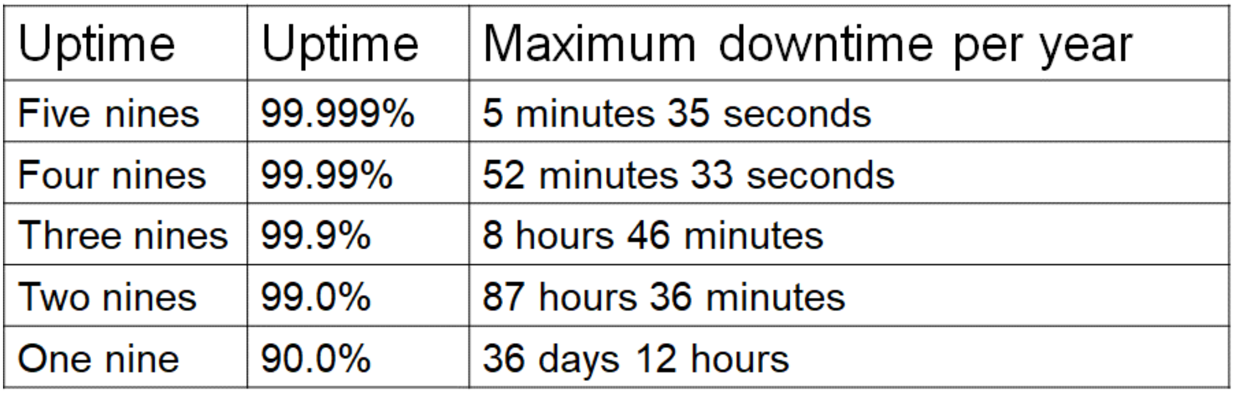
Disaster Recovery
Protection against unplanned outages, such as disasters, through reliable and predictable recovery

Protection of critical Business data

Recovery is predictable and reliable

Operations continue after a disaster

Costs are predictable and manageable

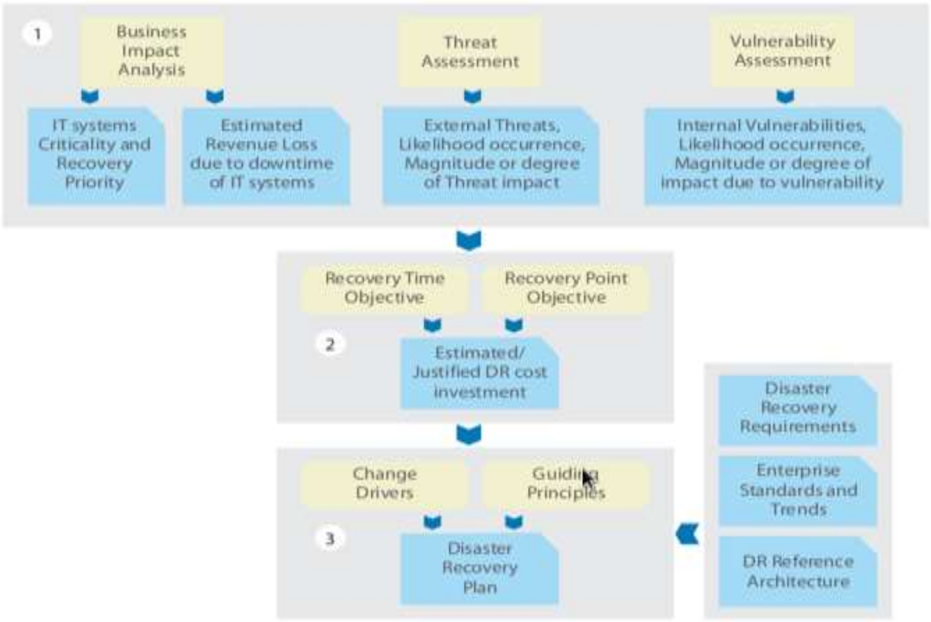


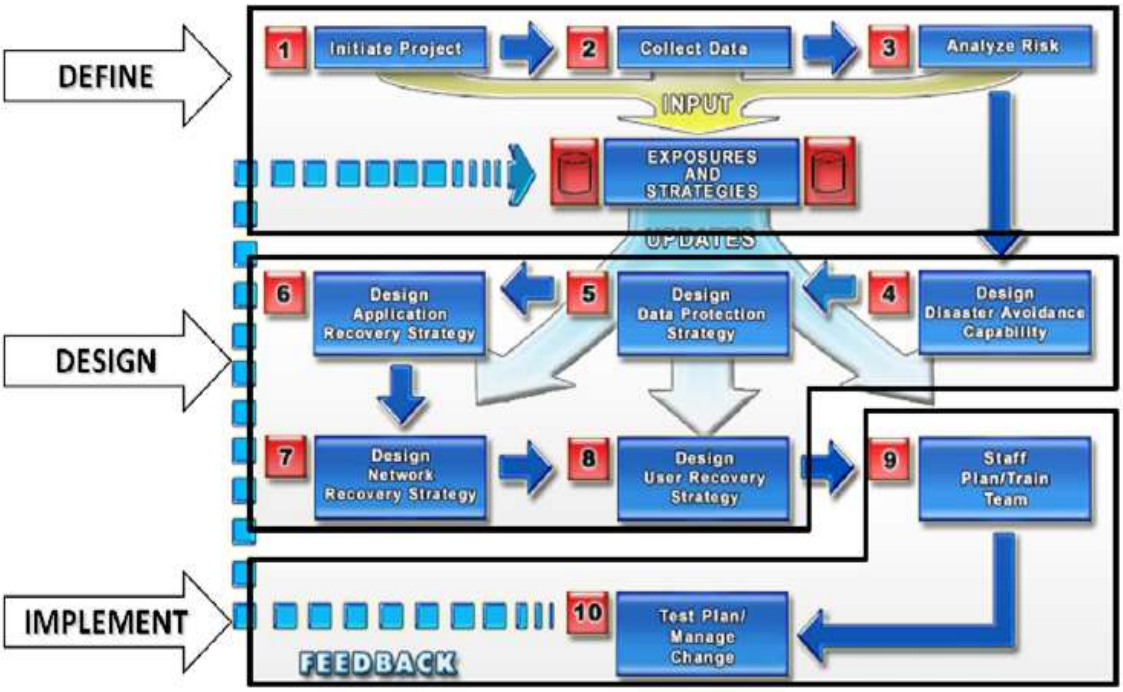
Uptime	Uptime	Maximum downtime per year
Five nines	99.999%	5 minutes 35 seconds
Four nines	99.99%	52 minutes 33 seconds
Three nines	99.9%	8 hours 46 minutes
Two nines	99.0%	87 hours 36 minutes
One nine	90.0%	36 days 12 hours

- Disaster in the IT environment refers to any incident which causes an unplanned outage or disruption to the business operations and supporting IT systems.
- Disaster can be caused due to natural calamity, catastrophic failures and / or human errors.
- Business Continuity Planning is a process of preparing the organization to restore and resume the business operations during an outage, which includes the key elements such as business office locations, people, information technology infrastructure, IT applications, data and business processes.
- Disaster Recovery planning is a subset of Business Continuity Planning. The objective of Disaster Recovery is to recover / restore the IT assets like infrastructure, business applications, business data and the supporting IT facilities from the potential disaster events based on their pre-defined recovery priorities.

