



---

# z/OS, TSO e ISPF

Roberto Gómez Cárdenas  
rogomez@itesm.mx

Lámina 1 Roberto Gómez C.

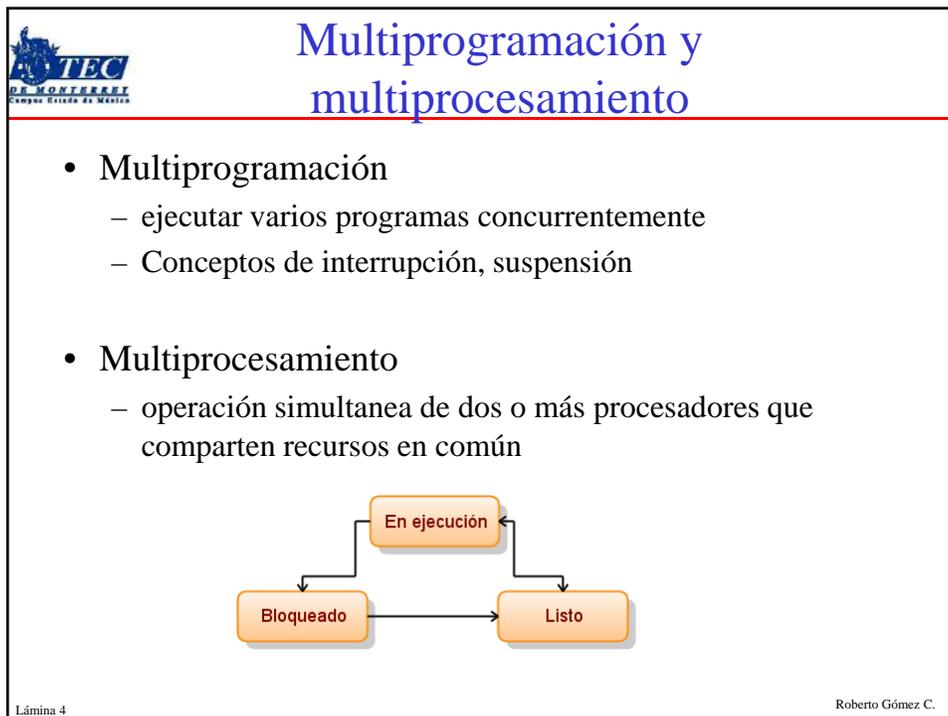


## ¿Qué es z/OS?

---

- Sistema operativo mainframe
  - sistema operativo de 64 bits
- Utilizado para procesar grandes cantidades de trabajo para varios usuarios concurrentes.
- Diseñado para
  - servir 1000s de usuarios concurrentemente
  - Computo I/O intensivo
  - Procesamiento de grandes cantidades de carga de trabajo
  - Ejecución segura de aplicaciones críticas

Lámina 2 Roberto Gómez C.





## Módulos, componentes sistemas y macros

- z/OS compuesto de instrucciones que controlan la operación del sistema.
  - Aseguran que el hardware esta trabajando de forma eficiente.
  - Por ejemplo: aceptar trabajos, convertir el trabajo a una forma que el sistema pueda reconocer.
- Grupo instrucciones relacionadas es una rutina o *módulo*.
- Conjunto módulos es un *componente del sistema*.
  - Por ejemplo:
    - WLM: Workload Management *componente* de z/OS que controla recursos del sistema
    - RTM: Recovery Termination Manager *componente* que maneja recuperación sistema
- Secuencia instrucciones que llevan a cabo de forma frecuente funciones del sistema son invocadas dentro de *macros*.

Lámina 5 Roberto Gómez C.



## Los control blocks

- Bloque de memoria con información dentro de ella.
- Se almacena información relacionada con la ejecución de un programa.
  - Algunas veces los programadores requieren ver bajo el cofre lo que esta sucediendo.
- Cuatro tipos
  - System-related control blocks
  - Resource-related control blocks
  - Job-related control blocks
  - Task-related control blocks
- Sirven como vehículos de comunicación a través de z/OS y contiene información del sistema.
- Sistema operativo puede buscar información acerca de una unidad de trabajo o recurso, que puede ser:
  - Datos actuales: un valor, una cantidad un parámetro o un nombre.

Lámina 6 Roberto Gómez C.



## Ejemplo control blocks usado por z/OS

- Sistema z/OS usa una gran variedad de bloques de control, muchos con propósitos muy específicos.
- Los tres más usados
  - TCB: Task Control Block
    - Representa una unidad de trabajo o tarea.
  - SRB: Service Request Block
    - Representa una petición para un servicio del sistema.
  - ASCB: Address Space Control Block
    - Representa un espacio de direcciones.

Lámina 7 Roberto Gómez C.



## La memoria virtual

**If it's there and you can see it**  
**- *it's real***

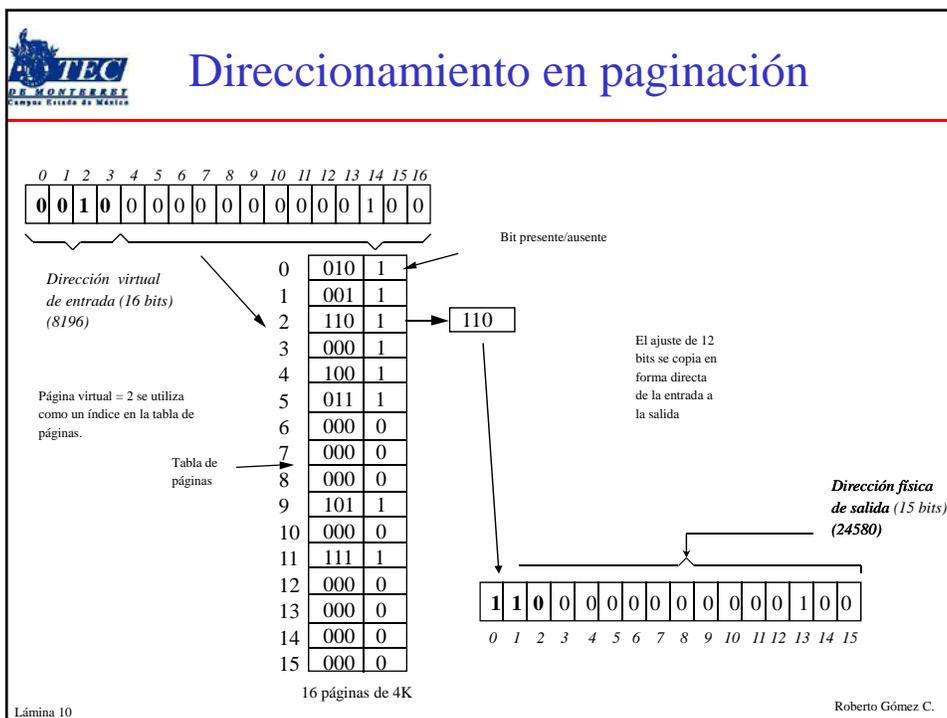
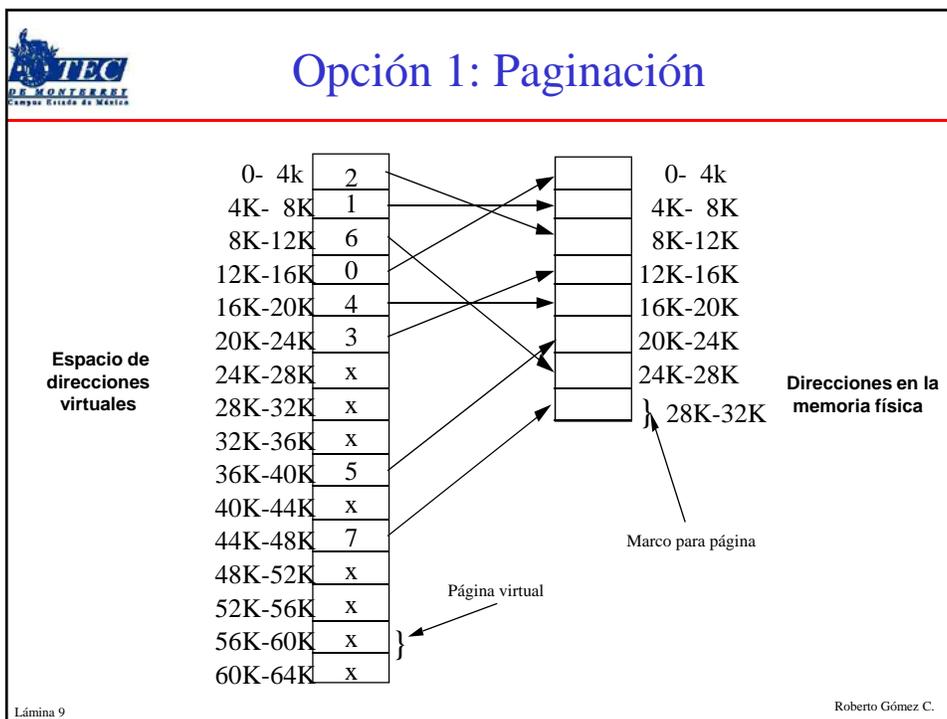
**If it's not there and you can see it**  
**- *it's virtual***

