

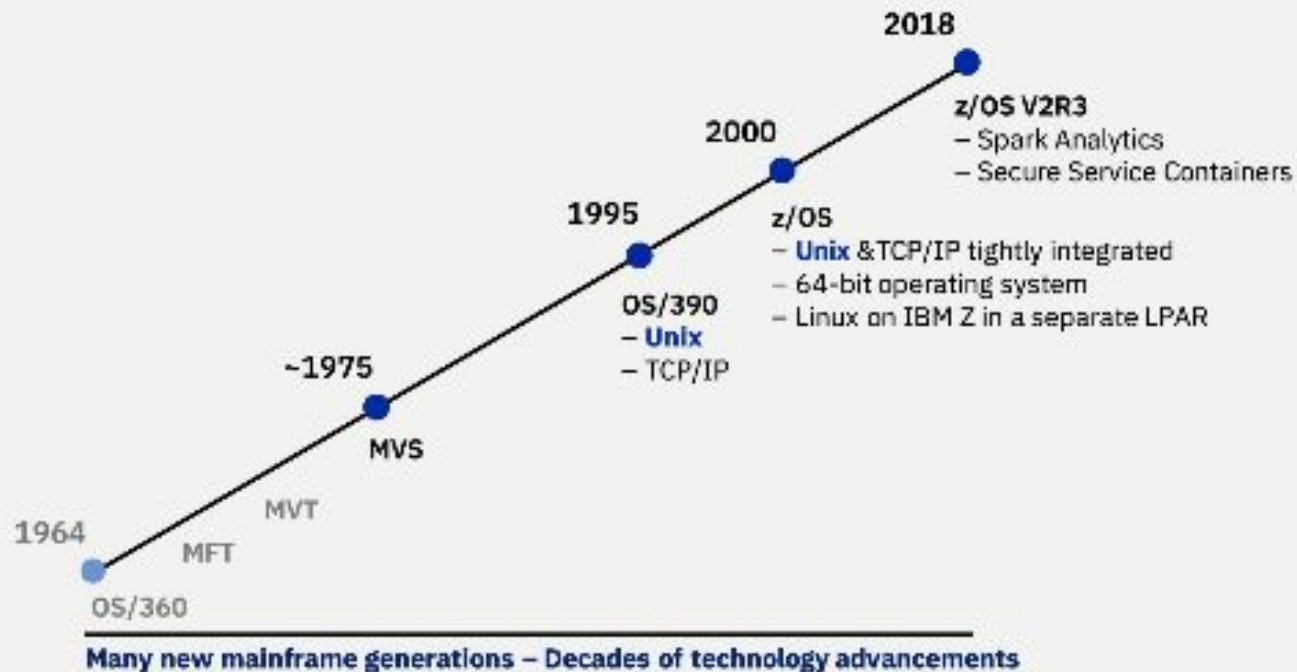
Unit 5:

Operating System Environment

- Unix System Services (USS),
- JCL - Job Control Language,
- JES - Job Entry Subsystem,
- System Catalog,
- Components, Messages, System Log,
- Virtual Storage and Address Spaces
- System and User Address Space Management,
- Key Controlled Protection for Address Spaces,

Unix System Services (USS)

z/OS = MVS + Unix



IBM Z

What is z/OS UNIX?

- The z/OS UNIX shell and utilities provide an interactive interface to z/OS.
- The shell and utilities can be compared to the TSO function in z/OS.
- To perform some command requests, the shell calls other programs, known as **utilities**.
- **The shell can be used to:**
 - Invoke shell scripts and utilities.
 - Write shell scripts (a named list of shell commands, using the shell programming language).
 - Run shell scripts and C language programs interactively, in the TSO background or in batch

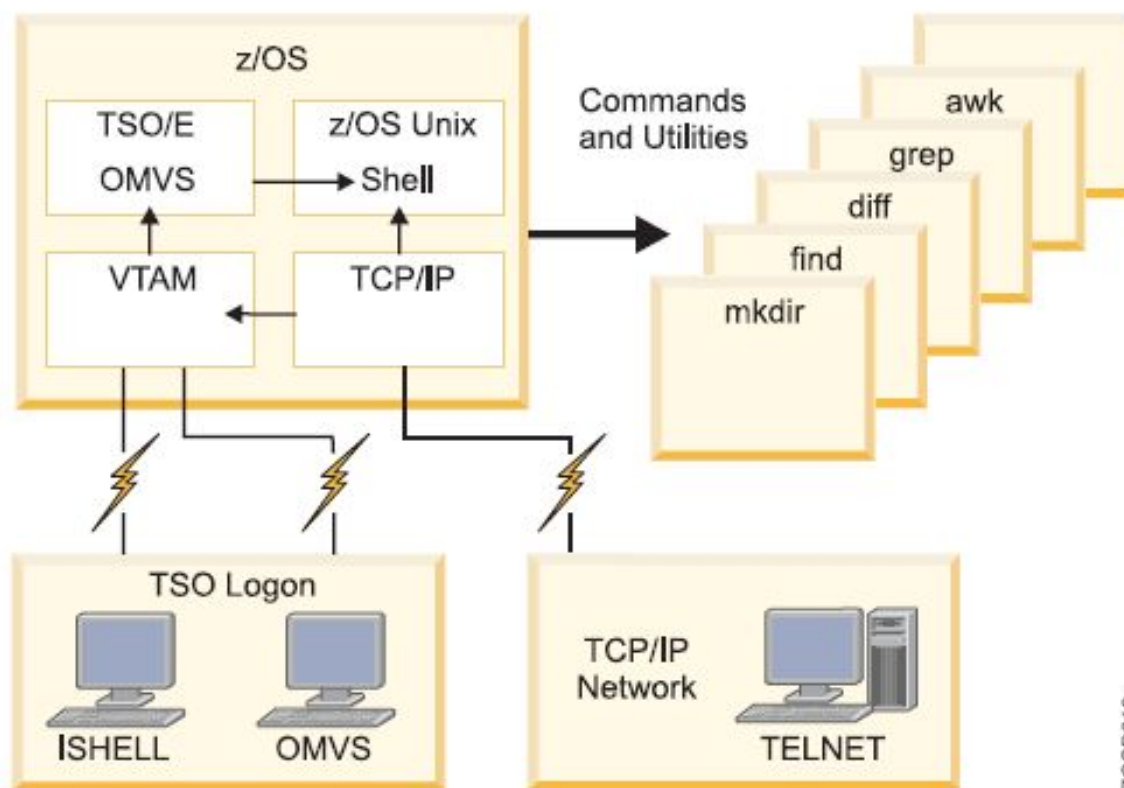
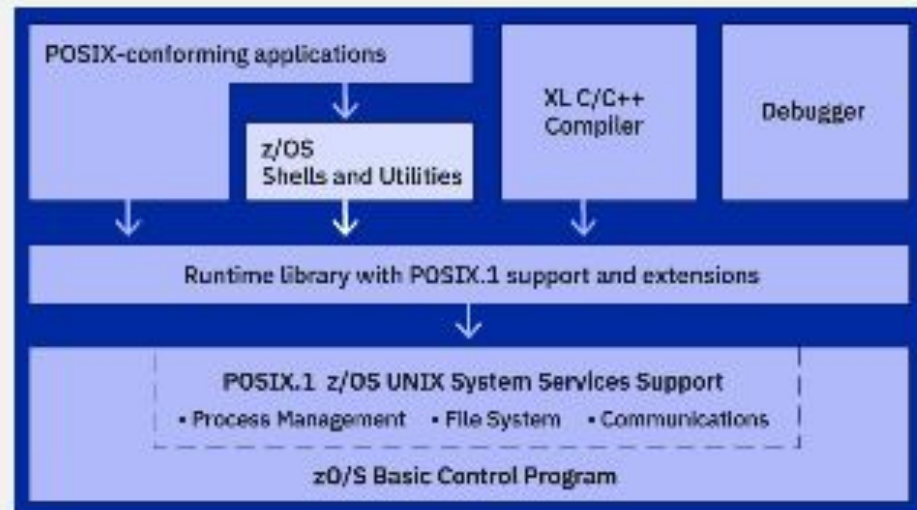


Figure 35. Shell and utilities

- A user can invoke the z/OS UNIX shell in the following ways:
- From a 3270 display or a workstation running a 3270 emulator
- From a TCP/IP-attached terminal, using the **rlogin and telnet commands**
- From a TSO session, using the OMVS command.

Unix System Services

- POSIX is the Unix standard
- z/OS has POSIX compliant Unix
- z/OS is POSIX compliant
- Complete POSIX capabilities
- “No Sides”
- Tightly integrated in z/OS BCP



- Full Unix File System Support

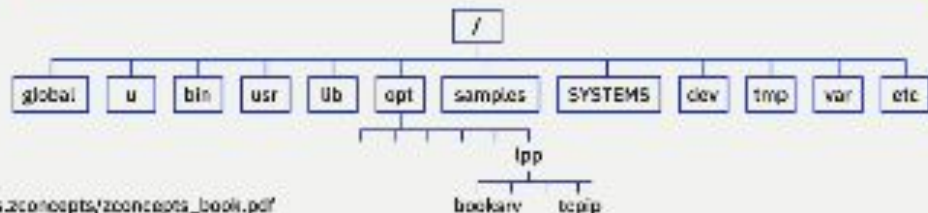
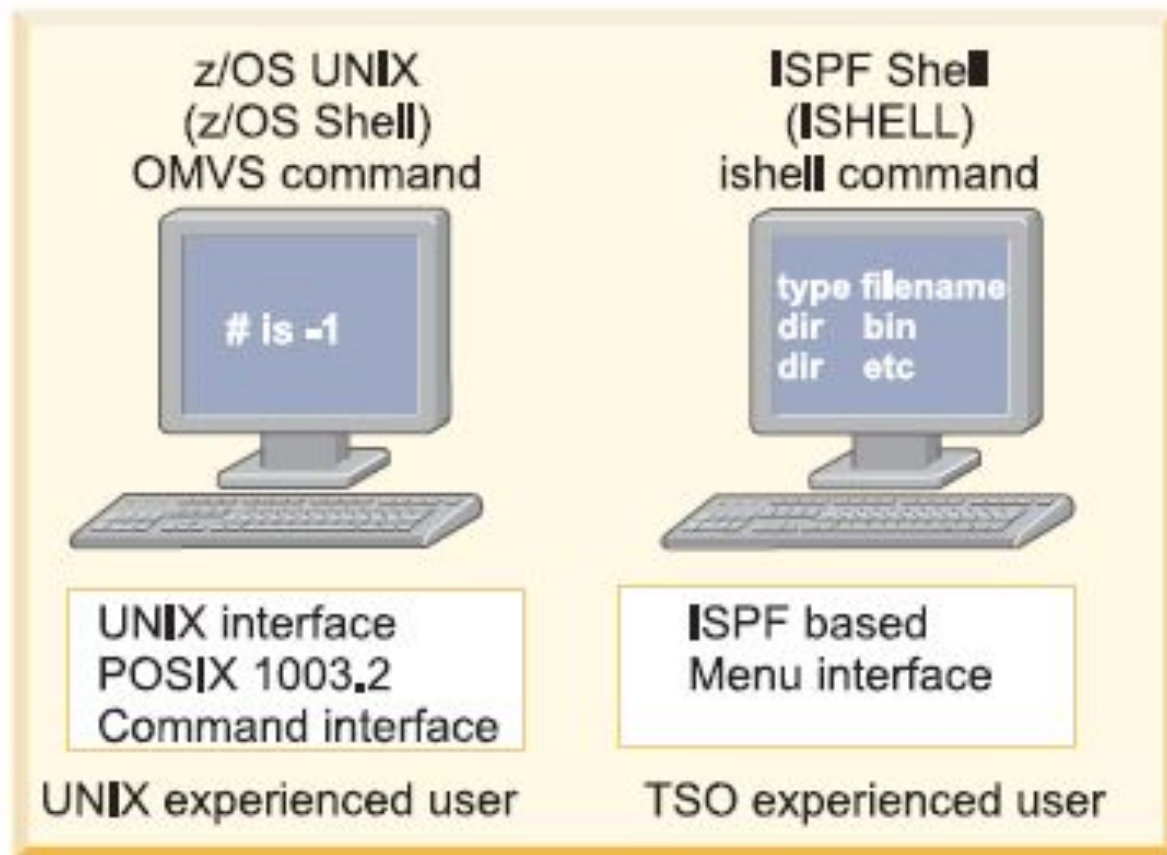