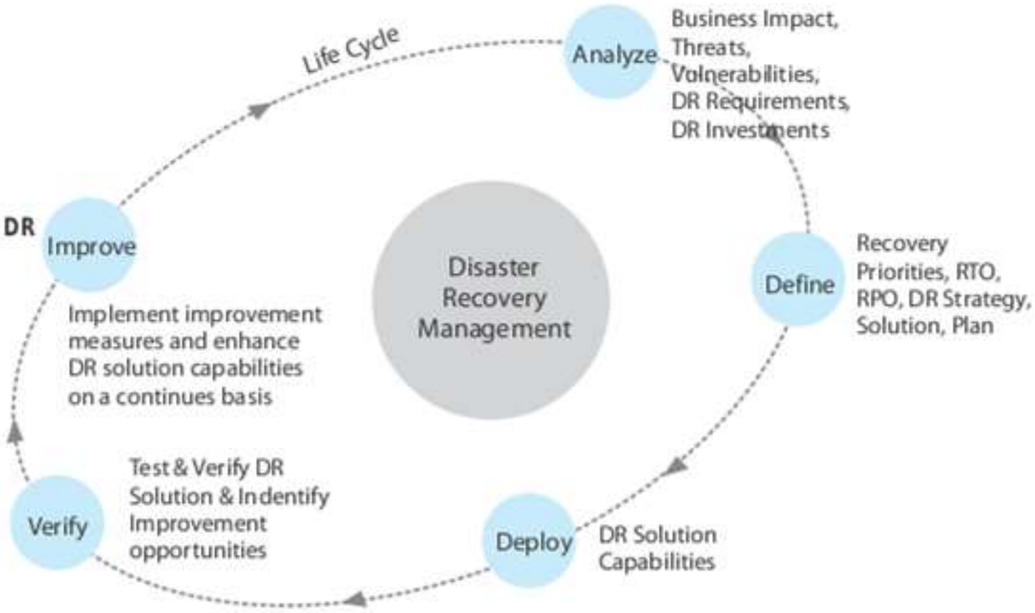
DR Plan

- Disaster Recovery plan usually contains the procedures, steps, people roles, responsibilities, escalation procedures, automation processes, applications, data, IT devices and recovery priorities that are necessary to restore the damaged mission critical IT systems and the peripheral IT infrastructure to resume business operations post unplanned outage.

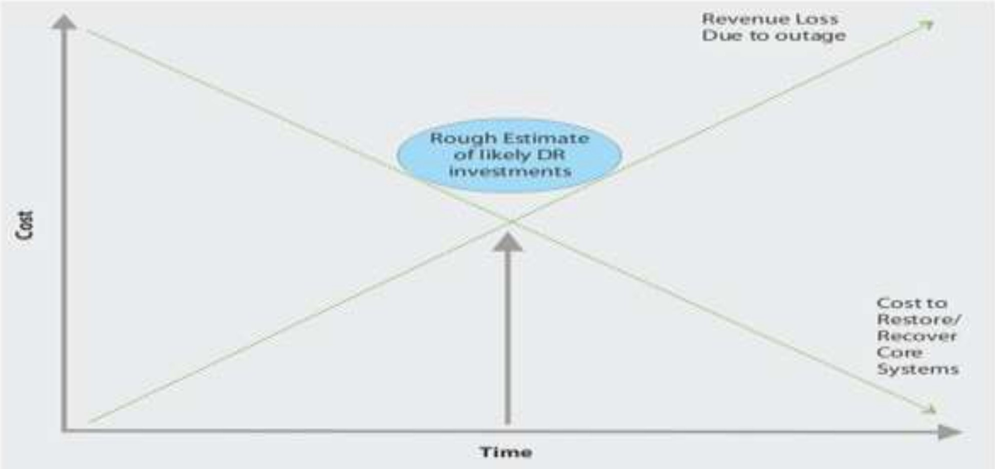

1
Business Impact Analysis
Threat Assessment
Vulnerability Assessment
IT systems Criticality and Recovery Priority
Estimated Revenue Loss due to downtime of IT systems
External Threats, Likelihood occurrence, Magnitude or degree of Threat impact
Internal Vulnerabilities, Likelihood occurrence, Magnitude or degree of impact due to vulnerability
Recovery Time Objective
Recovery Point Objective
2
Estimated/ Justified DR cost investment
Disaster Recovery Requirements
Enterprise Standards and Trends
DR Reference Architecture
Change Drivers
Guiding Principles
3
Disaster Recovery Plan


DEFINE
DESIGN
IMPLEMENT
1 Initiate Project
2 Collect Data
3 Analyze Risk
INPUT
EXPOSURES AND STRATEGIES
UPDATES
4 Design Disaster Avoidance Capability
5 Design Data Protection Strategy
6 Design Application Recovery Strategy
7 Design Network Recovery Strategy
8 Design User Recovery Strategy
9 Staff Plan/Train Team
10 Test Plan/ Manage Change
FEEDBACK

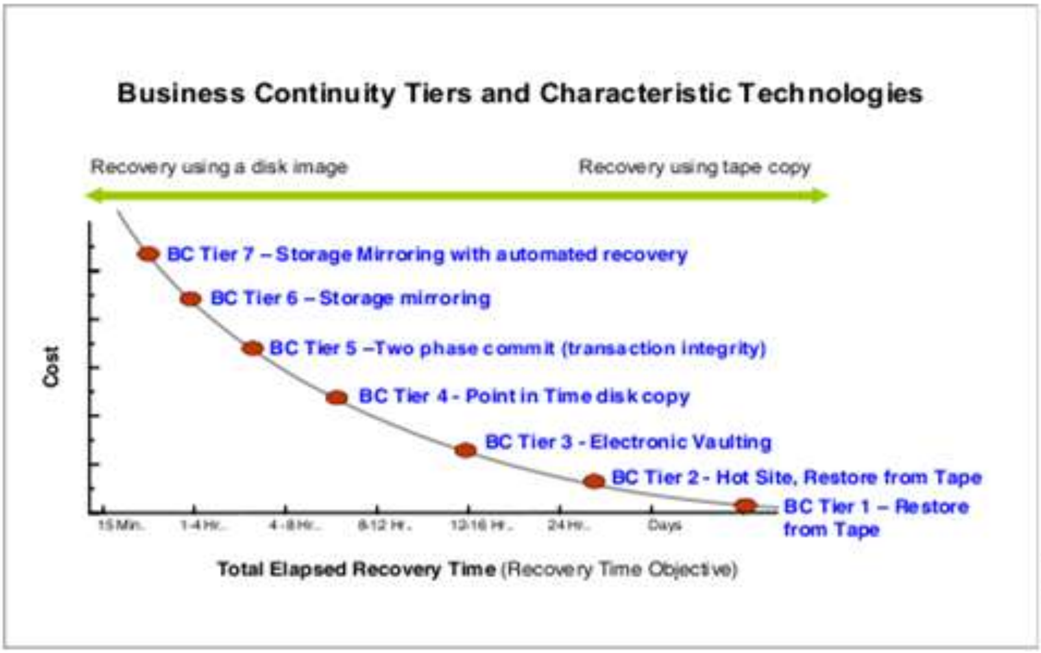
DR Management life cycle


Life Cycle
Analyze
Business Impact, Threats, Vulnerabilities, DR Requirements, DR Investments
Define
Recovery Priorities, RTO, RPO, DR Strategy, Solution, Plan
Deploy
DR Solution Capabilities
Verify
Test & Verify DR Solution & Indentify Improvement opportunities
DR Improve
Implement improvement measures and enhance DR solution capabilities on a continues basis
Disaster Recovery Management

DR Cost



Seven Tier DR



Tier 0

A business with a Tier 0 business continuity solution has no Business Continuity plan.

- There is no saved information, no documentation, no back-up hardware, and no contingency plan.
- The length of recovery time in this instance is unpredictable. In fact, it might not recover at all.

Tier 1

Business that uses Tier 1 Business Continuity solutions back up their data at an off-site facility. Depending on how often back-ups are made, they are prepared to accept several days to weeks of data loss, but their back-ups are securely stored off-site. However, this tier lacks the systems on which to restore data.

Examples for Tier 1 Business Continuity solutions / technologies are the Pickup Truck Access Method (PTAM), disk subsystem, or tape-based mirroring to locations without computing facilities, and using applications such as IBM Tivoli Storage Manager.

Tier 2

A business that uses a Tier 2 business continuity solution performs regular back-ups to tape. This is combined with an off-site facility and infrastructure (known as a hot site) in which they can restore computing services from those tapes in the event of a disaster. This tier still results in the need to recreate several hours to days worth of data, but it is less unpredictable in recovery time.

Examples for Tier 2 Business Continuity solutions and technologies are PTAM with a hot-site available and using applications such as IBM Tivoli Storage Manager.

Tier 3

Tier 3 solutions utilize components of Tier 2. Additionally, some mission critical data is electronically transferred to off-site storage. This electronically vaulted data is typically more current than data that is shipped through the PTAM. As a result there is less data recreation needed, or loss, after a disaster and thus allows an improved RPO.

Examples for Tier 3 Business Continuity solutions/technologies are electronic vaulting of data, and IBM Tivoli Storage Manager - Disaster Recovery Manager.