**If it's there and you can't see it**  
**- *it's transparent***

**If it's not there and you can't see it**  
**- *you erased it!***

*IBM poster explaining virtual memory, circa 1978*

Lámina 8 Roberto Gómez C.





## Tablas de páginas multinivel

Tabla superior:  
 c/entrada = 4M  
 => se tienen 4G de direcciones virtuales  
 $2^{32} = 4,294,967,296 = 4G$

Tabla de páginas de segundo nivel

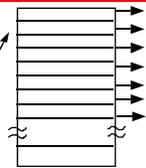
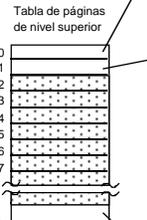


Tabla de páginas de nivel superior



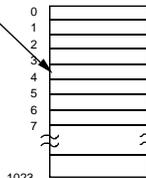
Bits

10	10	12
PT 1	PT 2	Offset

(a)

(a) Una dirección de 32 bits con dos campos para la tabla de páginas.

Tabla de páginas para los 4 M superiores de la memoria



(b) Tablas de páginas de dos niveles

Lámina 11

Roberto Gómez C.



## Ejemplo tablas multinivel

Espacio direcciones virtuales:  $2^{32} = 4,294,967,296 = 4G$   
 Ejemplo dirección virtual:  
 $0x00403004 = 0000\ 0000\ 0100\ 0000\ 0011\ 0000\ 0000\ 0100$   
 => **PT1 = 1** **PT2 = 3** **Offset = 4**

tabla 0

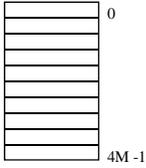
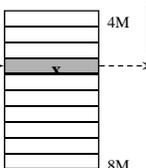
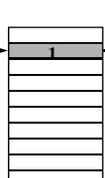


tabla 1



PT1 = 1



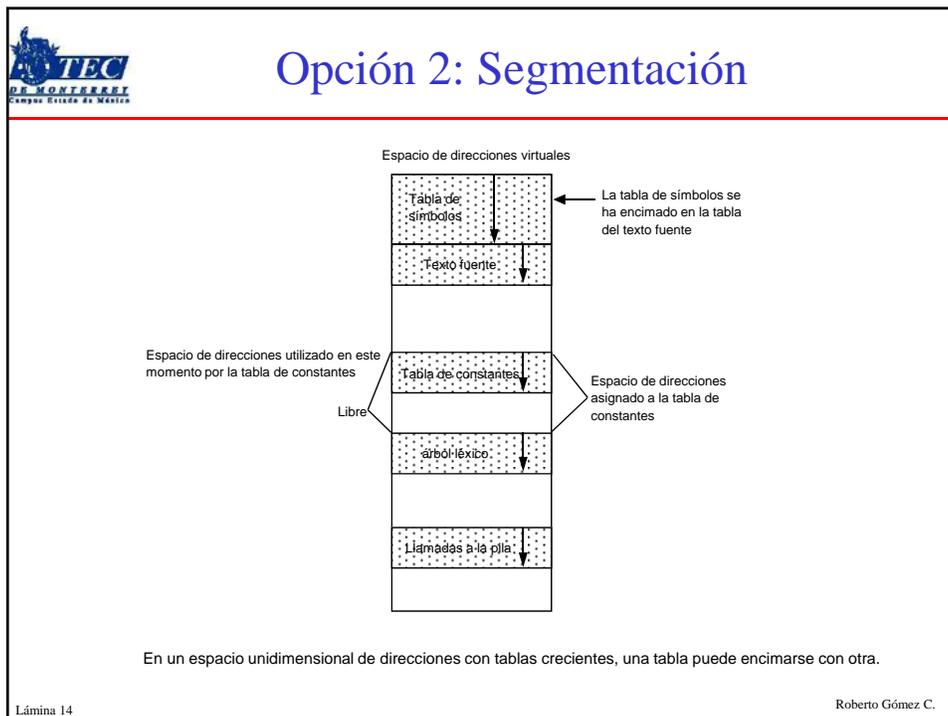
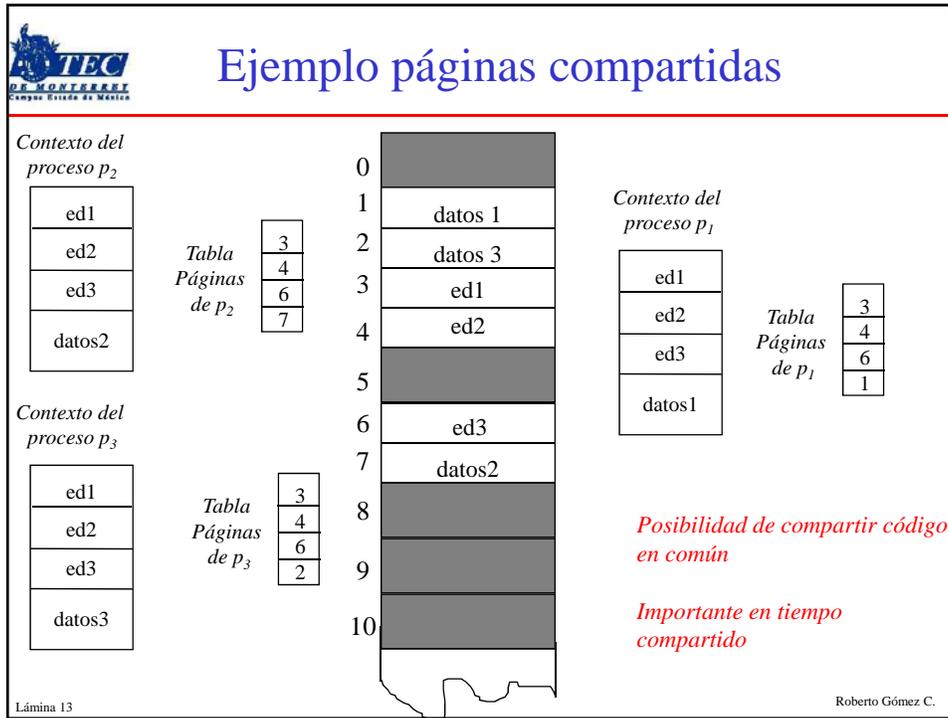
PT2 = 3