Diagram Source:
https://www.ibm.com/support/knowledgecenter/zosbasics/com.ibm.zos.zconcepts/zconcepts_book1.pdf



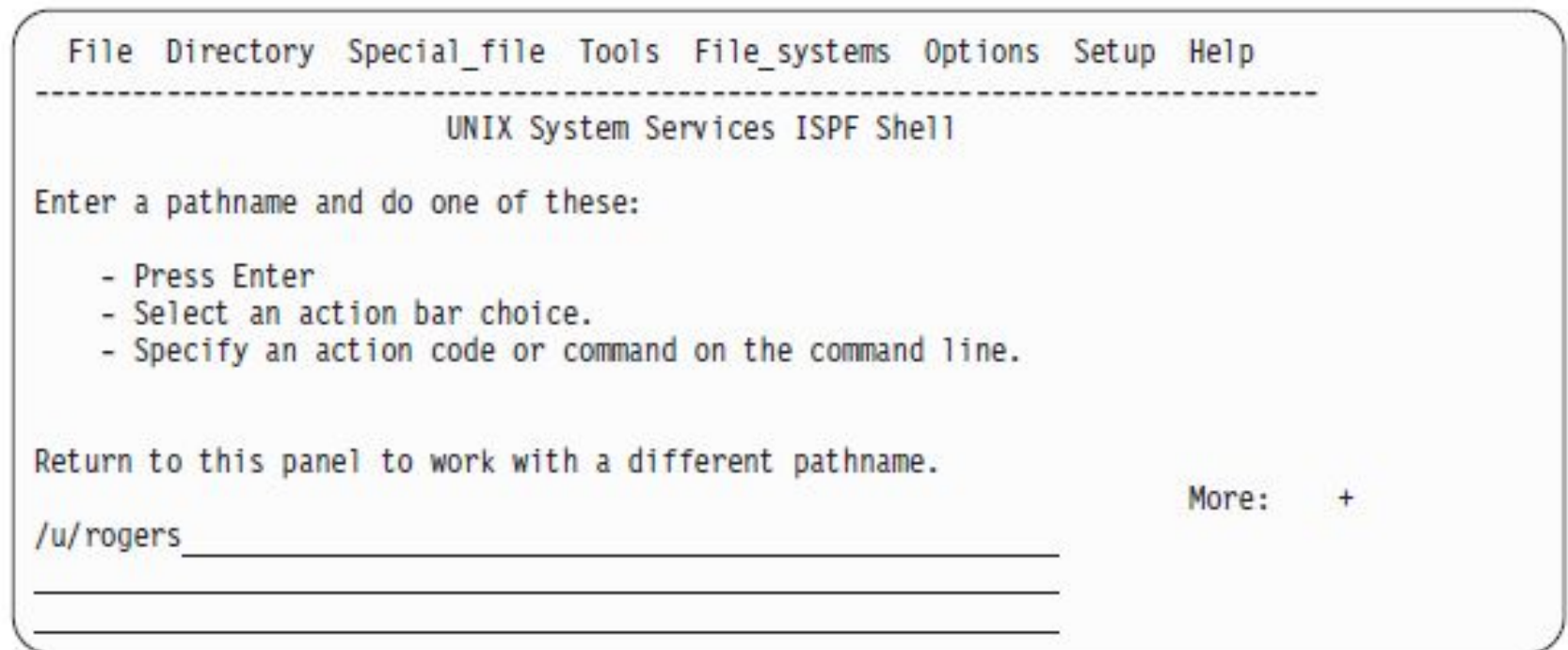
ZOSB020

Figure 36. z/OS UNIX interactive interfaces

- The z/OS UNIX shell is based on the UNIX System V shell and has some of the features from the UNIX Korn shell.
- You can store a sequence of shell commands in a text file that can be executed. This is called a **shell script**.

ISHELL command (ish)

- The ISHELL command invokes the ISPF panel interface to z/OS UNIX System Services.



The screenshot shows a terminal window with a menu bar at the top containing the following items: File, Directory, Special_file, Tools, File_systems, Options, Setup, and Help. Below the menu bar is a dashed line, followed by the title "UNIX System Services ISPF Shell". The main text area contains the instruction "Enter a pathname and do one of these:" followed by a bulleted list: "- Press Enter", "- Select an action bar choice.", and "- Specify an action code or command on the command line." Below this list is the text "Return to this panel to work with a different pathname." On the right side of the panel, there is a "More:" label followed by a "+" sign. At the bottom, there are three horizontal input lines; the first line contains the text "/u/rogers" followed by a cursor.

```
File  Directory  Special_file  Tools  File_systems  Options  Setup  Help
-----
                UNIX System Services ISPF Shell

Enter a pathname and do one of these:

- Press Enter
- Select an action bar choice.
- Specify an action code or command on the command line.

Return to this panel to work with a different pathname.

More:      +

/u/rogers _____
_____
_____
```

Figure 37. Panel displayed after issuing the ISH command

Directory List

Select one or more files with / or action codes. If / is used also select an action from the action bar otherwise your default action will be used. Select with S to use your default action. Cursor select can also be used for quick navigation. See help for details.

EUID=0 /u/rogers

Type	Perm	Changed-EST5EDT	Owner	-Size	Filename	Row 1 of 9
Dir	700	2002-08-01 10:51	ADMIN	8129	.	
Dir	555	2005-02-13 11:14	AAAAAAA	0	..	
File	755	1996-02-29 18:02	ADMIN	979	.profile	
File	600	1996-03-01 10:29	ADMIN	29	.sh_history	
Dir	755	2001-06-25 17:43	AAAAAAA	8129	data	
File	644	2004-06-26 11:27	AAAAAAA	47848	inventory.export	
File	700	2002-08-01 10:51	AAAAAAA	16	myfile	
File	644	2007-06-22 17:53	AAAAAAA	43387	print.export	
File	644	2007-04-28 18:03	AAAAAAA	84543	Sc.pdf	

Figure 38. Display of a user's files and directories

OMVS command shell session

- The OMVS command is used to invoke the z/OS UNIX shell.
- You use the OMVS command to invoke the z/OS UNIX shell.
- The shell is a command processor that you use to:
 - Invoke shell commands or utilities that request services from the system.
 - Write shell scripts using the shell programming language.
 - Run shell scripts and C-language programs interactively (in the foreground), in the background, or in batch.

```
ROGERS @ SC43:/>ls -al /u/rogers
total 408
drwx----- 3 ADMIN  SYS1      8192 Aug  1 2005 .
dr-xr-xr-x 93 AAAAAAA TTY      0 Feb 13 11:14 ..
-rwxr-xr-x 1 ADMIN  SYS1      979 Feb 29 1996 .profile
-rw----- 1 ADMIN  SYS1      29 Mar  1 1996 .sh_history
-rw-r--r-- 1 AAAAAAA SYS1    84543 Apr 28 2007 Sc.pdf
drwxr-xr-x 2 AAAAAAA SYS1     8192 Jun 25 2001 data
-rw-r--r-- 1 AAAAAAA SYS1   47848 Jun 26 2004 inventory.export
-rwx----- 1 AAAAAAA SYS1     16 Aug  1 2005 myfile
-rw-r--r-- 1 AAAAAAA SYS1   43387 Jun 22 2007 print.export
```

Figure 39. OMVS shell session display after issuing the OMVS command

Direct login to the z/OS UNIX shell

- **rlogin:** You can rlogin (remote log in) to the shell from a system that has rlogin client support. To log in, use the rlogin command syntax supported at your site.
- **telnet :** You can telnet into the shell. To log in, use the telnet command from your workstation or from another system with telnet client support

JCL - Job Control Language,

JCL – Syntax Basics

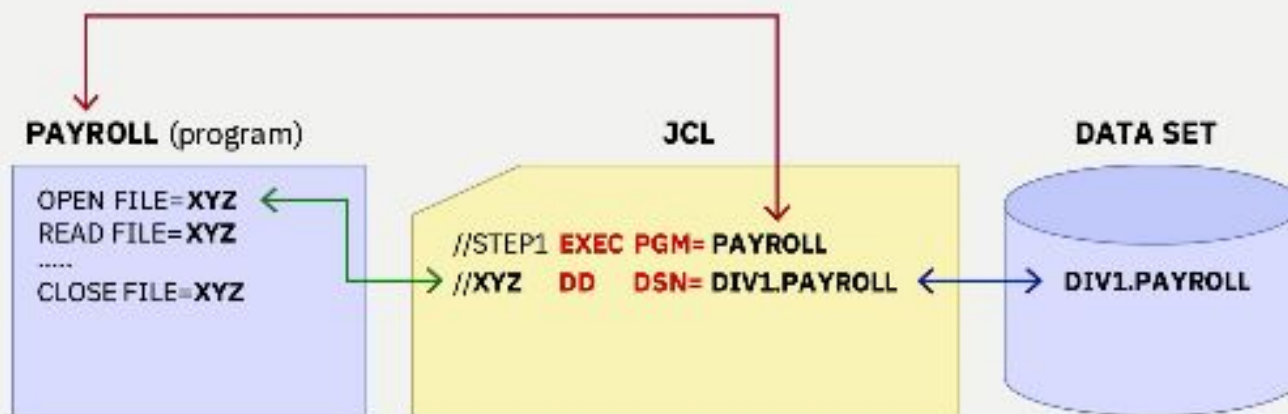
```
//JOBNAME JOB  
//STEPNAME EXEC  
//DDNAME DD  
//* ... this is a comment statement  
/* ... this indicates end of data  
// ... this indicates end for JCL
```