Tier 4

Tier 4 solutions are used by businesses who require both greater data currency and faster recovery than users of lower tiers. Rather than relying largely on transporting tape, as is common on the lower tiers, Tier 4 solutions begin to incorporate more disk-based solutions. Several hours of data loss is still possible, but it is easier to make point-in-time (PIT) copies with a greater frequency than can be achieved using tape-based solutions.

Examples for Tier 4 Business Continuity solutions/technologies are batch/online database shadowing and journaling, Global Copy, FlashCopy, FlashCopy Manager, Peer-to-Peer Virtual Tape Server, Metro/Global Mirror, IBM Tivoli Storage Manager - Disaster Recovery Manager, FlashCopy Backup/Restore for SAP Databases, and Integrated Backup for Databases.

Tier 5

Tier 5 solutions are used by businesses with a requirement for consistency of data between production and recovery data centers. There is little or no data loss in these solutions; however, the presence of this functionality is entirely dependent on the applications in use.

Examples for Tier 5 Business Continuity solutions and technologies are software and two-phase commits, such as DB2 remote replication, Oracle Data-Guard, and so on.

Tier 6

Tier 6 business continuity solutions maintain the highest levels of data concurrency and a very low RPO. They are used by businesses with little or no tolerance for data loss and who need to restore data to applications rapidly. These solutions reduce or eliminate the dependence on the application to provide data consistency.

Examples for Tier 6 Business Continuity solutions / technologies are Metro Mirror, Global Mirror, z/OS Global Mirror, GDPS HyperSwapTM Manager, Peer-to-Peer VTS with synchronous write, and PPRC Migration Manager.

Tier 7

Tier 7 solutions include all the major components being used for a Tier 6 solution with the additional integration of automation. This allows a Tier 7 solution to offer more consistency of data than Tier 6 solutions. Additionally, recovery of the applications is automated, allowing for restoration of systems and applications much faster and more reliably than is possible through manual business continuity procedures.

Examples for Tier 7 Business Continuity solutions and technologies are GDPS/PPRC with or without HyperSwap, GDPS/XRC, and AIX HACMP/XD with Metro Mirror.

zOS problems – Keep your eyes & ears opened

I will share a beautiful experience about a high severity-1 problem under OS/390 and how we found the root cause. Just read it as a story.

It was May 1999/2000 and I was with IBM Singapore. A non-sysplex banking customer in Indonesia had a Severity-1 problem where the CICS region suddenly experienced a degradation in transaction response followed by a system hung. All happened within one minute or so and occurred few times in couple of days. The Indonesian Government instructed the bank to get the best IBM support immediately onsite to investigate the problem.

Craig, from Australia, got delayed to travel. So, my manager told me to fly immediately to be onsite with the customer, till Craig arrived.

It was a public holiday in Singapore and I needed visa to travel to Indonesia. So, the Indonesian Government instructed the Embassy in Singapore to come to office immediately and issue me the visa.

With no time delay, I went to Indonesian Embassy and when I reached, 3 people were waiting and welcomed me with the greetings ‘you are the most wanted man here’. I got the visa stamped in 5 minutes and went straight to airport.

My manager had the agreement with IBM Indonesia to provide one IBMer as an escort during my stay there. I went straight from airport to the bank.

No SA dump was collected. WLM was not in goal mode. I spent the whole night looking at the SYSLOG, CICS log, recent dumps, LOGREP, RMFMON III, all related reports in RMF Postprocessor. I could not find anything obvious. Disappointed, we waited for the next occurrence.

Craig arrived next day. After a detailed discussion with him I left for Singapore.

After 2 days, during noon, Craig sent me a mail that, problem reoccurred and he managed to collect all documents and he planned to look at the dump after lunch.

Then after 30 minutes he sent me another mail saying that the root cause of the problem was found without him looking at any documents. It was a surprise to me.

It happened like this. Bank had a Shark DASD box which was shared between mainframe and Unix systems. All the system and CICS datasets were placed in this DASD box. An ‘Unix application’ was doing so much IO that exhausted the entire cache in the DASD box and mainframe could not do any IO due to unavailability of cache and went to hung state. Unfortunately, no IO related message in any log.

But how it was found this time and not earlier?

On the day of the problem, an IBM DASD engineer was running some periodic diagnostic on the box. Suddenly he noticed the cache in the box went to ZERO. After investigation, he found that, all IO came from Unix. When asked the Unix team, it was found to be a test application. That is how the root cause was identified.

We were lucky to have found the root cause so easily, otherwise would have taken long time in such cases. The Unix user did not have any impact as it was a test application, but mainframe had the disaster.

Cloud

"The cloud" refers to servers that are accessed over the Internet, and the software and databases that run on those servers. Cloud servers are located in data centers all over the world.

The five key terms that describe cloud computing are;

- XaaS (Anything-as-a-Service),
- SaaS (Software-as-a-Service),
- PaaS (Platform-as-a-Service),
- IaaS (Infrastructure-as-a-Service)
- BaaS (Backend-as-a-Service).

Cloud computing is an emerging style of information technology infrastructure designed for rapid delivery of computing resources.

This is potentially cost-efficient model for provisioning processes, applications, and services while making IT management easier and more responsive to the needs of the business.

Under this framework, services are delivered and made available in a simplified way. On-demand regardless of where the user is or the type of device they are using:

- Computation services.
- Storage services.
- Networking services.
- Whatever is needed.

Cloud computing is both a business delivery model and an infrastructure management methodology. The business delivery model provides a user experience by which hardware, software, and network resources are optimally utilized to provide innovative services over the web. Servers are provisioned in accordance with the logical needs of the service using advanced and automated tools. The cloud then enables the service creators, program administrators, and others to use these services through a web-based interface that abstracts away the complexity of the underlying dynamic infrastructure.

The infrastructure management methodology enables IT organizations to manage large numbers of highly virtualized resources as a single large resource. It also enables IT organizations to massively increase their data center resources without significantly increasing the number of people traditionally required to maintain that increase in data center resources.

The trend toward cloud computing started in the late 1980s with the concept of grid computing when, for the first time, a large number of systems were applied to a single problem, usually scientific in nature and requiring exceptionally high levels of parallel computation.

It is important to distinguish between grid computing and cloud computing. Grid computing specifically refers to leveraging several computers in parallel to solve a particular, individual problem, or to run a specific application. Cloud computing, on the other hand, refers to using multiple resources, including computing resources, to deliver a *service* to the end user.

The concept of cloud computing has evolved from the concepts of grid, utility and Software as a Service (SaaS). It is an emerging model through which users can gain access to their applications from anywhere, at any time, through their connected devices. These applications reside in massively scalable data centers where computing resources can be dynamically provisioned and shared to achieve significant economies of scale. Companies can choose to share these resources using public or private clouds, depending on their specific needs.

Public clouds expose services to customers, businesses, and consumers on the Internet. Private clouds are generally restricted to use within a company behind a firewall and have fewer security exposures as a result.

The strength of a cloud is its infrastructure management, enabled by the maturity and progress of virtualization technology to manage and better utilize the underlying resources through:

- Automatic provisioning.
- Re-imaging.
- Workload re-balancing.
- Monitoring.
- Systematic change request handling.
- A dynamic and automated security and resiliency platform.

IBM has recently introduced its vision of data centers which support a dynamic infrastructure that unifies the strengths of the web-centric cloud computing model and the conventional enterprise data center.

Cloud computing gives your business more flexibility. You can quickly scale resources and storage up to meet business demands without having to invest in physical infrastructure. Companies don't need to pay for or build the infrastructure needed to support their highest load levels.

One of the principal benefits of the cloud is outsourcing infrastructure management. Find a datacenter provider that hosts your mainframe workloads for you. This model can buy some time, reduce vendor lock in, and produce interim cost savings. Retire: Retire applications that are no longer needed before migration.

Amazon Web Services (AWS), Microsoft Azure, and Google Cloud Platform (GCP) are the big three cloud service providers today.

SYSMODs

There are different types of SYSMODs.