12292 (dir. abs 4,206,592)  
 a  
 16383 (dir. abs 4, 210,687)

x contiene el número de marco de la página con dirección 0x00403004

Lámina 12

Roberto Gómez C.

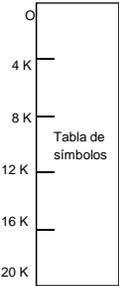




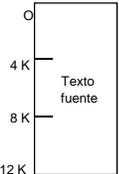
## Ejemplos de segmentos

---

Segmento 0



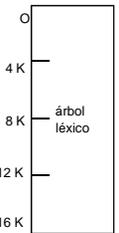
Segmento 1



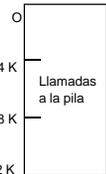
Segmento 2



Segmento 3



Segmento 4



*Una memoria segmentada permite que cada tabla crezca o se reduzca en forma independiente de las demás*

Lámina 15
Roberto Gómez C.



## La tabla de segmentos

---

**Tabla de Segmentos**

limite	base

**Memoria Principal**



CPU → (s, d)

↑ s

↓ d

base + d

d < limite

si → base + d

no → trap, error direccionamiento

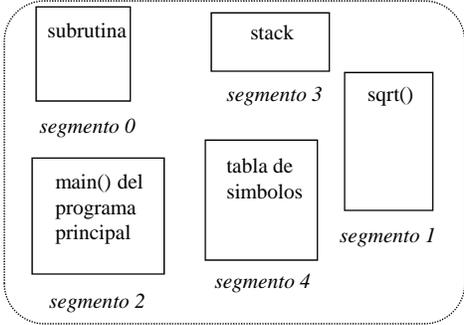
Lámina 16
Roberto Gómez C.



## Ejemplo segmentación

---

**Espacio direcciones virtuales**



**Tabla de Segmentos**

	limite	base
0	1000	1400
1	400	6300
2	400	4300
3	1100	3200
4	1000	4700

**Memoria Principal**



Lámina 17 Roberto Gómez C.



## Tipos de almacenamiento

---

- Conceptualmente se manejan dos tipos de almacenamiento:
  - Almacenamiento físico que reside dentro del procesador mismo.
    - Conocido como processor storage, real storage o central storage
    - El acceso es síncrono con el procesador, este debe esperar mientras los datos son extraídos.
  - Almacenamiento físico externo al mainframe.
    - Almacenamiento en discos o cintas.
    - Conocido como paging storage o auxiliary storage.
    - El acceso es asíncrono.
    - Acceso de hace a través de peticiones E/S

Lámina 18 Roberto Gómez C.



## Elementos internos z/OS

- Maneja tres diferentes tipos de almacenamiento:
  - almacenamiento real
  - almacenamiento auxiliar
  - almacenamiento virtual

Lámina 19 Roberto Gómez C.



## El espacio de direcciones

- Almacenamiento virtual
  - Ilusión creada con a través del manejo del almacenamiento real y auxiliar a través de tablas.
- Porciones ejecución de un programa son dejados en almacenamiento real
  - El resto es dejado en almacenamiento auxiliar.
- Espacio direcciones
  - Rango direccionable de almacenamiento virtual disponible para un usuario o programa.
  - El rango de direcciones empieza en cero y puede extenderse a la dirección más grande que permita la arquitectura del sistema operativo.
  - Cada usuario o programa ejecutando separadamente es representado por un espacio de direcciones.

Lámina 20 Roberto Gómez C.



## Espacio direcciones y programas

- Cada usuario cuenta con un espacio de direcciones único.
- z/OS mantiene una diferencia entre los programas y los datos que pertenecen a cada espacio de direcciones.
- Dentro de un espacio de direcciones un usuario puede arrancar varias tareas, usando los TCBs (Task Control Blocks).
  - Esto es lo que permite la multiprogramación.

Lámina 21 Roberto Gómez C.



## Espacio direcciones Z/OS y procesos Unix

- Un espacio de direcciones de z/OS es como un procesos UNIX.
- El identificador de espacio de direcciones (ASID) es como un identificador de procesos (PID).
- TCBs son como threads, ya que se puede contar con varias instancias de trabajo concurrentes.

Lámina 22 Roberto Gómez C.



## Espacios direcciones en z/OS

---

- z/OS usa muchos espacios de direcciones.
- Existe al menos un espacio de direcciones por cada trabajo en progreso y un espacio de direcciones por cada usuario conectado vía TSO, telnet, rlogin o FTO.
- Existen varios espacios de direcciones para funciones del sistema operativo como
  - Operadores de comunicación
  - Redes
  - Seguridad
  - Etc.

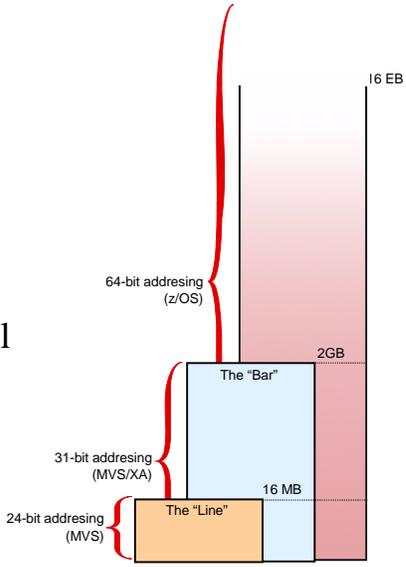
Lámina 23
Roberto Gómez C.



## El espacio de direcciones en z/OS

---

- z/OS soporta direcciones de 64 bits de largo
  - programa puede direccionar hasta 18,446,744,073,709,600,000 bytes (16 exabytes) de localidades de memoria.
- Dentro espacio direcciones el usuario puede lanzar varias tareas usando el TCB.
  - TCB: Task Control Block



The diagram illustrates the memory addressing capabilities of different systems. It shows three vertical bars representing memory ranges. The top bar, labeled '64-bit addressing (z/OS)', extends to 16 EB (exabytes). The middle bar, labeled '31-bit addressing (MVS/XA)', extends to 2 GB (gigabytes). The bottom bar, labeled '24-bit addressing (MVS)', extends to 16 MB (megabytes). The bars are stacked, with the 24-bit range at the bottom, the 31-bit range in the middle, and the 64-bit range at the top.