Fundamental JCL statements

Three basic JCL statements:

- 1) JOB who wants to process work
- 2) EXEC what program or procedure will be used
- 3) DD what are the program inputs and outputs

Program File Name and the JCL DD Name Relationship

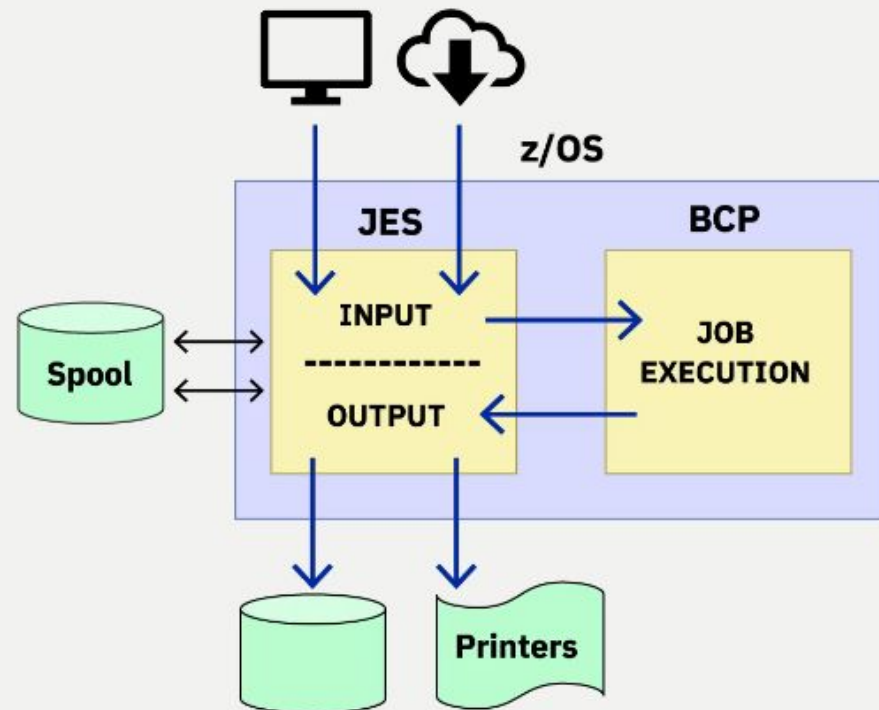


JES - Job Entry Subsystem,

What is a JES? Why is JES required by z/OS?

1. Receives work for processing
Input Queue
2. Schedules the work
Execution Queue
3. Controls output processing
Output Queue

The **Spool** stores and keeps track of inputs and outputs for z/OS



What is a catalog

- A catalog describes data set attributes and indicates the volumes on which a data set is located
- When a data set is cataloged, it can be referred to by name without the user needing to specify where the data set is stored.
- Data sets can
 - be cataloged, uncataloged, or recataloged.
- All system-managed DASD data sets are cataloged automatically in a catalog.
- In z/OS, the master catalog and user catalogs store the locations of data sets.
- Both disk and tape data sets can be cataloged.

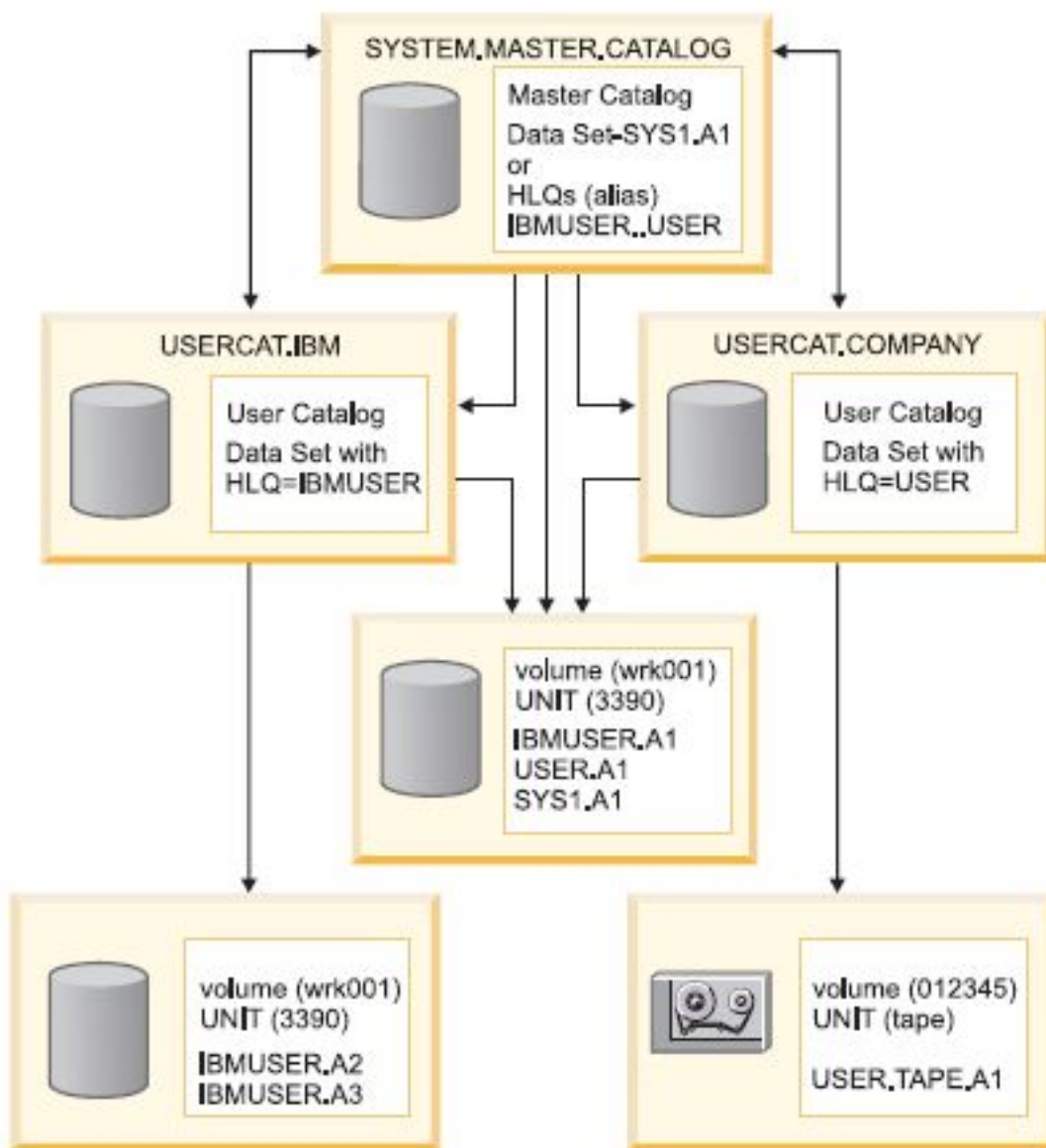
- To find a data set that you have requested, z/OS must know three pieces of information:
 - Data set name
 - Volume name
 - Unit (the volume device type, such as a 3390 disk or 3590 tape)
- You can specify all three values on ISPF panels or in JCL.
- A system catalog is used to store and retrieve the UNIT and VOLUME location of a data set.

Master catalogs and user catalogs

- A z/OS system always has at least one master catalog.
- If it had a single catalog, this catalog would be the master catalog and the location entries for all data sets would be stored in it.
- A single catalog, however, would not be efficient or flexible, so a typical z/OS system uses a master catalog and numerous *user catalogs connected to it*,
- The master catalog usually stores only the name of the user catalogs.

- A user catalog is a data set used to locate the DASD volume in which the requested data set is stored

- the data set name of the master catalog is `SYSTEM.MASTER.CATALOG`.
- This master catalog stores the full data set name and location of all data sets with a `SYS1` prefix, such as `SYS1.A1`.
- Two HLQ (alias) entries were defined to the master catalog, `IBMUSER` and `USER`.



Z069024

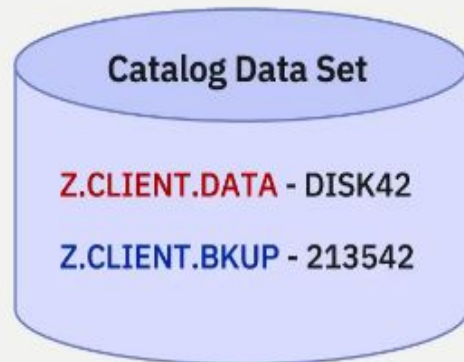
Figure 14. Catalog concept

System Catalog,

Locating a Data Set Name

```
//CLIENT DD DSN=Z.CLIENT.DATA,DISP=SHR
```

```
//BKUP DD DSN=Z.CLIENT.BKUP,DISP=SHR
```