Function SYSMOD

These introduce a new product, a new version or release of a product, or updated functions for an existing product into the system. There are two types of function SYSMODs:

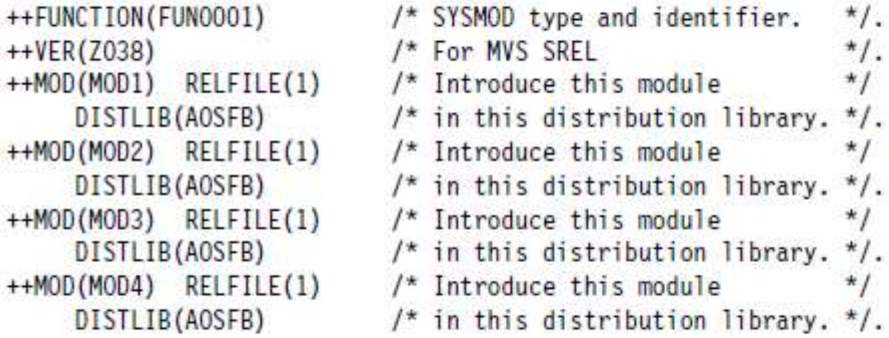
- A *base function* either adds or replaces an entire functional area in the system. It is a collection of elements (such as source, macros, modules, and CLISTs) that provides a general user function and is packaged independently from other functions. A base function is packaged as a function SYSMOD on a RELFILE tape, identified by an FMID (function modification identifier). The FMID is a 7-character identifier that needs to be unique to distinguish one product from another. Function SYSMODs for base functions are applicable to any z/OS environment, although they may have interface requirements that require the presence of other base functions. Examples of base functions are SMP/E and z/OS.

- A *dependent function* provides an addition to an existing functional area in the system. It is a collection of elements (such as source, macros, modules, and CLISTs) that provides an enhancement to a base function. It is called dependent because its installation depends on a base function already being installed. A dependent function is packaged as a function SYSMOD on a RELFILE tape, identified by an FMID. Function SYSMODs for dependent functions are only applicable to the parent base function. Each dependent function specifies an FMID operand on the ++VER MCS to indicate the base function to which it is applicable. Examples of dependent functions are the language features for SMP/E.

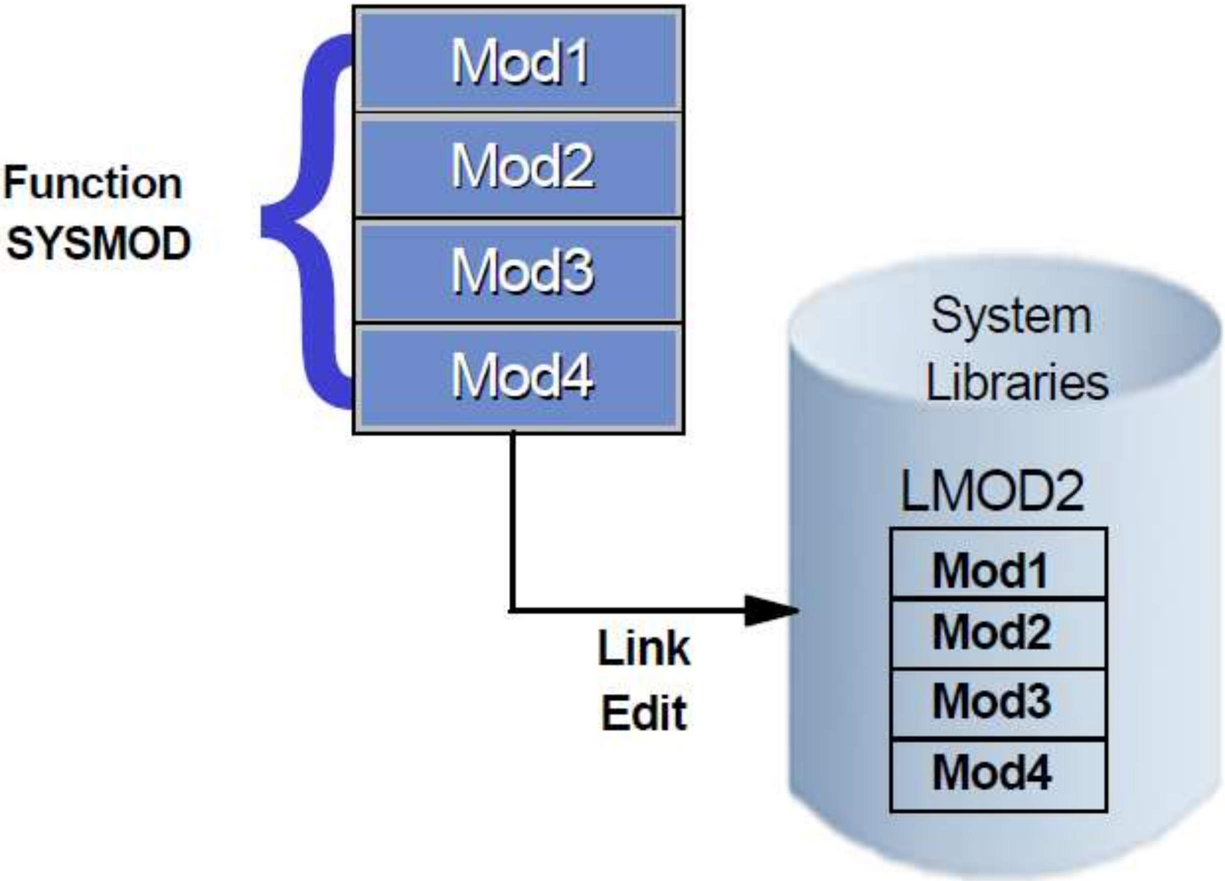
Introducing a Function

To introduce new elements into the system with SMP/E, you can install a function SYSMOD. The function SYSMOD introduces a new product, a new version or release of a product, or updated functions for an existing product into the system.

All other types of SYSMODs are dependent upon the function SYSMOD, because they are all modifications of the elements originally introduced by the function SYSMOD. Installing a function SYSMOD places all the product's elements in the system data sets, or libraries. Examples of these libraries are SYS1.LINKLIB, SYS1.LPALIB, and SYS1.SVCLIB. Figure 5-4 depicts the process of creating executable code in the production system libraries where the installation of a function SYSMOD link-edits object modules Mod1, Mod2, Mod3, and Mod4 to create load module LMOD2. The executable code created in load module LMOD2 is installed in the system libraries through the installation of the function SYSMOD.



```
++FUNCTION(FUN0001)         /* SYSMOD type and identifier.   */.
++VER(Z038)                 /* For MVS SREL                  */.
++MOD(MOD1)  RELFILE(1)     /* Introduce this module         */
     DISTLIB(AOSFB)         /* in this distribution library. */.
++MOD(MOD2)  RELFILE(1)     /* Introduce this module         */
     DISTLIB(AOSFB)         /* in this distribution library. */.
++MOD(MOD3)  RELFILE(1)     /* Introduce this module         */
     DISTLIB(AOSFB)         /* in this distribution library. */.
++MOD(MOD4)  RELFILE(1)     /* Introduce this module         */
     DISTLIB(AOSFB)         /* in this distribution library. */.
```



Service SYSMODs

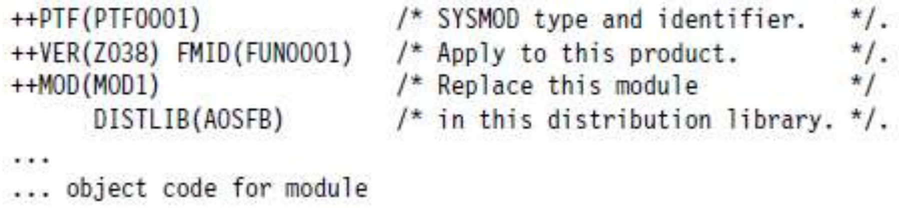
PTF, APAR, and USERMOD type SYSMODs are also known as service SYSMODs.