Lámina 24
Roberto Gómez C.



## Aislamiento espacio de direcciones

- Espacios direcciones en z/OS permite distinguir entre los programas y datos que pertenecen a cada espacio de direcciones.
- Espacio direcciones abarcan zonas públicas y privadas
  - áreas privadas en espacio direcciones del usuario son aislados de otras áreas privadas en otros espacios de direcciones.
  - espacio direcciones contienen áreas comunes que son accesible por cualquier otro espacio de direcciones.

Lámina 25 Roberto Gómez C.



## Comunicación espacios direcciones

- Aplicaciones requieren medios para comunicar entre espacios de direcciones.
- z/OS proporciona dos métodos de comunicación entre espacios de direcciones,.
  - Calendarizar un SRB (Service Request Block)
    - Proceso asíncrono.
    - Se lanza un proceso en otro espacio de direcciones o en el mismo.
  - Utilizar servicios tipo cross-memory y acceso a registros.
    - Proceso síncrono.
    - Permite acceso a un espacio de direcciones de otro usuario de forma directa.
    - Parecido al concepto de memoria compartida de Unix.
    - Requiere de autorizaciones especiales.

Lámina 26 Roberto Gómez C.



## DAT: Dynamic Address Translation

- Proceso de traducir una dirección virtual durante una referencia de almacenamiento a la correspondiente dirección real.
  - Si la dirección virtual ya se encuentra en memoria principal, el proceso DAT es más rápido.
  - Si la dirección virtual no se encuentra en memoria principal, ocurre una interrupción de fallo de página y se trae la página de la memoria secundaria.
- Implementado en hardware y software, haciendo uso de
  - Tablas de páginas, tablas de segmentos, tablas de regiones y buffers de traducciones.

Lámina 27 Roberto Gómez C.



## Campos dirección virtual

- Página
  - espacios direcciones son divididos en unidades de 4Kb de almacenamiento virtual llamadas páginas
- Segmento
  - unidades de 1 Mb
  - secuencias de direcciones virtuales
  - por ejemplo, espacio direcciones virtuales de 2Gb, consiste de 2048 segmentos
- Región
  - espacio direcciones son divididos en unidades de 2 a 8 gigabytes llamadas regiones
  - por ejemplo, espacio direcciones virtuales de 2Tb, consiste de 2048 regiones

Lámina 28 Roberto Gómez C.



## Formato dirección virtual

---

- Dirección virtual se divide en cuatro campos:
  - bits 0 - 32: RX, índice de región
  - bits 33 - 43: SX, índice de segmento
  - bits 44 - 51: PX, índice página
  - bits 52 - 63: BX, byte index

RX	SX	PX	BX
0	33	44	52
			63

- RX se encuentra dividido en tres campos
  - bits 0 - 10: RFX, Region First Index
  - bits 11 - 21: RSX, Region Second Index
  - bits 22 - 32: RTX, Region Third Index

RFX	RSX	RTX	
0	11	22	33

Lámina 29
Roberto Gómez C.

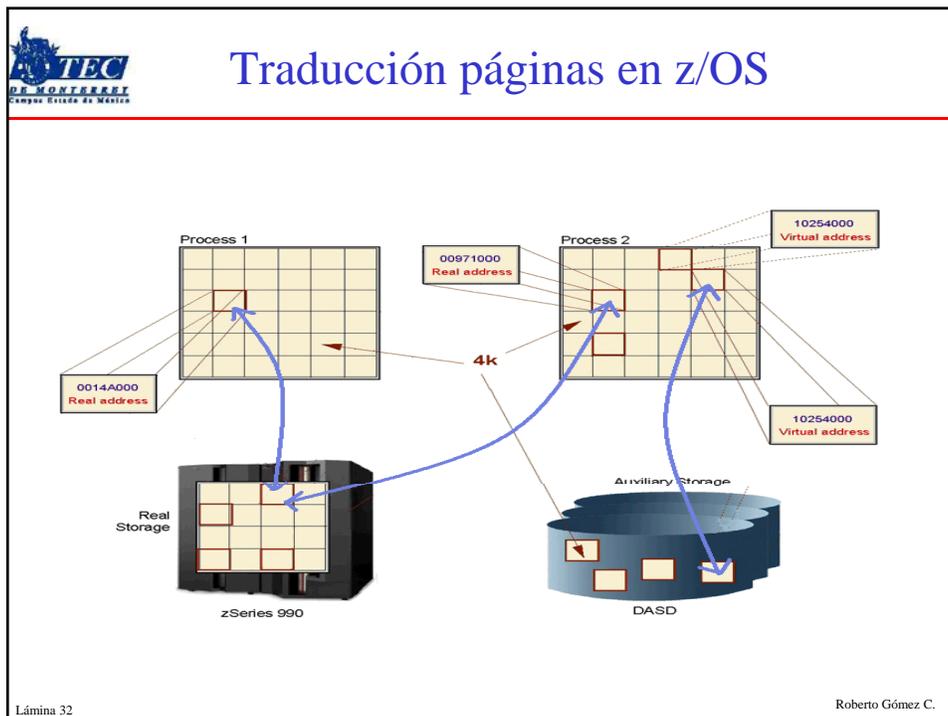
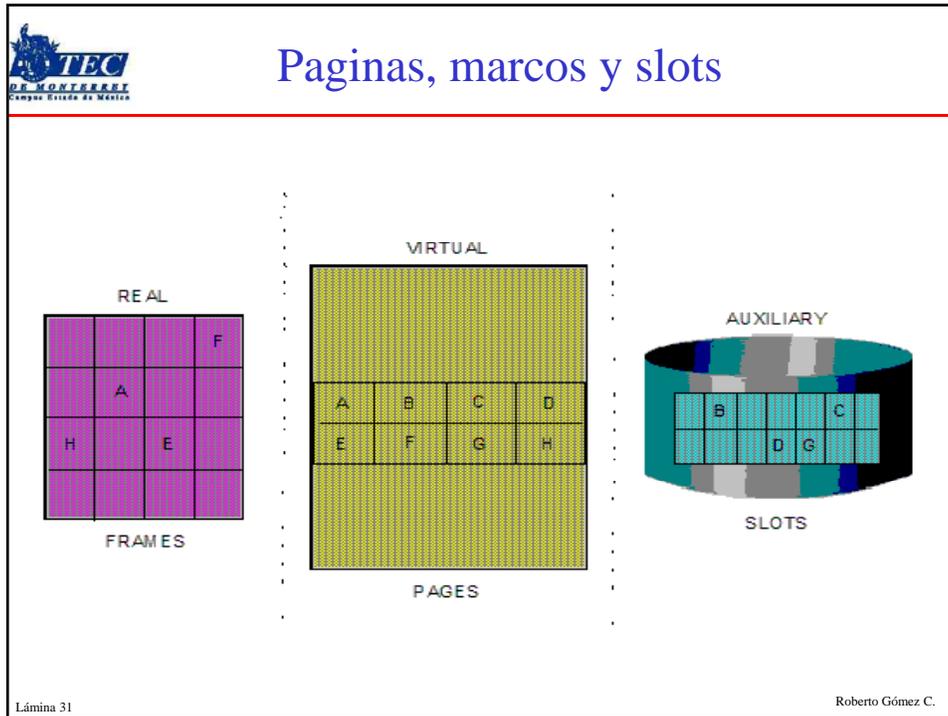


## Páginas, marcos y slots

---

- Piezas programa ejecutando en almacenamiento virtual debe ser movido entre almacenamiento real y auxiliar
  - Un bloque de almacenamiento real es un marco
  - Un bloque de almacenamiento virtual es una página
  - Un bloque de almacenamiento auxiliar es un slot
- Una página, un marco, un slot son del mismo tamaño:  
4096 byte = 4Kb
- Para el programador, el programa entero ocupa espacios contiguos en el almacenamiento real todo el tiempo.