Each Catalog Data Set Entry includes a Volume Label

z/OS Data Set Name Locate is Directed to the Volume Label

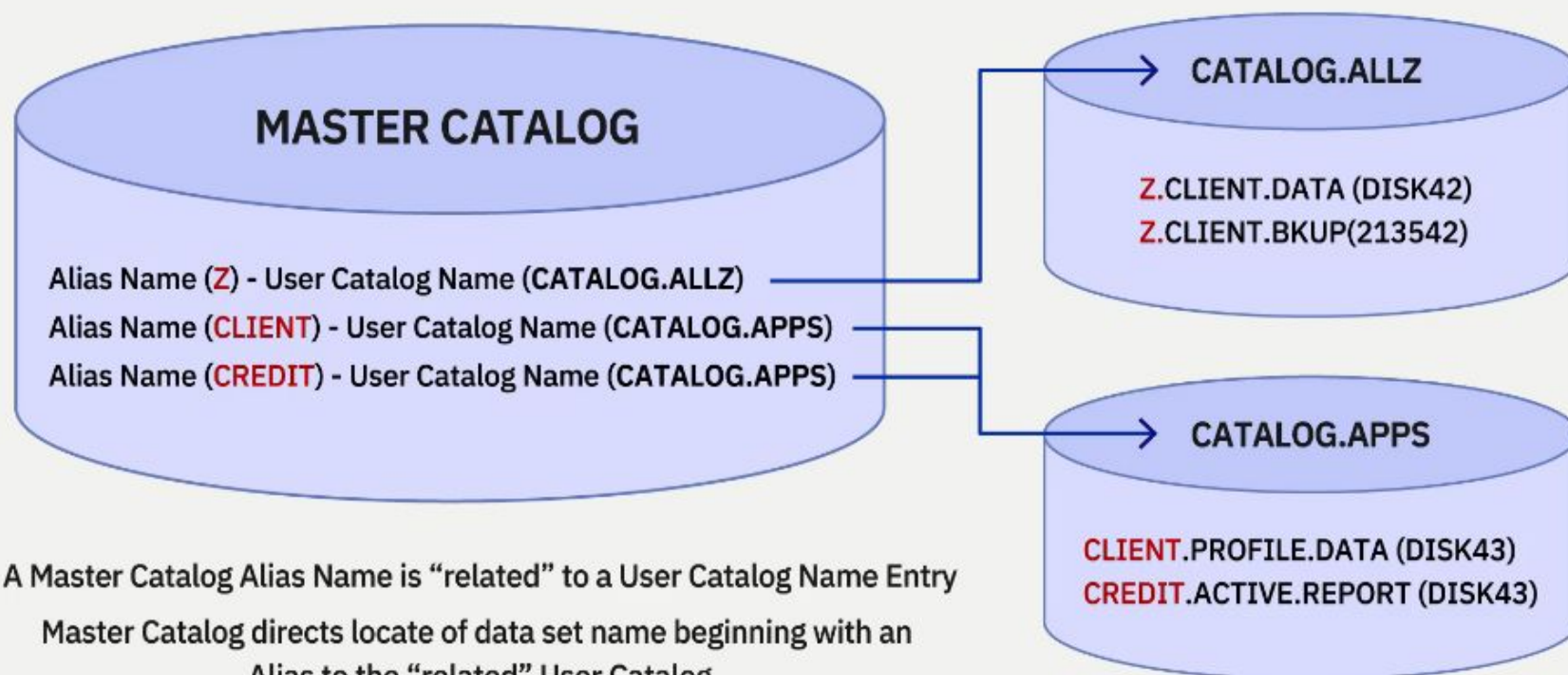
VTOC



Tape Label



Catalog Structure – Master Catalog Relationship to User Catalogs



A Master Catalog Alias Name is “related” to a User Catalog Name Entry
Master Catalog directs locate of data set name beginning with an
Alias to the “related” User Catalog

IBM Z

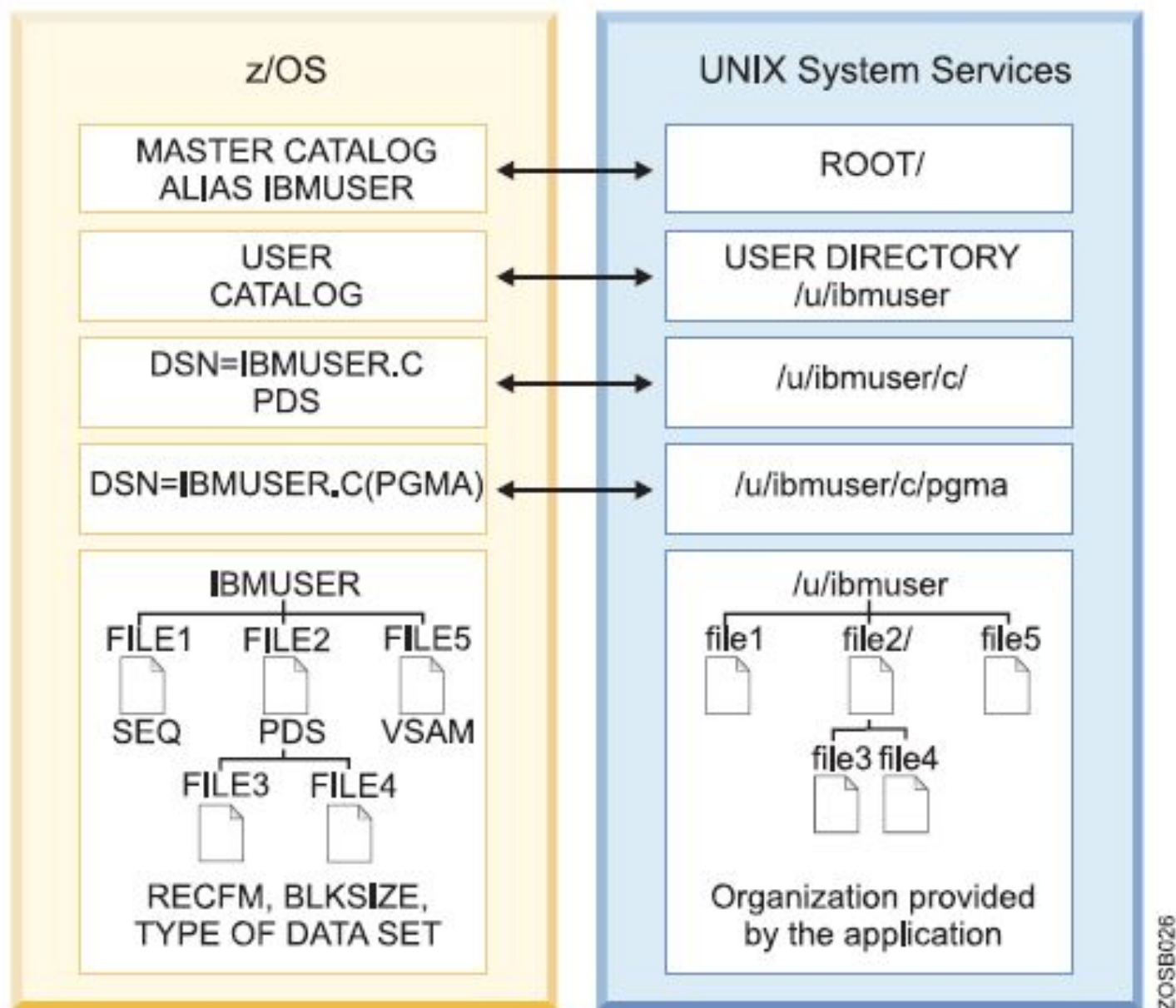


Figure 16. Comparison of z/OS data sets and file system files

Components, Messages, System Log,

z/OS is a collection of components

A **component** is a **collection of modules**

- **Base** components are always included in the operating system
 - Base components are also known as Base ‘elements’ which deliver essential operating system functions
- **Optional** components are installed in addition to the Base components
 - Optional components are also known as Optional ‘**features**’ which are requested separately from the Base ‘**elements**’

z/OS is a collection of components – Base & Optional

A **unique prefix** is assigned to each component

- Typical **component prefix** is **3 characters**
- **IKJ** (TSO, Time Sharing Option)

Component module names begin with the unique prefix

- **IKJ**EFT01 (TSO terminal monitor program)

MVS Diagnosis: Reference → z/OS Internet Library (z/OS MVS Bookshelf)

Table 1. Relating a module prefix to component and product

Module prefix	Component name
IKJ	TSO/E
IKJ	TSO terminal input/output controller (TIOC)

z/OS is a collection of components

Messages written by a component module begins with the same **unique prefix** IKJ56646I (IKJEFT01 message)

The same **message format** is used by both the base components and optional components with very few exceptions

The message format (divided into three parts) helps isolate and solve problems

1. reply identifier (optional)
2. message identifier
3. message text

z/OS Format of the Message Body

Message body consists of three parts:

1 – Optional reply identifier **2** – Message identifier **3** – Message text

1	2	3
id CCCnnn		text
id CCCnnns		text
id CCCnnnns		text
id CCCnnnnns		text
id CCCSnnns		text

CCC – component

S – subcomponent

nnnn – unique message number

s – **A** (action), **E** (eventual), **I** (information), **W** (warning), **S** (severe)

z/OS Example Format of the Message Body

CCCnnns text

IJK144I UNDEFINED USERID(S)

CCC – component (IKJ)

S – subcomponent (none)

nnnn – unique message number (144)

s – type code (I-Information)

z/OS Messages

The ability to read and interpret messages is an important skill within any operating system environment

z/OS messages follow a format which enables an experienced technician to quickly identify who wrote the message and why the message was written

Messages provide the ability to assess the status of the operating system, optional software products and applications

z/OS Systems Messages Manual

Explanation

The meaning of the message, including why the system issued the message

System Action

What the system did as a result of the system condition reported by the message

What the system did as a result of user input

Response

Instructions for the:

- | | | |
|---------------------|-------------------------|--------------------------|
| - Operator | - User | |
| - System Programmer | - Storage Administrator | - Security Administrator |

Problem Determination

Additional instructions for determining the cause of the problem

Source

Element, product, or component that issued the message

What is the SYSLOG?

- The system log (SYSLOG) is a chronological listing of messages about z/OS system activity.
- The z/OS operating system and other major middleware software products (ie CICS, DB2, WebSphere, etc.) write informational, warning, error and action messages to SYSLOG.
- The output of system commands are written to SYSLOG
- When an unexpected system problem occurs, the SYSLOG is the first place to look to gather information about the problem.
- System automation is a function of reading messages from SYSLOG and automating actions based on message content. System programmers can filter unnecessary messages from the SYSLOG to reduce message volume.

SYSLOG Format

message identifier and message text

z/OS MVS System Messages Volume - Introduction

Each SYSLOG entry has the following format:

t **crrrrrrr** **sysname** **yyddd** **hh:mm:ss.th** **ident** **msgflags** < message >

↓ ↓ ↓ ↓ ↓ ↓ ↓

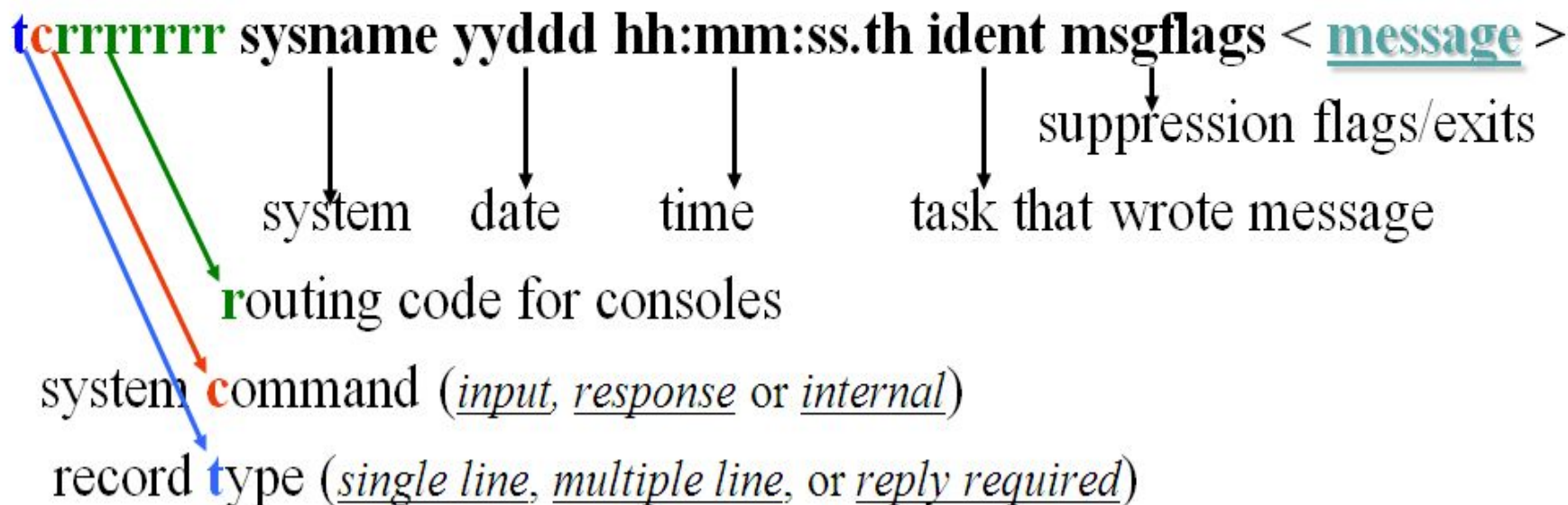
system date time task that wrote message

↓

routing code for consoles

system **c** command (*input*, *response* or *internal*)

record **t** type (*single line*, *multiple line*, or *reply required*)



The diagram illustrates the SYSLOG format with a sample entry: **t crrrrrrr sysname yyddd hh:mm:ss.th ident msgflags < message >**. Annotations include: an arrow from 'message identifier and message text' pointing to the '< message >' part; arrows from 'system', 'date', and 'time' pointing to 'sysname', 'yyddd', and 'hh:mm:ss.th' respectively; an arrow from 'task that wrote message' pointing to 'ident'; an arrow from 'routing code for consoles' pointing to 'crrrrrrr'; an arrow from 'system command' pointing to 'c'; and an arrow from 'record type' pointing to 't'.