PTFs

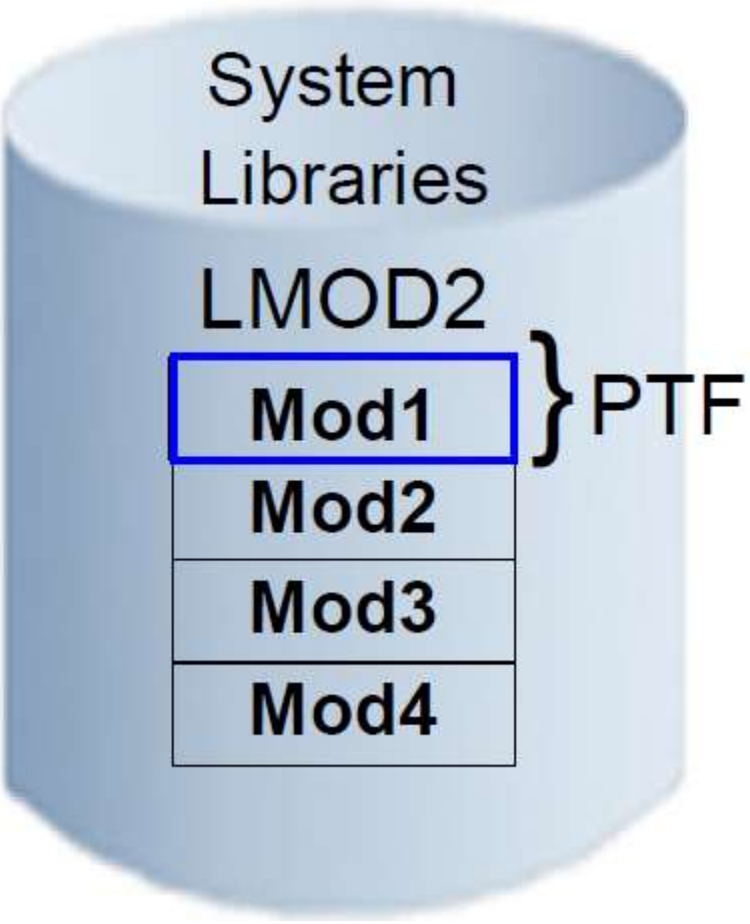
These are IBM-supplied, tested fixes for reported problems. They are meant to be installed in all environments. PTFs may be used as preventive service to avert certain known problems that may have not yet appeared on your system, or they may be used as corrective service to fix problems you have already encountered. The installation of a PTF must always be preceded by that of a function SYSMOD, and often other PTFs as well. Each PTF has a unique, 7-character name called a SYSMOD ID.

Fixing a problem with PTF

When a problem with a software element is discovered, IBM supplies its customers with a tested fix for that problem. This fix comes in the form of a program temporary fix (PTF). Although you may not have experienced the problem the PTF is intended to prevent, it is wise to install the PTF on your system. The PTF SYSMOD is used to install the PTF, thereby preventing the occurrence of that problem on your system. Usually, PTFs are designed to replace or update one or more complete elements of a system function. In Figure 5-5, we see a previously installed load module LMOD2. If we want to replace the element Mod1, we should install a PTF SYSMOD that contains the module Mod1. This PTF SYSMOD replaces the element in error with the corrected element. As part of the installation of the PTF SYSMOD, SMP/E relinks LMOD2 to include the new and corrected version of Mod1. PTF SYSMODs are always dependent upon the installation of a function SYSMOD. In some cases, some PTF SYSMODs may also be dependent upon the installation of other PTF SYSMODs. These dependencies are called *prerequisites* (a typical PTF prerequisite is shown in Figure 5-9 on page 304). Following is an example of a simple PTF SYSMOD MCS.



```
++PTF(PTF0001)              /* SYSMOD type and identifier.   */.
++VER(Z038) FMID(FUN0001)   /* Apply to this product.        */.
++MOD(MOD1)                 /* Replace this module           */
      DISTLIB(AOSFB)        /* in this distribution library. */.
...
... object code for module
```

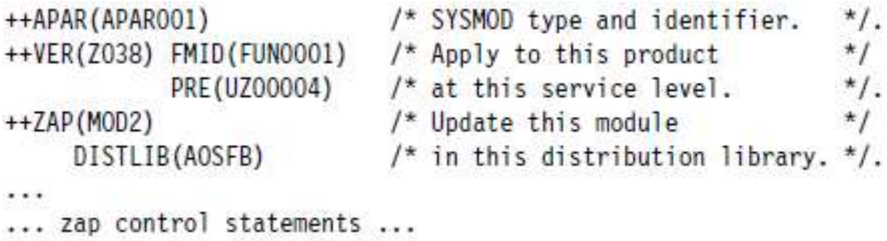


APAR fixes

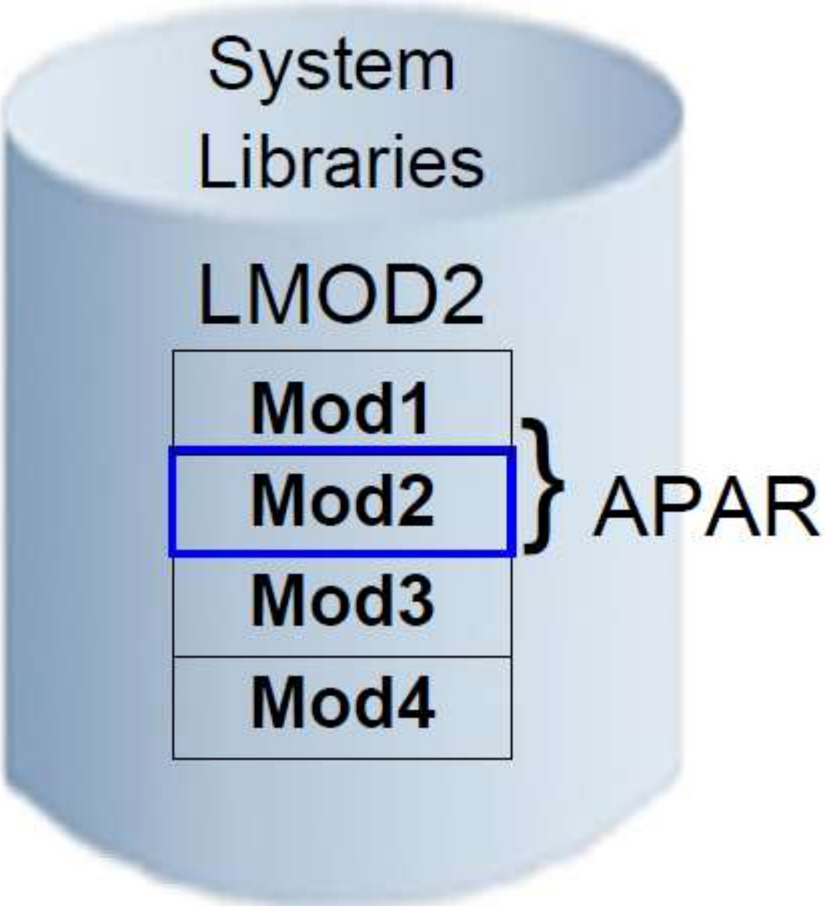
Authorized Program Analysis Reports (APARs) are temporary fixes designed to fix or bypass a problem for the first reporter of the problem. These fixes may not be applicable to your environment. The installation of an APAR must always be preceded by that of a function SYSMOD, and sometimes of a particular PTF. That is, an APAR is designed to be installed on a particular preventive-service level of an element. Each APAR has a unique, 7-character name called a SYSMOD ID.