Lámina 30
Roberto Gómez C.





## Paginación en z/OS

- Es transparente al usuario
- Page stealing.
  - Substituir un marco ocupado por otra página.
- Unreferenced interval count (uic)
  - Cuanto tiempo ha pasado desde que un programa referencio dicha página.
  - Bit de referencia.
    - Si esta apagado (no ha sido referenciado) se suma el número de segundos desde la última vez que fue referenciado al uic.
    - Si esta activado, (ha sido referenciado) el sistema lo apaga y asigna cero al uic.
    - Los marcos con iuc más grande son los utilizados.

Lámina 33 Roberto Gómez C.



## Swapping y working set

- Swapping
  - Swapping-in: espacio direcciones activo, las páginas están en la memoria principal y en memoria secundaria.
  - Swapping-out: espacio direcciones inactivo, las páginas residen en memoria secundaria y no pueden ejecutarse.
- Solo un subconjunto de las páginas del espacio de direcciones (working set) puede encontrarse en memoria principal.
- Swapping mueve todo el espacio de direcciones.
  - Uno de los métodos usado por z/OS para balancear la carga de trabajo.
- Swapping es llevado a cabo por el System Resource Management (SRM) en respuesta a recomendaciones del Workload Management (WLM).

Lámina 34 Roberto Gómez C.



## Protección de páginas

- z/OS usa las siguientes técnicas para preservar la integridad del trabajo de cada usuario.
  - Un espacio privado direcciones por usuario.
  - Protección páginas.
  - Protección de direcciones bajas.
  - Múltiples llaves de protección de almacenamiento.

Lámina 35 Roberto Gómez C.



## Llaves protección almacenamiento

- Usadas para prevenir cambios no autorizados en la memoria.
- Necesario contar con una llave para cambiar.
- Llave por cada 4K de memoria .
- Numeradas del 0 al 15
  - Llaves del 0 al 7 son usados por el BCP (Base Control Program) y varios subsistemas y productos middleware.
  - Llave 0: llave maestra.
  - Llaves 8 a 15: asignadas a usuarios.
- ¿Quién da seguimiento a todo esto?
  - La llave reside en el PWS (Program Status Word)

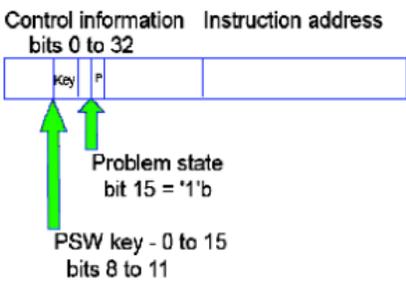
Lámina 36 Roberto Gómez C.



## El PSW

---

- Cada trabajo del sistema tiene asignado un PSW
- Entre otras cosas, indica la llave de protección de almacenamiento usada por la instrucción.
- Indica el estado en que la instrucción se encuentra corriendo.
- La llave se encuentra en los bits 8 a 11



Control information bits 0 to 32    Instruction address

Key    P

Problem state  
bit 15 = '1'b

PSW key - 0 to 15  
bits 8 to 11

Lámina 37
Roberto Gómez C.



## ¿Quién puede modificar la memoria?

---

- Cualquiera que cuente con la misma llave
- Cualquiera con llave 0
- Como obtener una llave 0
  - La mayor parte de los programadores OS cuentan con una llave 0
  - Ejecutar macro llamada MODESET
    - permite cambiar la llave de protección de almacenamiento que se encuentra dentro del PSW

Lámina 38
Roberto Gómez C.



## La buena noticia sobre MODESET

- No todo mundo puede usarla.
- Usuarios deben contar con uno o más privilegios especiales
  - Cualquiera (o cualquier programa) con llave 0 (actualmente, es probable que sean las llaves 0 - 7)
  - Supervisor (system) state
  - Authorized Program Facility

Lámina 39 Roberto Gómez C.



## Estados del sistema operativo y la APF

- Estados del sistema
  - Supervisor (system) state
    - trabajo realizado por el sistema
  - Problem program (user) state
    - trabajo realizado por el usuario
  - El sistema se encuentra en un estado o en otro
- Authorized Program Facility (APF)
  - no es un estado, sino una característica especial
  - un programa APF autorizado debe residir en una librería designada por APF que se encuentra en SYS1.PARMLIB
  - programa debe ser ligado con dicha librería

Lámina 40 Roberto Gómez C.



## Llaves del PSW

---

- Llaves de 0 a 7 son usadas por el BCP (Base Control Program) y varios subsistemas y productos middleware

0	MVS-OS/390-z/OS
1	JES
2-4	Reservada por IBM
5	Data management - DFSMS
6	VTAM
7	IMS
8	V=V (virtual) - batch, TSO users
9-15	V=R (real)

Lámina 41 Roberto Gómez C.



## Asignación llaves 8 a 15

---

- Llaves 8 a 15 son asignados a usuarios.
- Todos los usuarios se encuentran aislados en espacio de direcciones
  - aquellos usuarios cuyos programas corran en región virtual puede usar la misma llave de protección
    - estos usuarios son llamados V=V (virtual = virtual) y se les asigna una llave 8
  - algunos usuarios corren en memoria central
    - usuarios conocidos como V=R (virtual=real) y requieren llaves de almacenamiento individuales ya que sus direcciones no están protegidas por el proceso DAT

Lámina 42 Roberto Gómez C.



## El Program Properties Table

---

- Vista del sistema

```

SYSVIEW 7.4 CPU1                                PROGRAM PROPERTIES TABLE
Entries Available 28, deleted 0

Program  Orig  Ncan  Nswp  Priv  Syst  Ndsi  Npsw  Key  Affn  Sprf  Lprf  Nprf
-----  -
IEDQTCAM  IBM      NCAN  NSWP                SYST                NPSW      6  NONE                NPRF
ISTINM01  IBM      NCAN  NSWP                SYST                NPSW      6  NONE                NPRF
IKTCAS00  IBM      NCAN  NSWP  PRIV  SYST                NPSW      6  NONE                NPRF
AHLGTF   IBM      NCAN  NSWP                SYST                NPSW      0  NONE                NPRF
HHLGTF   IBM      NCAN  NSWP                SYST                NPSW      0  NONE                NPRF
IHLGTF   IBM      NCAN  NSWP                SYST                NPSW      0  NONE                NPRF
IEFLIC   IBM      NCAN  NSWP  PRIV  SYST                NPSW      0  NONE                NPRF
IEEMB860 IBM      NCAN  NSWP                SYST  NDSI  NPSW      0  NONE                NPRF
IEEVMNT2  IBM      NCAN  NSWP                SYST                NPSW      0  NONE                NPRF
HASJES20 IBM      NCAN  NSWP                SYST  NDSI      1  NONE                NPRF
DPSMVR00 USER    NCAN  NSWP                SYST                NPSW      7  NONE                NPRF
    
```

Lámina 43 Roberto Gómez C.