SYSLOG Format - tcrrrrrrr

- **t** - The first character indicates the record type:
 - D - Data line of a multiple-line message; this line may be the last line of the message.
 - E - End line or data-end line of a multiple-line message.
 - L - Label line of a multiple-line message.
 - M - First line of a multiple-line message.
 - N - Single-line message that does not require a reply.
 - O - Operator LOG command.
 - S - Continuation of a single-line message or a continuation of the first line of a multi-line message. This continuation may be required because of the record length for the output device.
 - W - A message that requires a reply.
 - X - A log entry that did not originate with a LOG command or a system message.
- **C** - The second character indicates whether the line was generated because of a command:
 - C - Command input.
 - R - Command response.
 - I - Command issued internally. The job identifier contains the name of the internal issuer.
 - blank - Neither command input nor command response.
- **rrrrrrrr** - routing codes s 1 through 28.

Message Identifier and Text

- reply identifier number

Only present if message requires a response.
It appears if an operator reply is required.
The operator specifies # to reply.

Assigned **prefix** to identify the component, subsystem, or product that produced the message.

- # CCCnnn text.....
- # CCCnnns text.....
- # CCCnnnns text.....
- # CCCnnnnns text.....
- # CCCxnnns text.....

Subcomponent identifier

A **unique number** to identify the individual message.
The unique number is three, four, or five decimal digits.

Message ID suffix

A-Action

D-Decision

E-Error/Eventual action

I-Information

S-Severe error

W-Wait or Attention

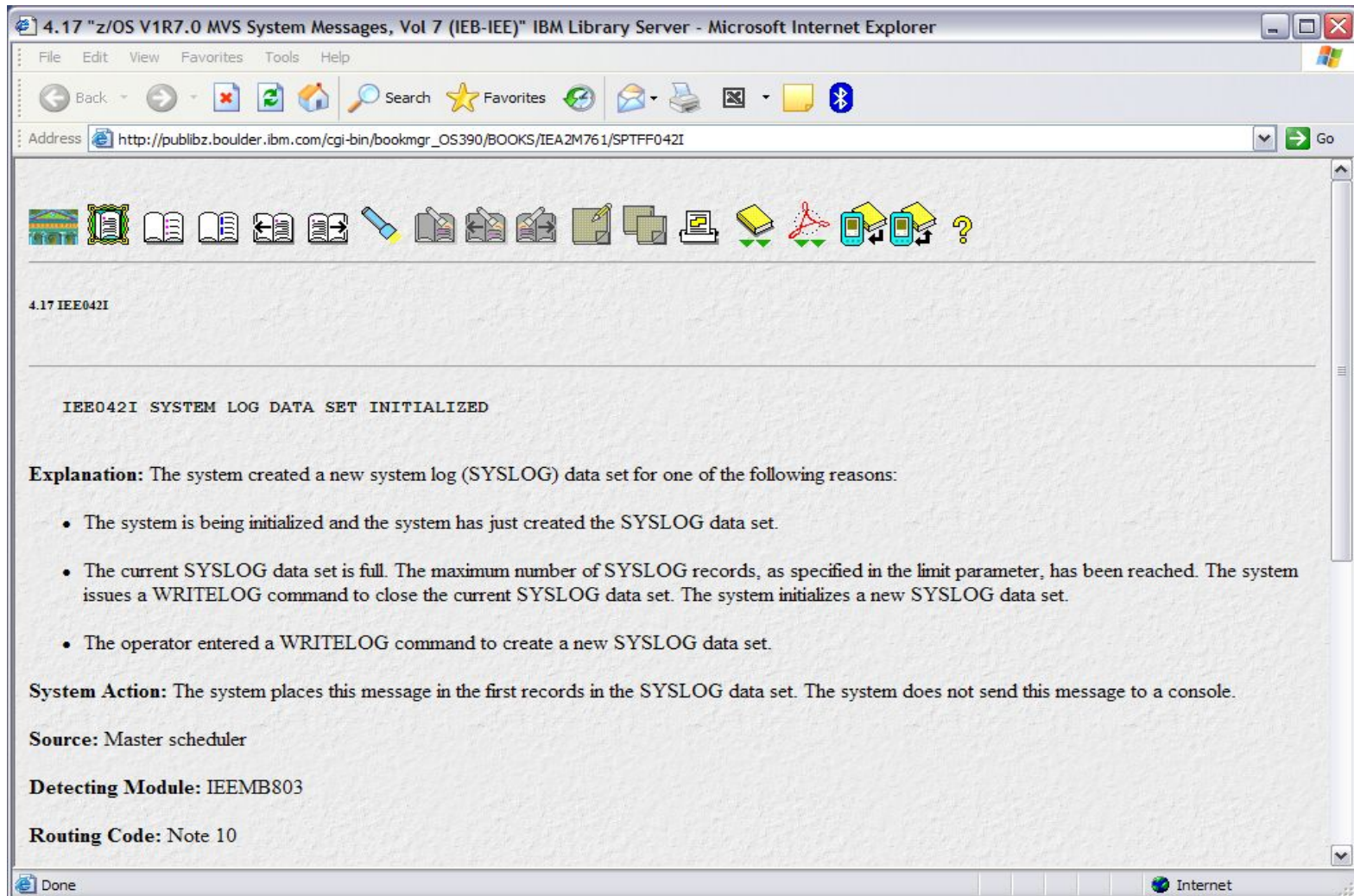
System Log (SYSLOG) Format

```
NC0000000 SOW1 17358 09:41:40.18 IBMUSER 00000290 D IPLINFO
MR0000000 SOW1 17358 09:41:40.18 IBMUSER 00000090 IEE254I 09.41.40 IPLINFO DISPLAY 690
DR          690 00000090 SYSTEM IPLED AT 19.57.49 ON 12/17/2017
DR          690 00000090 RELEASE z/OS 02.02.00 LICENSE = z/OS
DR          690 00000090 USED LOADW1 IN SYS1.IPLPARM ON 00CE3
DR          690 00000090 ARCHLVL = 2 MTLSHARE = N
DR          690 00000090 IEASYM LIST = (W1,SV,VN)
DR          690 00000090 IEASYS LIST = (00,LV,SV,VN) (OP)
DR          690 00000090 IODF DEVICE: ORIGINAL(00CE3) CURRENT(00CE3)
ER          690 00000090 IPL DEVICE: ORIGINAL(01000) CURRENT(01000) VOLUME(VIMVSB)

N 0020000 SOW1 17358 09:08:52.07 JOB05298 00000090 ICH70001I Z08019 LAST ACCESS AT 08:53:57 ON SUNDAY, DECEMBER 24, 201
↑
tccccrrrrr          ↑
CCCnnnnns
```

IBM Z

<http://www.ibm.com/servers/eserver/zseries/zos/bkserv/lookat/>



Key Manual to Understand z/OS SYSLOG & Messages

- MVS System Messages Volume 1
(Introduction)

Computer processing needs storage for:

OS Code	Operating System instructions / code
OS Data	Operating System data
App Code	Application Program instructions / code
App Data	Application Program data



In the example below:

0 thru 12 are the computer provided addresses for code and data

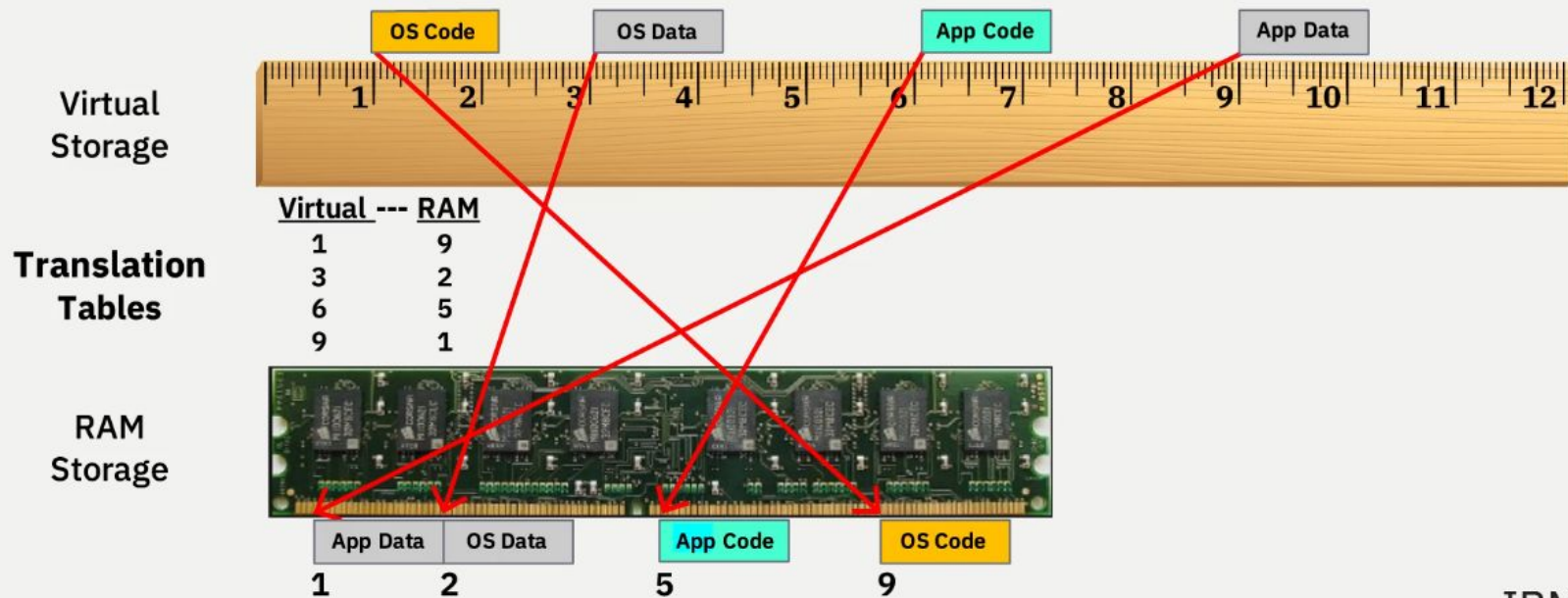
The address range for the code and data is called “virtual storage”