Fixing a problem with APAR

You may sometimes find it is necessary to correct a serious problem that occurs on your system before a PTF is ready for distribution. In this situation, IBM supplies you with an Authorized Program Analysis Report (APAR). An APAR is a fix designed to quickly correct a specific area of an element or replace an element in error. You install an APAR SYSMOD to implement a fix, thereby updating the incorrect element. In Figure 5-6, the highlighted section is an area of Mod2 containing an error. The processing of the APAR SYSMOD provides a modification for object module Mod2. During the installation of the APAR SYSMOD, Mod2 is updated (and corrected) in load module LMOD2. The APAR SYSMOD always has the installation of a function SYSMOD as a prerequisite, and can also be dependent upon the installation of other PTF or APAR SYSMODs. Some APARs do not change modules; instead, they register problems while the final solution is being developed. Another APAR function is to give tips about customization and to alert for probable customer errors. Following is an example of a simple APAR SYSMOD.



```
++APAR(APAR001)            /* SYSMOD type and identifier.   */.
++VER(Z038) FMID(FUN0001)  /* Apply to this product         */
            PRE(UZ00004)   /* at this service level.        */.
++ZAP(MOD2)                /* Update this module            */
     DISTLIB(AOSFB)        /* in this distribution library. */.
...
... zap control statements ...
```

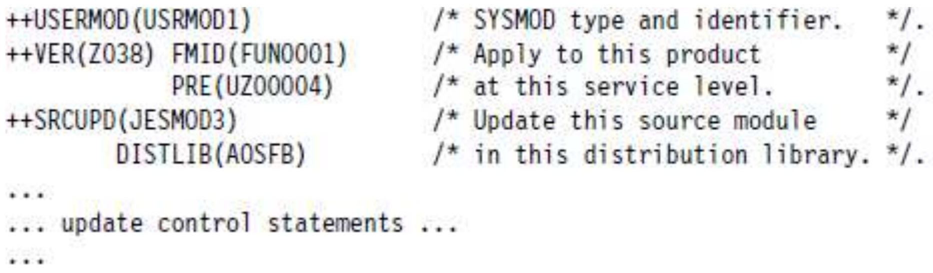


User modifications (USERMODs)

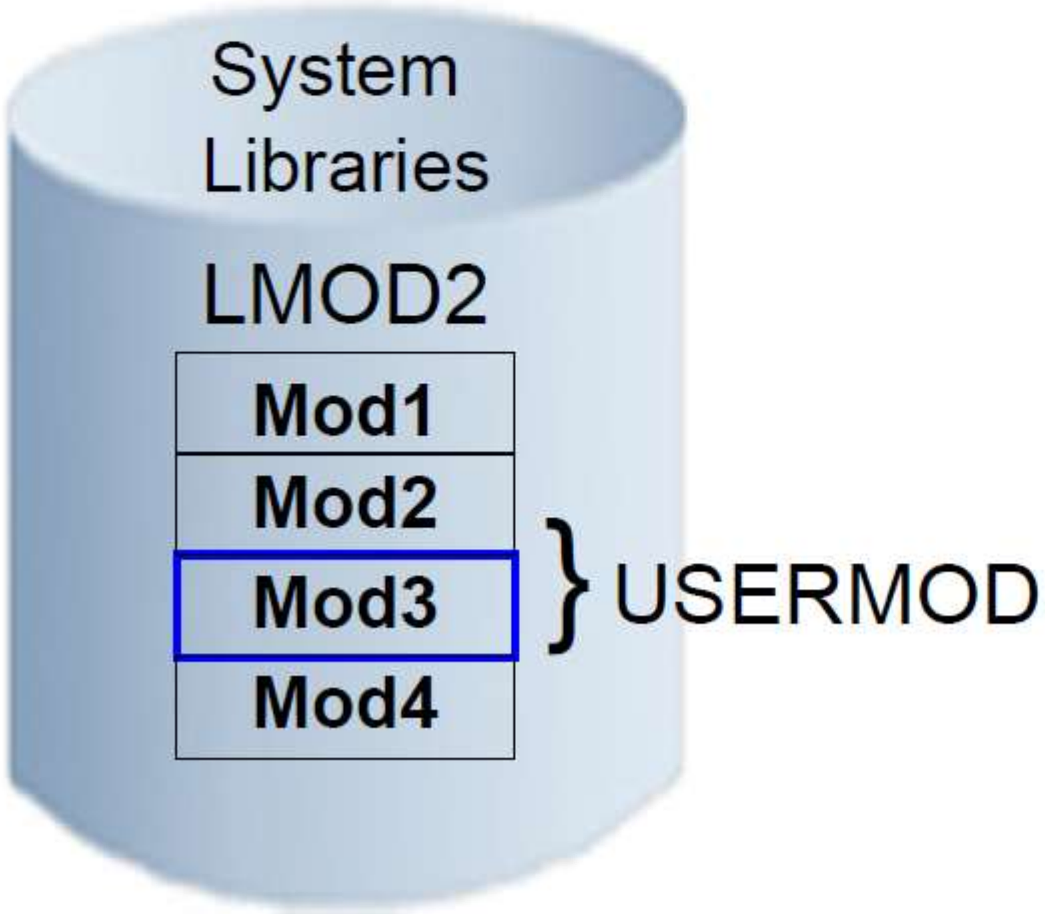
These are SYSMODs built by you, either to change IBM code or to add independent functions to the system. The installation of a USERMOD must always be preceded by that of a function SYSMOD, sometimes certain PTFs, APAR fixes, or other USERMODs. Each USERMOD has a unique, 7-character name called a SYSMOD ID.

Customizing an Element

If you have a requirement for a product to perform differently from the way it was designed, you might want to customize that element of your system. IBM provides you with certain modules that allow you to tailor IBM code to meet your specific needs. After making the desired changes, you add these modules to your system by installing a USERMOD SYSMOD. This SYSMOD can be used to replace or update an element, or to introduce a totally new user-written element into the system. In either case, the USERMOD SYSMOD is built by you either to change IBM code or to add your own code to the system. In Figure 5-7, Mod3 has been updated through the installation of a USERMOD SYSMOD. Prerequisites for USERMOD SYSMODs are the installation of a function SYSMOD, and possibly the installation of other PTF, APAR, or USERMOD SYSMODs. Following is an example of a simple USERMOD SYSMOD.



```
++USERMOD(USRMOD1)             /* SYSMOD type and identifier.   */.
++VER(Z038) FMID(FUN0001)      /* Apply to this product         */
            PRE(UZ00004)       /* at this service level.        */.
++SRCUPD(JESMOD3)              /* Update this source module     */
        DISTLIB(AOSFB)         /* in this distribution library. */.
...
... update control statements ...
...
```



Capacity, Performance and Optimization (CPO)

You can download read my book on this topic at link

https://www.linkedin.com/groups/2318931?q=highlightedFeedForGroups&highlightedUpdateUrn=urn%3Ali%3AgroupPost%3A2318931-7061904453267836928&lipi=urn%3Ali%3Apage%3Ad_flagship3_profile_view_base_recent_activity_content_view%3BI7Ie7BPeRISE%2BfihyIz3Nw%3D%3D.

If the link does not work, you can find it under my postings in LinkedIn.

- **Capacity -** Planning, estimating, and provisioning sufficient resource capacity in a cost-effective manner to meet the current and future service needs for all applications and users hosted under z/OS.

- **Performance -** To make the best use of the current resources to meet the stated objectives and Service Level Agreements (SLA).

- **Optimization -** To understand what is going on in the systems, identify the bottlenecks and processing inefficiencies through resource utilization analysis, real time monitoring, and identifying opportunities to define objectives to reduce resource usage and save costs.

MIPS

MIPS (Millions of Instructions per Second), is probably the most common unit used when talking about mainframe capacity. When mainframes were still young, manufacturers could measure MIPS capacity by repeatedly running a small standard routine. However, MIPS has not been a meaningful measurement for decades. IBM mainframes have a huge number of instructions; some are simple and quick, and others are complicated and slow. For example, one application using five million simple instructions will use a lot less CPU than one using five million complicated ones. In addition, the number of instructions available is increasing with each new mainframe processor type. In summary, MIPS is an indication of the speed of a processor and is very much workload dependent. IBM publishes the MIPS rating of its various processor models, but there is no single MIPS number for any CEC and no tool that reports MIPS numbers.

CPU Second

Traditionally **CPU seconds** - the number of seconds a CPU is actually in use - was a measurement of how much work is performed by a processor. z/OS (Mainframe OS) records the number of CPU seconds that each unit of work has consumed, providing an excellent way of measuring workload consumption for billing. However, **the amount of work that can be done in one CPU second is not the same for each processor type**. For example, one CPU second used by a z15 processor type is different to a CPU second used by a z16 processor type due to differing clock speeds. This makes the measurement inconsistent when upgrading mainframe hardware from one processor type to another.

Service Unit

SU (Service Units) allows cross processor type comparisons. IBM publishes SU ratings, which differ by processor model, even under the same processor type. Further complications arise when a CEC has multiple LPARs with a varying number of engines allocated to each LPAR. The SU_SEC for an LPAR is determined at system start time and is based on the number of engines online to that LPAR. For example, in a z14 CEC with 7xx model, one LPAR with 2 engines delivers 88397 SU per second while another LPAR with 9 engines delivers 76555 SU per second even though they are hosted in the same CEC. Therefore, this is not a good metric for CPU measurement and billing.