## Rol administradores almacenamiento

---

- Todo el manejo de memoria es manejado por componentes separados de z/OS
- Real storage manager (RSM)
  - seguimiento contenido de la memoria central
  - actividades paginación: page-in, page-out, page-stealing
- Auxiliary storage manager (ASM)
  - seguimiento de los slots en memoria secundaria
  - trabaja con RSM cuando se requiere insertar/retirar páginas para localizar los marcos de memoria principal y los slots de memoria secundaria
- Virtual storage manager (VSM)
  - obtener y liberar almacenamiento virtual
  - seguimiento memoria virtual de cada espacio de direcciones

Lámina 44 Roberto Gómez C.



## Breve historia direccionamiento

---

- 1970
  - Sistema/370 define direcciones almacenamiento de 24 bits de longitud
  - proporcionar direccionamiento de hasta 16MB de almacenamiento virtual
- 1983
  - Sistema/370-XA extiende direcciones a 31 bits
  - almacenamiento virtual: se extiende de 16M a 2GB
- 2000
  - Arquitectura z extiende direcciones a 64 bits
  - almacenamiento virtual: 16EB

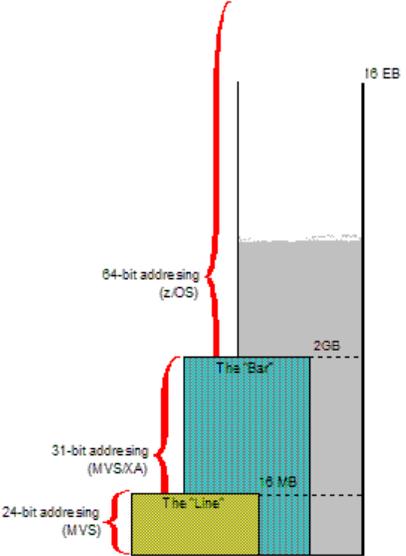
Lámina 45
Roberto Gómez C.



## Breve historia direccionamiento

---

- Dirección 16MB
  - punto división entre dos arquitecturas
  - conocido como la *línea*
- Preservar compatibilidad
  - MVS/XA no usa un bit
  - usado para diferenciar el tipo de dirección 31 bits (bit 0 on) o 24 bits (bit 0 off)
- Compatibilidad z/OS
  - disposición memoria igual bajo 2G (soporte 24/32 bits)
  - división conocida como la *barra*



64-bit addressing (z/OS)  
 16 EB  
 2GB  
 The "Bar"  
 31-bit addressing (MVS/XA)  
 16 MB  
 The "Line"  
 24-bit addressing (MVS)

Lámina 46



## Mapa direccionamiento 64 bits

---

- 0 a  $2^{31}$ 
  - misma disposición
- $2^{31}$  a  $2^{32}$ 
  - de 2GB a 4GB es considerada la barra
- $2^{32}$  a  $2^{41}$ 
  - área no compartida
  - empieza en 4GB
- $2^{41}$  a  $2^{50}$ 
  - area compartida
- $2^{50}$  a  $2^{64}$ 
  - área alta no compartida

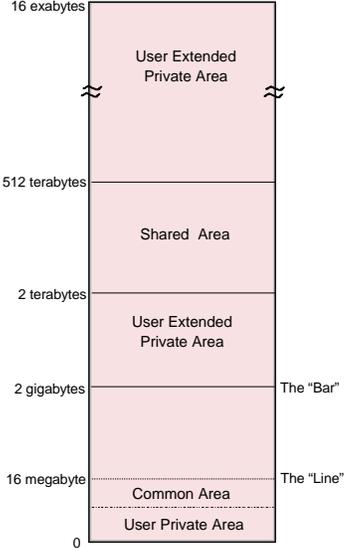


Lámina 47 Roberto Gómez C.



## Below-the-line-storage

---

- Programas y datos z/OS residen en almacenamiento virtual que, cuando es necesario, es respaldado por la memoria principal.
- La mayor parte de los programas no dependen de las direcciones reales.
- Algunos programas dependen de direcciones reales y alguno requieren que estas direcciones reales sean menores que 16 megabytes.
- Los programadores se refieren a este almacenamiento como el de “below the 16 megabyte line”.

Lámina 48 Roberto Gómez C.



## Residence mode

---

- Atributo conocido como *residence mode* o RMODE.
- Especifica si el programa debe residir (se cargado) en área de almacenamiento:
  - Por abajo de los 16 Megabytes: RMODE(24).
  - En cualquier parte de la memoria virtual: RMODE(31).
- Ejemplo programas RMODE(24)
  - Cualquier programa que asigne un DCB (Data Control Block).
  - Cualquier programa escrito antes MVS/XA.
- Nuevas aplicaciones se ejecutan con atributo RMODE(31).

Lámina 49
Roberto Gómez C.



## ¿Qué hay dentro de un espacio de direcciones?

---

- Memoria arriba 2GB
  - high virtual storage
  - solo programas en modo 64 bits
- Áreas extendidas arriba 16 MB
  - imagen espejo área abajo 16MB
- Nucleus
  - área Sistema Operativo
  - llave 0
- SQA
  - llave 0
  - área que contiene información del sistema compartida por diferentes espacios de memoria
- PLPA/FLPA/MLPA
  - contiene link pack areas
  - direccionable por programas que corren en modo 24 bits

Private	High User Region	16 EB
Shared Area	Default Shared Memory Addressing	512 TB
Low User Private	Low User Region	2 TB
Private	Reserved	4G
Extended Private	Extended LSQA/SWA/228/230	2G
Private	Extended User Region	
Extended Common	Extended CSA	
Common	Extended PLPA/FLPA/MLPA	
Private	Extended SQA	
Private	Extended Nucleus	16 MB
Common	Nucleus SQA	
Common	PLPA/FLPA/MLPA	
Common	CSA	
Private	LSQA/SWA/228/230	
Private	User Region	24 K
Private	System Region	8 K
Common	PSA	0

Lámina 50



## ¿Qué hay dentro de un espacio de direcciones?

---

- **CSA**
  - Common Area Storage
  - disponible para todas las aplicaciones
  - tamaño establecido en el IPL
- **LSQA/SWA/subpool 228/subpool 230**
  - usado por funciones del sistema cuando estás requieren espacios de direcciones aisladas
- **User Region**
  - cualquier programa que corra en el espacio de direcciones del usuario
  - direccionable por programas corriendo en modo de 24 bits
- **System Region**
  - área pequeña, solo 4 paginas
  - reservada para uso de *region control task* de cada espacio de direcciones
- **Prefix Save Area (PSA)**
  - referenciada como Low Core
  - área común de almacenamiento virtual para direcciones entre cero y 8191