Virtual storage addresses with code and data need physical resources



Storage

Code and data is written to Random Access Memory, RAM using tables to translate virtual storage addresses to RAM addresses

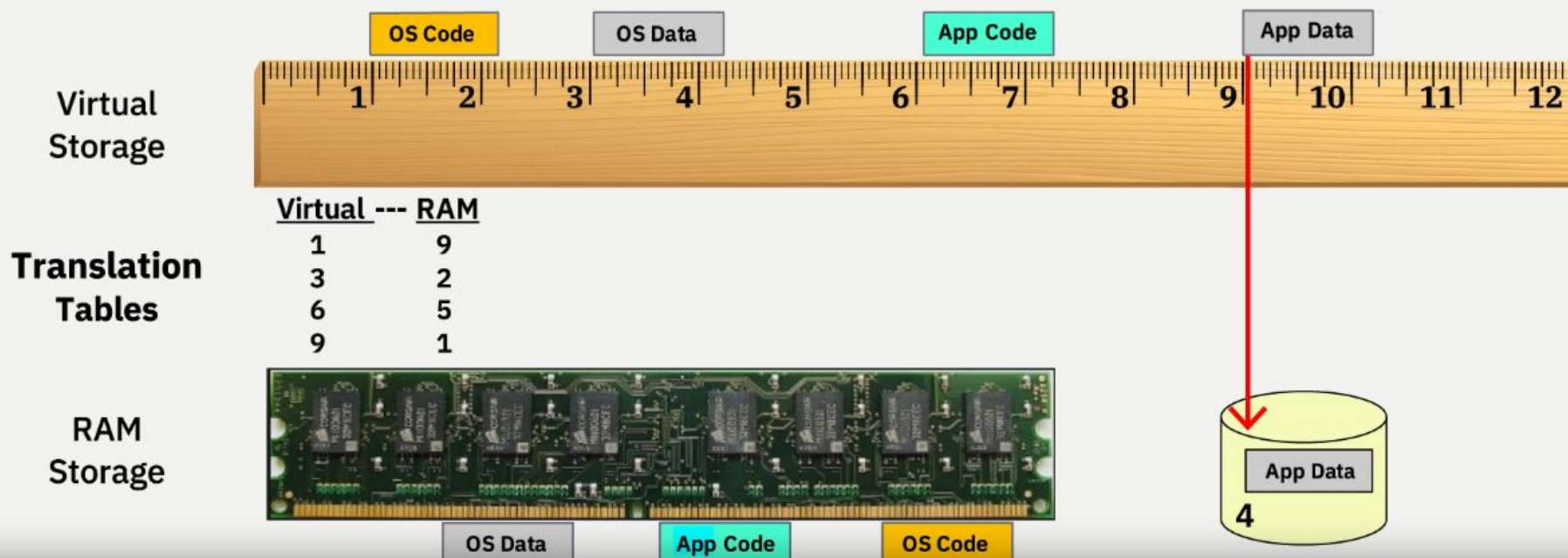


Auxiliary storage

Auxiliary storage is a secondary form of physical storage for code and data

Auxiliary storage addresses are available for code and data

Tables are used to translate virtual storage addresses to Auxiliary addresses



What is virtual storage

- Virtual storage means that each running program can assume it has access to all of the storage defined by the architecture's addressing scheme.
- The only limit is the number of bits in a storage address.
- This ability to use a large number of storage locations is important because a program may be long and complex, and both the program's code and the data it requires must be in central storage for the processor to access them.

What is virtual storage

- Virtual storage is an illusion created by the architecture, in that the system seems to have more memory than it really has.
- Each user or program gets an address space, and each address space contains the same range of storage addresses.
- Only those portions of the address space that are needed at any point in time are actually loaded into central storage.
- z/OS keeps the inactive pieces of address spaces in auxiliary storage.

What is an address space

- The range of virtual addresses that the operating system assigns to a user or separately running program is called an *address space*.
- This is the area of contiguous virtual addresses available for executing instructions and storing data. The range of virtual addresses in an address space starts at zero and can extend to the highest address permitted by the operating system architecture.
- z/OS provides each user with a unique address space and maintains the distinction between the programs and data belonging to each address space.
- Within each address space, the user can start multiple tasks by using *task control blocks* (TCBs) that allow multiprogramming.

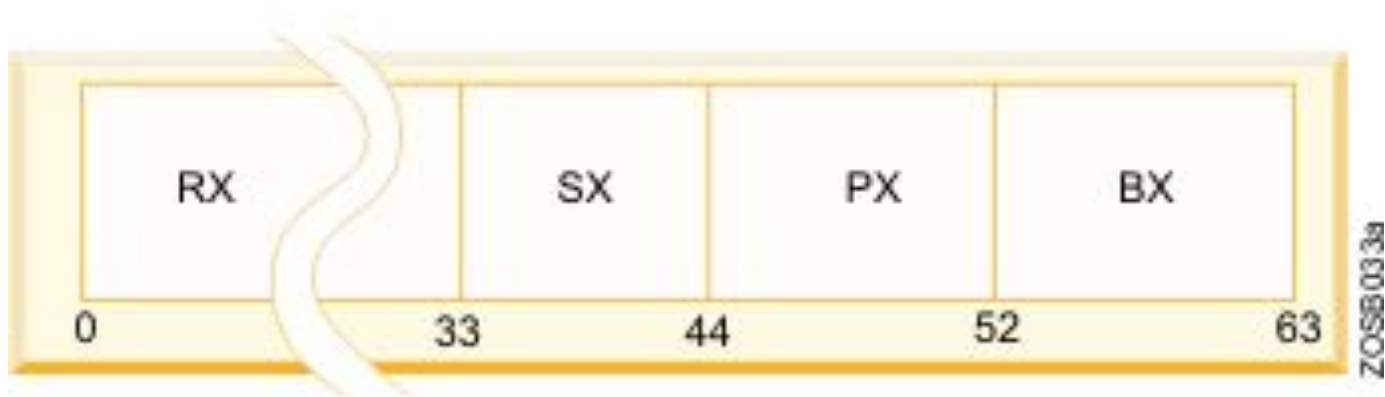
What is virtual storage

- In other ways, a z/OS address space is like a UNIX process, and the address space identifier (ASID)¹⁰ is like a process ID (PID).
- Further, TCBs are like UNIX threads in that each operating system supports processing multiple instances of work concurrently.

- z/OS manages address spaces in units of various sizes, as follows:
- Page address spaces are divided into 4-kilobyte units of virtual storage called pages.
- Segment address spaces are divided into 1-megabyte units called segments. A segment is a block of sequential virtual addresses spanning megabytes, beginning at a 1-megabyte boundary. A 2-gigabyte address space, for example, consists of 2048 segments.
- Region address spaces are divided into 2-8 gigabyte units called regions. A region is a block of sequential virtual addresses spanning 2-8 gigabytes, beginning at a 2-gigabyte boundary. A 2-terabyte address space, for example, consists of 2048 regions.
- A virtual address, accordingly, is divided into four principal fields: Bits 0-32 are called the region index (RX), bits 33-43 are called the segment index (SX), bits 44-51 are called the page index (PX), and bits 52-63 are called the byte index (BX).

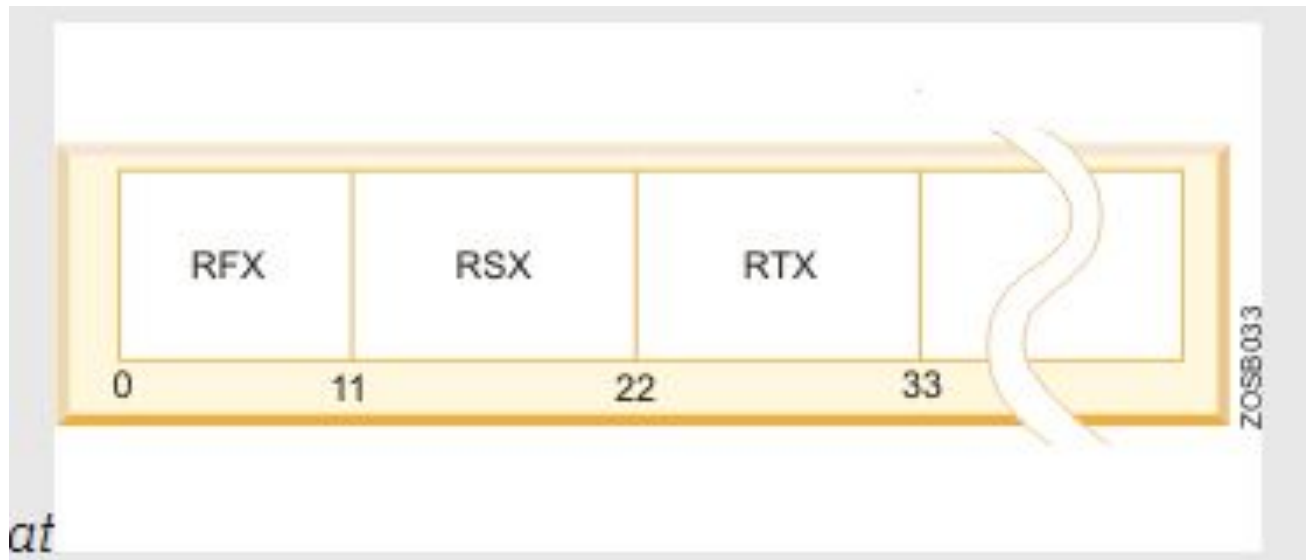
virtual storage addressing

- A virtual address has the format shown in [Figure 1](#)



virtual storage addressing

- The RX part of a virtual address is itself divided into three fields. Bits 0-10 are called the region first index (RFX), bits 11-21 are called the region second index (RSX), and bits 22-32 are called the region third index (RTX). Bits 0-32 of the virtual address have the format shown in Figure .



What is paging?