MSU (Million Service Units per hour)

MSU was created from Service Units. MSU measures the rate of CPU usage, but can also refer to the capacity of the processor model, e.g., a processor with an MSU rating of 100 can process up to 100 million service units per hour.

The original idea of MSUs was to use it as an indicator of CPC capacity. MSU is calculated as:

MSU = The SU/Sec for the model * number of engines * 3600 / 1,000,000

Example: The z15 701 processor model has a rating of: 103225.8065 SU/Sec * 1 * 3600 / 1,000,000 = 371.6 or 372 MSUs

MSU is primarily intended for software licensing. Most vendors scale their software licensing fees to an MSU rating. IBMs sub-capacity licensing also uses MSU when calculating the final bill.

Understanding MSU (Hardware and Software)

To help customers reduce software licensing charges, post the introduction of the z990 processor type, IBM began tweaking the MSU capacity of processors (in order to lure users to a newer machine). IBM publishes the MSU ratings of the various processor models based on their internal measurements and adjustments, which are only used for software license charging.

When IBM started altering the MSU number as a way of discounting software cost, it gave rise to two definitions for MSU:

"Hardware" MSU – calculated using the original formula above, this is the basis for measuring and reporting in some of the mainframe usage.

"Software" MSU – also based on the calculation above, but adjusted depending on the CPU type and model and is a fixed value for a specific model. SW MSU is used as the basis for software charging and LPAR capping and is used in reporting some of the CPU measurements.

Example: The z15 701 is rated at 253 "Software" MSUs against 372 "Hardware" MSU, as calculated above.

In short, the "Hardware" MSU is an indicator of capacity while "Software" MSU is used for SW license charge and billing.

Program execution

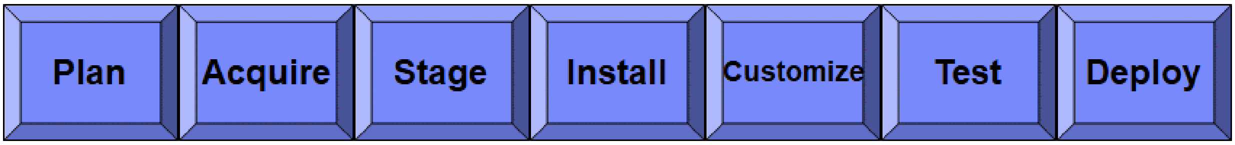
Mainframe has processors running at certain speeds. Which means, each processor delivers a certain number of CPU cycles and executes a certain number of instructions per second. This, in general, is measured in MIPS (millions of instructions executed per second). Fundamentally, a processor (CPU) does this:

1. Fetch the instruction
2. Decode the instruction
3. Fetch the operand
4. Execute the instruction
5. Store the result
6. Go to next instruction

With this, the processor is in an infinite loop just performing ‘Fetch and Execute’. Who tells the processor what to execute? The simple answer is ‘our programs’. We write the programs in COBOL, PL/I, Assembler, Java etc., compile and link edit to create load modules. When executed (e.g., batch job or CICS transaction), it calls for systems services, which are again programs, supplied by the operating system and software products. During processing, it executes a certain number of instructions that are accounted for as: how many MIPS the program or transactions have consumed. If our program can somehow result in executing a smaller number of instructions, this will account to less MIPS and directly result in saving $$$. So, the big question is, do we have a control over our programs to execute a smaller number of instructions? The answer is certainly YES. We are the creator of the program and we have options available at each step (coding, compiling, link editing etc.) to optimize our program. This is called the **art of creating optimized and efficient programs,** which will result in executing a smaller number of instructions in total to process the same amount of data and generate the same output.

zOS Installation and Maintenance

Installation Planning



Plan Acquire Stage Install Customize Test Deploy

1. Plan what hardware and software products and features are needed or desired
2. Acquire the products and features
3. Stage the software
4. Install the software
5. Customize the software
    - Configure features, override defaults (if necessary)
    - Migrate existing customization and perform required migration actions
    - Install/Connect middleware, ISV code, and applications
6. Test the system
7. Deploy the system
    - To other test systems, then to production systems

Note: Steps can involve multiple people with different responsibilities (roles)

SMPE Installation

- SMP/E is the utility used on the z/OS platform to install and maintain all IBM and some ISV software products.

- Whether you are installing a new release of a product or installing fixes or patches (known as maintenance), we start with an SMP/E Installation Package – either a SYSMOD or HOLDDATA.

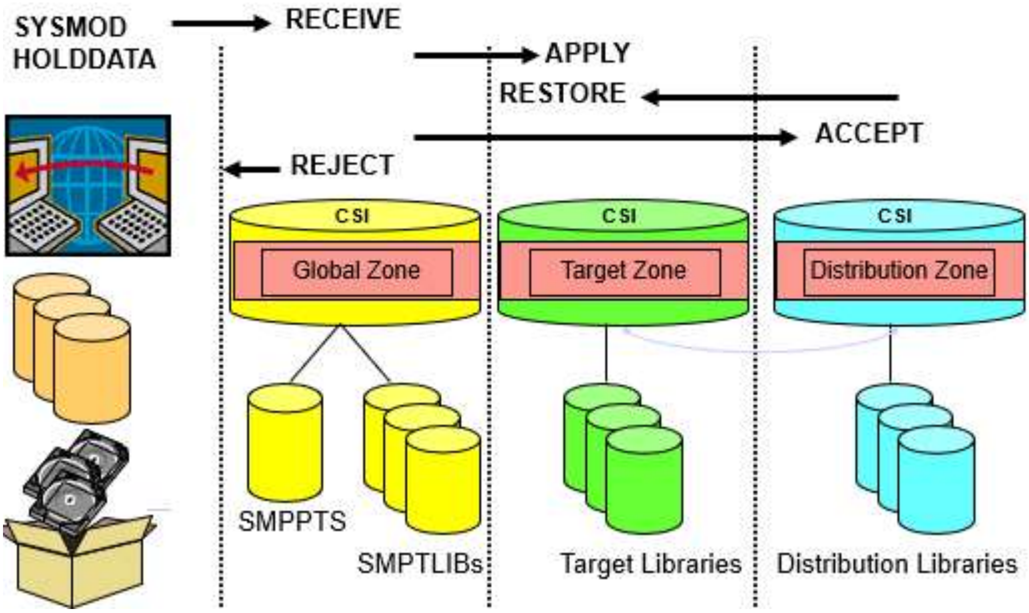
- The install package is used as input to SMP/E and can be delivered electronically. It can be stored on DASD locally, or it could come off a tape. The packages are loaded into SMP/E via the RECEIVE command where SMP/E will record the information about the SYSMOD in the Global zone, store the SYSMOD itself into the SMPPTS data set associated with the Global Zone, and load any RELFILES into SMP/E TLIB data sets.

- If for any reason the SYSMOD is not required to be installed, you can use the SMP/E REJECT command to remove the SYSMOD and/or HOLDDATA.

- To proceed with the installation, we use the SMP/E APPLY command which will move the elements from the GLOBAL zone to the TARGET zone. It does this by calling various system utilities.

- If for any reason the SYSMOD has to be removed from the Target zone, you can use the RESTORE command. In this process the elements are moved from the Distribution Zone back to the Target Zone. Note however that a suitable “synch point” must have already been created in the Distribution Zone before a SYSMOD can be restored.

- Once you have tested the new SYSMODs and are happy that the maintenance/product is performing as desired, then the SMP/E ACCEPT command can be run which moves the elements from the GLOBAL zone to the Distribution zone.