Private	High User Region	16 EB
Shared Area	Default Shared Memory Addressing	512 TB
Low User Private	Low User Region	2 TB
	Reserved	4G
	Extended LSQA/SWA/228/230	2G
Extended Private	Extended User Region	
	Extended CSA	
Extended Common	Extended PLPA/FLPA/MLPA	
	Extended SQA	
	Extended Nucleus	16 MB
	Nucleus SQA	
Common	PLPA/FLPA/MLPA	
	CSA	
	LSQA/SWA/228/230	
Private	User Region	24 K
	System Region	8 K
Common	PSA	0

Lámina 51



## Espacio direcciones y el master scheduler

---

- Cuando arranca z/OS rutinas inicialización maestras inicializan los servicios del sistema.
  - Sistema bitácoras y de comunicación.
  - Se inicializa el espacio de direcciones del master scheduler.
- Después el master scheduler arranca JES.
- Después todos los sistemas son inicializados.
- Subsistemas son definidos en un archivo especial que contiene configuraciones del sistema.
  - El archivo se conoce como parameter library o PARLMLIB.
  - Estos subsistemas son subsistemas secundarios.

Lámina 52 Roberto Gómez C.



## Identificador y tipos espacios direcciones

- Espacios direcciones cuentan con un número asociado
  - Conocido como adress space ID ( o ASID )
  - El master-scheduler tiene asignado el ASID 1.
- Tipos espacio direcciones: Sistema, Subsistema y otros.
- Espacio direcciones sistema
  - Son creados después de la inicialización del master scheduler
  - Llevan a cabo funciones para otros espacios de direcciones en z/OS
- Espacio direcciones subsistema
  - JES y otros subsistemas como DB2, CICS e IMS
- Otros
  - Espacios direcciones TSO/E son creados por cada usuario que firma en z/OS
  - Espacio direcciones para cada trabajo en lote que corre sobre z/OS

Lámina 53 Roberto Gómez C.



## Administración carga de trabajo

- En z/OS la administración de los recursos es responsabilidad del componente de administración de carga de trabajo (WLM).
- Administra el procesamiento de cargas de trabajo en el sistema de acuerdo a los objetivos de la compañía, tales como tiempo de respuesta.
- WLM también administra el uso de los recursos del sistema, tales como procesadores y almacenamiento para cumplir con los objetivos.

Lámina 54 Roberto Gómez C.



## Objetivos WLM

- Cumplir con los objetivos del negocio que son definidos por la instalación, al asignar recursos a cargas de trabajo basado en su importancia y objetivos.
  - Goal achivement
- Lograr un uso óptimo de los recursos del sistema desde el punto de vista del sistema.
  - Throughput (rendimiento)
- Lograr un uso óptimo de los recursos del sistema desde el punto de vista del espacio de direcciones individual
  - Response y turnaround time (tiempo de entrega)

Lámina 55 Roberto Gómez C.



## Prioridades objetivos

- El cumplir con los objetivos es la primera y más importante tarea de WLM.
- Optimizar *throughput* (rendimiento) y minimizar tiempos de *turnaround* (tiempo respuesta) vienen después.
- Frecuentemente estos dos objetivos son contradictorios.
  - Optimizar rendimiento involucra el tener a los recursos ocupados.
  - Optimizar tiempo respuesta requiere que los recursos se encuentren disponible cuando sea necesario.
- WLM debe tomar decisiones que proporcionen un equilibrio entre objetivos contrapuestos.
  - El concretar el objetivo de un espacio de direcciones puede provocar en empeorar el tiempo de respuesta de un espacio de direcciones menos importante.

Lámina 56 Roberto Gómez C.



## Balanceo throughput y turnaround

- Para lograr un balanceo entre throughput y turnaround, WLM lleva a cabo las siguientes acciones
  - Monitorea el uso de recursos por parte de varias espacio de direcciones.
  - Monitorea el uso de recursos del sistema para determinar si son utilizados completamente.
  - Determina cuales espacios de direcciones intercambiar (swap out) y cuando.
  - Inhibe la creación de nuevos espacios de direcciones, o roba páginas cuando se da una escasez de almacenamiento central.
  - Selecciona los dispositivos a ser asignados, si existe una selección de dispositivos, para lograr un uso balanceado de dispositivos de E/S.

Lámina 57 Roberto Gómez C.



## Notificaciones al WLM

- Varios componentes de z/OS, administradores de transacciones y administradores de bases de datos pueden informar de un cambio de status al WLM.
- Ejemplos de notificación al WLM son
  - Almacenamiento central es configurado dentro o fuera del sistema.
  - Se va a crear un espacio de direcciones.
  - Un espacio de direcciones es borrado.
  - Un swap-out empieza o termina.
  - Rutinas de asignación puede elegir los dispositivos a ser asignados dada una petición/solicitud.

Lámina 58 Roberto Gómez C.



## ¿Cómo es usado WML?

- Instalación mainframes puede influir en casi todas las decisiones hechas por el WLM estableciendo un conjunto de políticas .
- A las cargas de trabajo se les asigna objetivos, (p.e. tiempo de respuesta promedio) e importancia (que tan importante es para el negocio que una carga de trabajo alcance sus objetivos).

Lámina 59 Roberto Gómez C.



## Supervisando ejecución trabajo en el sistema

- Para habilitar multiprogramación, z/OS requiere el uso de número de controles a nivel supervisor:
  - Interrupción de procesamiento.
  - Creación unidades de trabajo
    - TCB: Task Control Blocks
    - SRB: Service Request Blocks
  - Atención y ejecución de trabajos
  - Serializar el uso de recursos.
    - Enqueuing
    - Locking

Lámina 60 Roberto Gómez C.



## Procesamiento de interrupciones

- Interrupción: evento que altera la secuencia en la cual el procesador ejecuta instrucciones.
- Puede ser planeada o no-planeada.
- z/OS usa seis tipos interrupciones:
  - Supervisor calls o SVC interrupts
  - I/O interrupts
  - External interrupts
  - Restart interrupts
  - Program Interrupts
  - Machine check interrupts

Lámina 61 Roberto Gómez C.



## Supervisor calls o SVC interrupts

- Programa solicita un servicio del sistema.
- Se interrumpe el programa que se está ejecutando y se pasa el control al supervisor, de tal forma que lleve a cabo el servicio.
- Programas solicitan este tipo de servicios a través de macros como:
  - OPEN: abrir un archivo.
  - GETMAIN: obtener memoria
  - WTO: escribir un mensaje al operador del sistema

Lámina 62 Roberto Gómez C.



## Interrupción de E/S

- Ocurre cuando el subsistema de canales notifica de un cambio de status:
  - Se completo una operación de E/S.
  - Ocurrió un error.
  - Un dispositivo de E/S, i.e. impresora, esta lista para recibir trabajos.

Lámina 63 Roberto Gómez C.



## External interruptions

- Pueden indicar varios eventos importantes.
  - Expiración de un intervalo de tiempo.
  - El operador presionando la llave de interrupción en la consola.
  - Procesador recibiendo una señal de otro procesador.

Lámina 64 Roberto Gómez C.