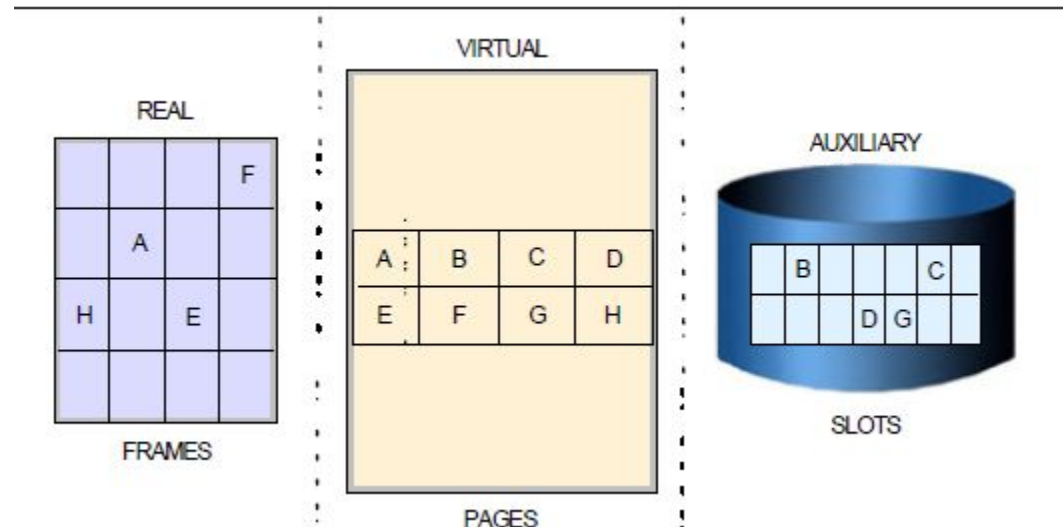
- When a program is selected for execution, the system brings it into virtual storage, divides it into pages of four kilobytes, transfers the pages into central storage for execution.
- To the programmer, the entire program appears to occupy contiguous space in storage at all times.
- Actually, not all pages of a program are necessarily in central storage, and the pages that are in central storage do not necessarily occupy contiguous space.
- The pieces of a program executing in virtual storage must be moved between real and auxiliary storage.
- To allow this, z/OS® manages storage in units, or **blocks**, of four kilobytes.

Frames, pages, and slots

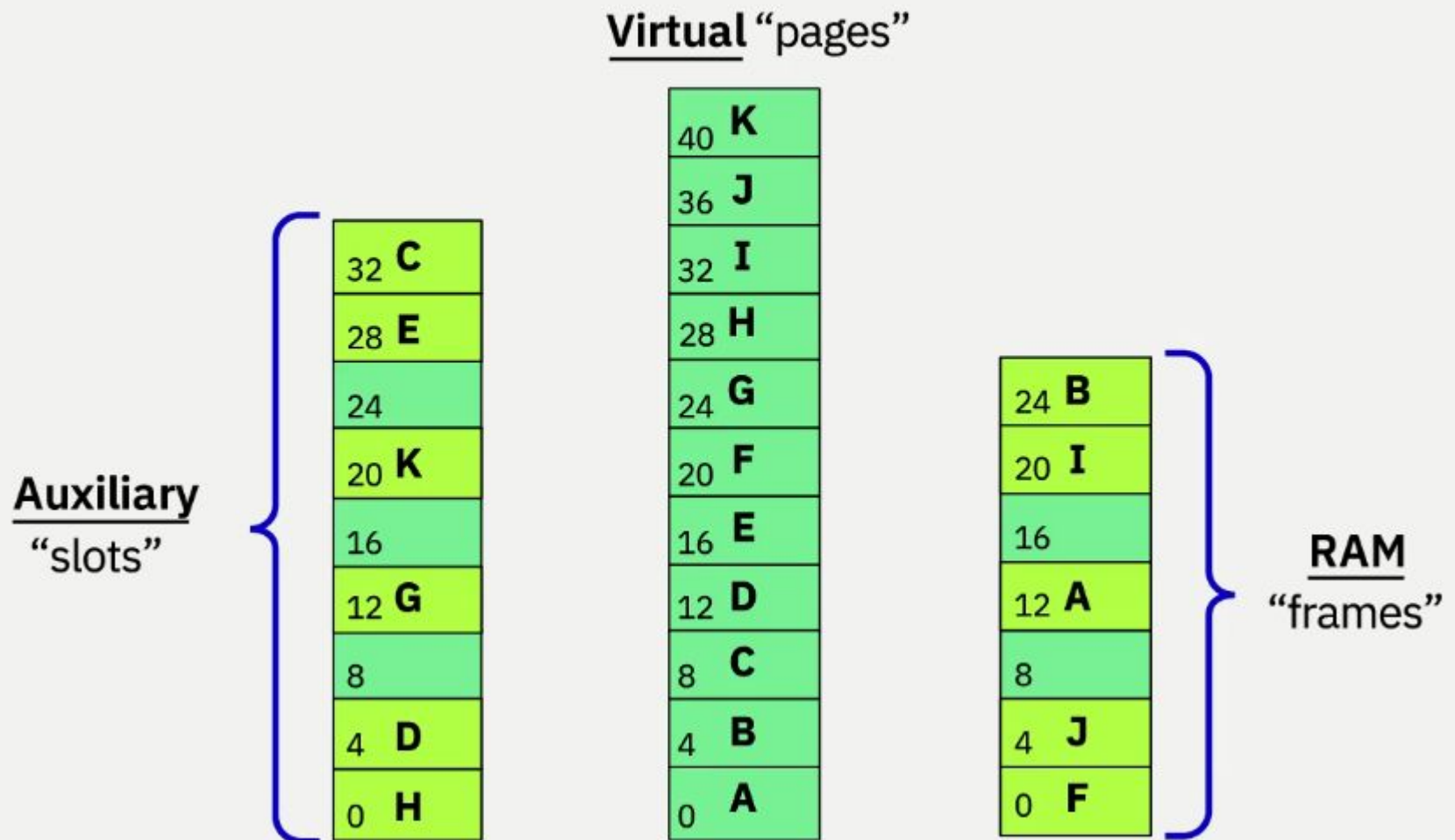
- A block of central storage is a **frame**.
- A block of virtual storage is a **page**.
- A block of auxiliary storage is a **slot**.
- A page, a frame, and a slot are all the same size: 4 KB.
- An active virtual storage page resides in a central storage frame.
- A virtual storage page that becomes inactive resides in an auxiliary storage slot (in a paging data set). Figure shows the relationship of pages, frames, and slots.

Frames, pages, and slots

- z/OS is performing paging for a program running in virtual storage.
- The lettered boxes represent parts of the program.
- In this simplified view, program parts A, E, F, and H are active and running in central storage frames, while parts B, C, D, and G are inactive and have been moved to auxiliary storage slots.
- All of the program parts, however, reside in virtual storage and have virtual storage addresses.



Frame and slot addresses



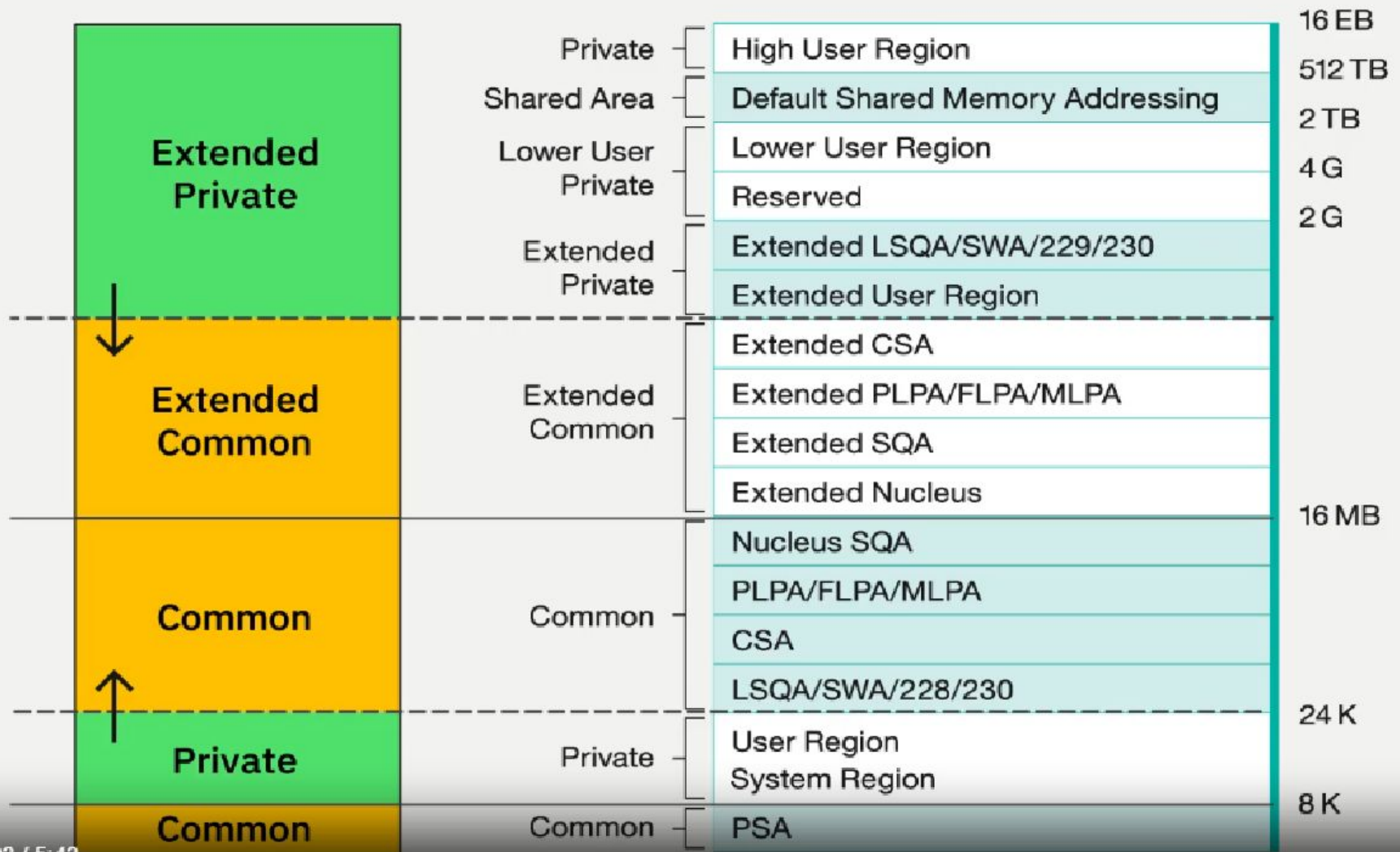
Address Space – An Overview

1. An address space is a **consecutive sequence of integer numbers** (virtual addresses), together with the specific transformation parameters which allow each number to be associated with a **byte location in storage**.
2. z/OS provides each user with a unique address space and maintains the distinction between the **programs** and **data** belonging to each address space.
3. While an address space includes **system code** as well as **user code** and **data**, all of the available addresses are mapped.
4. Some of the available addresses are mapped for system code which is “**common**” to user code and data.