Installation Cloning

Some customers can have more than one cloned image. It is quite common to have this situation where you have 3 environments:

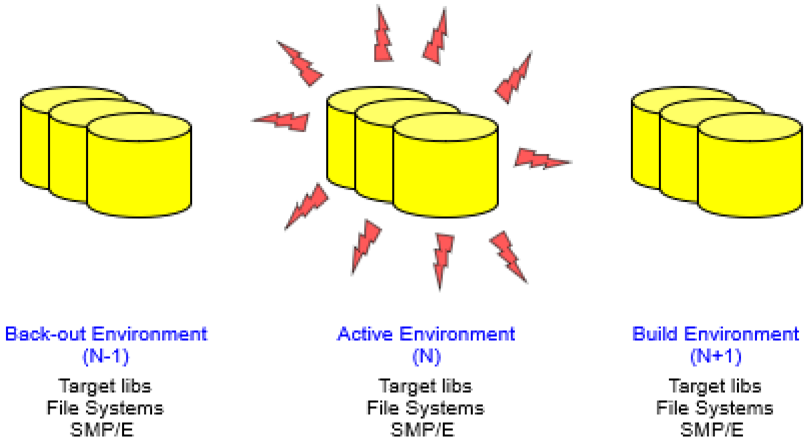
1. The Active Environment - known as N

2. The Back-Out Environment – known as N minus 1 (N – 1)

    – Always available to back-out the current active environment if required

3. The Build Environment (N + 1)

This is the environment that is currently being built to be implemented in the future



Maintenance Types

- **Corrective Maintenance**
  - Fix problems that <u>you</u> have experienced
- **Preventative Maintenance**
  - Install fixes for problems that <u>others</u> have experienced
  - Enables you to avoid experiencing those problems
- **Toleration Maintenance**
  - Required PTFs, needed for the function to work
  - "Recommended PTFs" - fixes to known problems related to the new function
  - Hardware
  - Software
- **HIPER Maintenance**

  - High impact PTFs
  - Additional symptom flags can also be added to help identify why the APAR is HIPER such as:
    - Loss of Data,
    - System Outage,
    - Loss of Major function or Subsystem,
    - Severe Performance Impact
  - Some Independent Software Vendors now using the same term
  - Sometimes written at "H**Y**PER"

- **Integrity Maintenance**
  - Programs run authorized or non-authorized
  - Non-authorized programs should not be able get authorized
  - This is the basis of the integrity of z/OS's security
  - When maintenance that fix integrity problems are called Integrity Maintenance

- **PUT maintenance**
  - PTFs that closed in a given month (identified after the month ends)
  - PUTyymm yy = year mm=month
  - PUT0901 means PTF was closed January 2009

- **RSU/CST maintenance**
  - Customer-like Parallel Sysplex environment used to test latest preventive maintenance
    - Online and batch workloads

        - Specific testing for z/OS, CICS, IMS, DB2, MQ, and WAS
        - Steady state and stress testing
    - Stress and saturation testing, failure and recovery testing, and rolling IPL maintenance test of previous Quarterly RSU to current Quarterly RSU
    - Testing covers integrated workloads across all participating z/Series and S/390 products
        - Provides one clear consistent recommendation for the platform
        - Workloads evolving over time as new products are introduced and customer input is evaluated
    - CST environment available to product/element owners for additional testing
- **Level Set PTF / SP Level**
    - Many fixes combined together into a single major fix or level
    - Example: - Windows XP SP1, SP2 and SP3 levels
- **ADHOC maintenance**
    - Occasionally used to describe corrective maintenance, given the ADHOC nature of corrective maintenance.
- **Dependent maintenance**
    - Similar toleration maintenance
    - One PTF has to be installed on all sharing systems before the dependent PTF can be installed
- **Cumulative Maintenance**
    - The “cum-tape” as it was known as was a tape containing multiple months of PUT maintenance.

- **Verified/Unverified Fixes**
  - These terms are used by ISVs to classify whether their fixes have actually verified by the reporting customer
- **Test Fix**
  - A test fix could be a unverified fix or it could be a special fix similar to IBM's APAR fix.
- **PSP maintenance**
  - **The hardware buckets identify**
    - Required service for device support or exploiting specific hardware functions.
    - Additional recommended service
  - **The software buckets identify:**
    - Recommended service to be installed when you install the software
    - Additional information on installation, documentation, or about the product
  - **The functional buckets**
    - The functional buckets identify the list of Service Recommendations needed for the function
- **FIX categories (FIXCAT)**
  - New with SMP/E V3.5
  - Associates fixes with one or more fix categories
  - Used to ensure resolving PTFs are for the fix categories that you are interested in as installed when you install a new FMID
  - Can be used to install preventive service that includes PTFs for categories that you are interested in

- Can be used to report on missing fixes for fix categories that you are interested in

    - The SMP/E REPORT MISSINGFIX command identifies missing fixes, and will create a sample job to acquire (RECEIVE) any PTF not already staged and install (APPLY) the missing fixes.

Typical Maintenance steps

1. Clone the environment (either SYSRES or product)

2. Prepare the cloned environment

    1. Mount UNIX filesystems at /SERVICE mount point

3. Stage maintenance

    1. If using SMP/E RECEIVE ORDER, the maintenance may already be RECEIVED into your GLOBAL CSI.

4. Receive maintenance into GLOBAL (using clone)

5. Perform APPLY with CHECK (using clone)

6. Resolve any error conditions and re-run APPLY with CHECK

7. Perform APPLY (using clone)

8. IPL clone and verify maintenance functions ok

9. Clone and prepare next maintenance environment

10. Perform ACCEPT with CHECK

11. Resolve any error conditions and re-run ACCEPT with CHECK

12. Perform ACCEPT

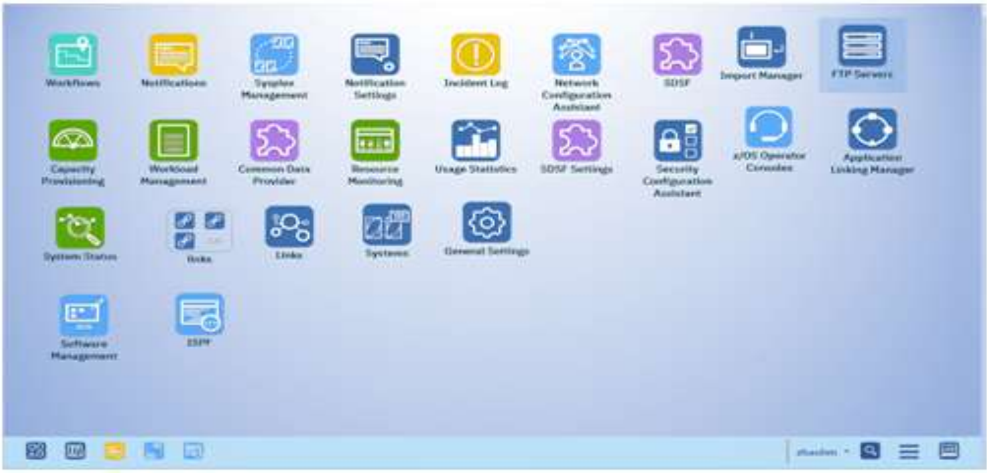
z/OSMF

IBM z/OS Management Facility (z/OSMF) provides system management functions in a task-oriented, web browser-based user interface with integrated user assistance so that you can more easily manage the day-to-day operations and administration of your mainframe z/OS systems. By streamlining some traditional tasks and automating others, z/OSMF can help to simplify some areas of z/OS system management.

z/OSMF provides a framework for managing various aspects of a z/OS system through a web browser interface. z/OSMF provides you with a single point of control for:

- Viewing, defining, and updating policies that affect system behavior
- Monitoring the performance of the systems in your enterprise
- Managing software that runs on z/OS
- Performing problem data management tasks
- Consolidating your z/OS management tools.

z/OSMF allows you to communicate with the z/OS system through a web browser, so you can access and manage your z/OS system from anywhere. Multiple users can log in to z/OSMF using different computers, different browsers, or multiple instances of the same browser.



Zowe

Zowe is an open source software which provides both an extensible framework, and a set of tools that allow mainframe development and operation teams to securely manage, develop, and automate resources and services on z/OS family mainframes. Zowe offers modern interfaces to interact with z/OS and allows users to interact with the mainframe system in a way that is similar to what they experience on cloud platforms today. Users can work with these interfaces as delivered or through plug-ins and extensions created by customers or third-party vendors. All members of the IBM Z platform community, including Independent Software Vendors (ISVs), System Integrators, and z/OS consumers, benefit from the modern and open approach to mainframe computing delivered by Zowe.

Zowe is a member of the Open Mainframe Project governed by Linux Foundation.

Please visit https://www.zowe.org for more details.

Conclusion

I have provided here some of my notes kept in the scratchpad over many years. I have just put them together with a hope that, it may help you in building up or refreshing your zOS basics knowledge.