Paging

- Z/OS uses a series of tables to determine whether a page is in real or auxiliary storage, and where.
- To find a page of a program, z/OS checks the table for the virtual address of the page, rather than searching through all of physical storage for it. z/OS then transfers the page into central storage or out to auxiliary storage as needed. This movement of pages between auxiliary storage slots and central storage frames is called paging.
- z/OS paging is transparent to the user. During job execution, only those pieces of the application that are required are brought in, or paged in, to central storage.
- The pages remain in central storage until no longer needed, or until another page is required by the same application or a higher-priority application and no empty central storage is available.
- To select pages for paging out to auxiliary storage, z/OS follows a "Least Used" algorithm

System address spaces and the master scheduler



- The program status word (PSW) is a 128-bit data area in the processor that,
- along with a variety of other types of registers (control registers, timing registers, and prefix registers) provides details crucial to both the hardware and the software.
- The current PSW includes the address of the next program instruction and control information about the program that is running.
- Each processor has only one current PSW. Thus, only one task can execute on a processor at a time.
- The PSW controls the order in which instructions are fed to the processor, and indicates the status of the system in relation to the currently running program.

- Under z/OS, the information in central storage is protected from unauthorized use by means of multiple storage protect keys. A control field in storage called a key is associated with each 4K frame of central storage.

Security on z/OS

- An installation's data and application programs must be protected from unauthorized access — both internally (employees) and externally (customers, business partners, or hackers). In working with z/OS®, you need to understand the importance of security and the z/OS facilities used to implement it.

Resource Access Control Facility

- Resource Access Control Facility (RACF®) is a security program. It is a component of the Security Server for z/OS®. RACF controls what you can do on the z/OS operating system. You can use RACF to protect your resources. RACF protects information and other resources by controlling the access to those resources.
- RACF provides security by:
- identifying and verifying users;
- authorizing users to access protected resources;
- recording and reporting access attempts.

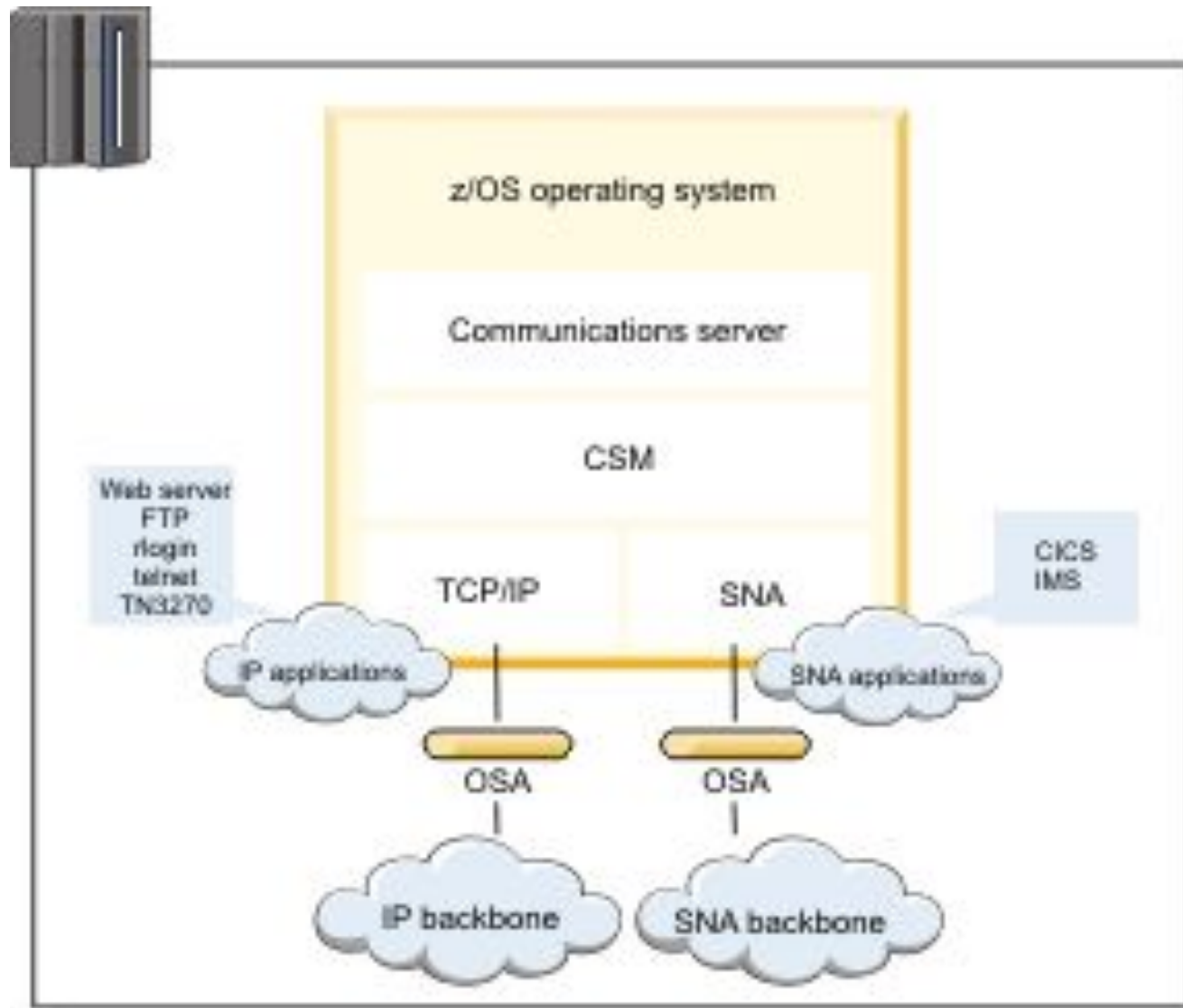
RACF

- User identification and authentication: RACF authenticates users with a password, password phrase, digital certificate, Kerberos ticket or a PassTicket.
- Authorizes access to protected resources: RACF retains information about your users, resources and access authorities in its database. This database determines access to protected mainframe system resources based on your security policy.
- Logging and reporting: RACF has logging and reporting functions that identify users who attempt to access the resource, either successfully or unsuccessfully. Detection of security exposures or threats is made possible by this feature.

z/OS Communications Server

- The z/OS operating system includes a software component called z/OS Communications Server.
- z/OS Communications Server implements the SNA and TCP/IP protocols.
- SNA applications and transaction servers (like CICS) can use SNA or TCP/IP to send and receive data. Industry standard internet applications can use TCP/IP to send and receive data. For example, a z/OS server may run FTP, telnet, web servers (HTTP), and mail programs (Simple Mail Transfer protocol, or SMTP).
- z/OS Communications Server provides a set of communications protocols that support connectivity functions for both local and wide-area networks, including the Internet. z/OS Communications Server also provides performance enhancements that can benefit a variety of well-known TCP/IP applications. These performance enhancements, which may be software-based or hardware-based, are discussed in their appropriate contexts.

z/OS Communications Server



z/OS Communications Server

- z/OS Communications Server includes three major components, which are:
- The TCP/IP protocol stack.
- The SNA protocol stack contained in Virtual Telecommunications Access Method (VTAM).
- The Communications Storage Manager (CSM), which provides a shared I/O buffer area for both TCP/IP and VTAM data flow. The CSM function allows authorized host applications to share data without having to physically move the data.

Unit_5_mainframe .PPTX

File Edit View Insert Format Slide Arrange Tools Help

Background Layout Theme Transition

- z/OS manages address spaces in units of various sizes, as follows:
- Page address spaces are divided into 4-kilobyte units of virtual storage called pages.
- Segment address spaces are divided into 1-megabyte units called segments. A segment is a block of sequential virtual addresses spanning megabytes, beginning at a 1-megabyte boundary. A 2-gigabyte address space, for example, consists of 2048 segments.
- Region address spaces are divided into 2-8 gigabyte units called regions. A region is a block of sequential virtual addresses spanning 2-8 gigabytes, beginning at a 2-gigabyte boundary. A 2-terabyte address space, for example, consists of 2048 regions.
- A virtual address, accordingly, is divided into four principal fields: Bits 0-32 are called the region index (RX), bits 33-43 are called the segment index (SX), bits 44-51 are called the page index (PX), and bits 52-63 are called the byte index (BX).

NAYANA THAKUR has left the meeting

Click to add speaker notes

aparna mete

AARYA BATKADLI

DEVANSH GOYAL
CS-AI-16

RUTVIK GAIKWAD
IT-A -52

- ADITYA SAKHARE
- 8:07 PM
- CS-D-26
- ABHAY SHANKHAPAL
- 8:07 PM
- IT-C 40
- SUJIT KANAWADE
- 8:07 PM
- CS-B-42
- PRANAV PATEL
- 8:07 PM
- IT-B-45
- AADITYA GAIKWAD
- 8:07 PM
- CS B 5
- SHLOK SONKUSARE
- 8:07 PM
- CSAI-A 62
- VRUSHAL PATIL
- 8:07 PM
- AI-C 60
- TANVI GUNJAL
- 8:07 PM
- CS B 16
- SHIVEN AGRAWAL
- 8:07 PM
- IT-A-03
- NAYANA THAKUR
- 8:07 PM
- CS_D_64
- BHAT ADITI
- 8:07 PM
- IT-A 02
- HARSHADA DESHINGKAR
- 8:07 PM
- AI-A-45
- BHAT SNEHA
- 8:07 PM
- CS-D-49
- ARYA LOKHANDE
- 8:07 PM
- CSAI-A-03
- AARYA BATKADLI
- 8:07 PM
- AI-A 20

Meet - cez-ztbd-vdn

Unit 5_mainframe.pptx - Google

RACF commands for general us

z/OS Communications Server -

meet.google.com/cez-ztbd-vdn?pli=1

Meet - cez-ztbd-vdn (ganesh.dongre@vit... EduPlusCampus Google LinkedIn jQuery 100% Inbox (594) - apam... Introduction to JSP |... PECL :: Package :: m... All Bookmarks

T

TEJASVINI WAGH

A

ANJALI DONGRE

D

DEVANSH JAISWAL

H

HARSHADA DESHINGKAR

A

ADITYA SAKHARE

RUPALI PAIMODE

A

ADITI KOHALE

VAISHNAVI GOSAVI

D

DURVESH CHAUDHARI

MANISH VARUN

A

AKASH GADEKAR

29

ARYAN CHALPE

D

RENUKA DESHPANDE

D

PRASAD KAMBLE

45 S

7 others

SHLOK SONKUSARE
CSAI - A- 62

To see more people, change your layout to show more tiles [Change layout](#)

More options

9:54 AM | cez-ztbd-vdn

Microphone

Video

Chat

Reactions

Screen share

Hand raise

More options

End call

22

Info

Participants

Messages

Settings

Lock

Type here to search

WhatsApp 88

Calendar

Chrome

File Explorer

Word

Spotify

PowerPoint

PDF Reader

ENG 09:55 AM 20-10-2024