## Restart Interrupts

- Ocurre cuando el operador selecciona la función de reinicio de la consola.
- También ocurre cuando una instrucción de reinicio SIGP (SIGnal Processor) de otro procesador es recibida.

Lámina 65 Roberto Gómez C.



## Program Interrupts

- Provocadas por errores en los programas
  - P.e. programa intenta llevar a cabo una operación inválida
- Se puede producir debido a fallos de páginas.
  - Página no se encuentra en memoria principal.
- También se generan a partir de una petición para monitorear un evento.

Lámina 66 Roberto Gómez C.



## Machine check interrupts

- Provocadas por un mal funcionamiento de la máquina.

Lámina 67 Roberto Gómez C.



## ¿Qué pasa cuando se da una interrupción?

- El hardware almacena información acerca del programa que fue interrumpido.
- Si es posible, deshabilita al procesador para futuras interrupciones del mismo tipo.
- El hardware le pasa el control a la rutina apropiada para que atienda la interrupción.
- El PSW (Program Status Word) es un recurso clave en este proceso.

Lámina 68 Roberto Gómez C.



## Program Status Word (PSW)

- Es un área de 128 bits del procesador junto con otra variedad de otros tipos de registros.
- Junto con otros registros, proporciona detalles cruciales tanto al hardware como al software
- El PSW actual incluye la dirección de la siguiente instrucción del programa e información de control acerca del programa que esta corriendo.
- Cada procesador cuenta con solo un PSW, por lo que solo una tarea se puede ejecutar en un procesador al mismo tiempo.

Lámina 69 Roberto Gómez C.



## PSW e interrupciones

- Cada procesador cuentan con un solo PSW, pero es útil pensar en tres tipos de PSWs para entender el procesamiento de una interrupción.
  - Actual PSW
    - Contiene la siguiente instrucción a ejecutar.
    - Indica para que interrupciones esta habilitado o deshabilitado.
      - Habilitado: la interrupción puede ocurrir
      - Deshabilitado:
  - Nuevo PSW
    - Contiene dirección rutina que puede procesar la interrupción asociada.
  - Viejo PSW
    - Sirve como temporal en el caso de una interrupción

Lámina 70 Roberto Gómez C.



## Interrupción y PSW

- Cuando ocurre una interrupción, si el procesador esta habilitado para dicha interrupción los PSWs son intercambiados:
  - PSW actual se almacena en el PSW viejo, con el tipo de interrupción que ocurrió.
  - Se carga contenido del PSW nuevo, dependiendo del tipo de interrupción, en el actual

Lámina 71 Roberto Gómez C.



## Registros y PSW

- Mainframe cuenta con registros para llevar un seguimiento de las cosas.
- El PSW es un registro usado para almacenar información requerida durante la ejecución de un programa.
- Se cuenta con otros registros
  - Access Registers
    - Especifica el espacio direcciones en donde se encuentran los datos.
  - General registers
    - Almacenar datos usuarios y direccionar datos almacenados

Lámina 72 Roberto Gómez C.



## Esquema PSW y registros

---

- Floiting point registers
  - Almacenar datos numéricos en forma de punto flotante
- Control registers
  - Utilizados por el mismo sistema operativo, p.e. referenciar tablas traducciones

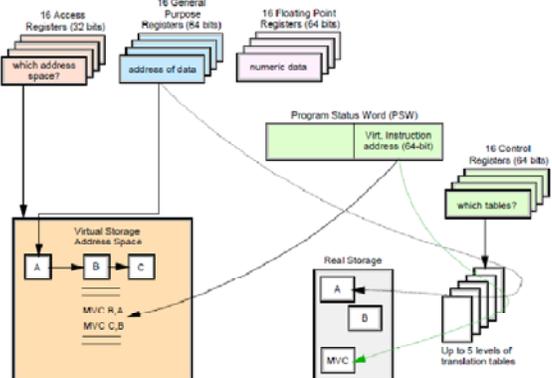


Lámina 73 Roberto Gómez C.



## Creando una unidad de trabajo

---

- Unidades de trabajo son representadas por dos tipos de control blocks:
  - Task control blocks (TCBs)
    - Representan tareas ejecutando dentro de un espacio de direcciones.
    - Programas usuario
  - Service request blocks (SRBs)
    - Peticiones para ejecutar una rutina de servicio del sistema.
    - Son creados cuando un espacio de direcciones detecta un evento que afecta a otro espacio de direcciones.
    - Proporcionan un mecanismo de comunicación entre espacio de direcciones.
    - Solo programas que corren en modo supervisor pueden crear un SRB.

Lámina 74 Roberto Gómez C.



## Creación de un TCB

---

- TCBs son creados en respuesta a un macro ATTACH.
- Usando este macro un programa de usuario o una rutina de sistema puede empezar la ejecución del programa especificado en el macro, como una sub-tarea de la tarea que llamó al macro.
- Como una sub-tarea el programa especificado puede competir por tiempo de procesador y puede usar algunos recursos ya asignados a la tarea que lo creó.
- Un TCB contiene información acerca de la tarea que se ejecuta, como la dirección de cualquier área de almacenamiento que haya creado.

Lámina 75 Roberto Gómez C.



## Creación de un SRB

---

- Se crea cuando un espacio de direcciones esta ejecutando y ocurre un evento que afecta otro espacio de direcciones.
- La rutina que lleva a cabo el servicio se conoce como SRB routine.
- El proceso de arranque se conoce como *scheduling an SRB*.
- La rutina SRB corre en un modo conocido como modo SRB.
- Solo programas corriendo en modo supervisor pueden crear un SRB.

Lámina 76 Roberto Gómez C.



## Prioridades SRB

---

- El programa que crea el SRB utiliza el macro SCHEDULE, indicando si el SRB cuenta con prioridad local (system-wide) o local (address space-wide).
- El sistema coloca al SRB en la fila de atención adecuada, donde permanecerá hasta se convierta en el trabajo de mayor prioridad en la fila.
- SRBs con prioridad global cuentan con una prioridad mayor a la de una local.
- SRBs con prioridad local cuentan con una prioridad similar al del espacio de direcciones donde será ejecutada, pero con mayor prioridad que cualquier TCB en dicho espacio.

Lámina 77 Roberto Gómez C.



## Preemptable vs non-preemptable

---

- Non-preemptable
  - Unidad de trabajo puede ser interrumpida.
  - Pero debe ser atendida una vez que se atendió la interrupción.
  - Ejemplo: SRBs
- Preemptable
  - Si es interrumpida, el control regresa al sistema operativo cuando la atención de interrupción es completada.
  - Ejemplo TCBs

Lámina 78 Roberto Gómez C.



## El despachador de trabajos

- Es responsable de ceder el control a la unidad de trabajo con la prioridad más alta que se encuentre listo para ser ejecutado.
- Elige el trabajo a ejecutar de acuerdo al siguiente orden:
  - Special exits
    - Salidas a rutinas que cuentan con una prioridad alta, debido a condiciones específicas en el sistema.
  - SRBs que cuentan con una prioridad global
  - Espacios de direcciones listos, de acuerdo a su prioridad.
- Si no hay ningún trabajo listo, z/OS asume un estado denominado *enabled state*.

Lámina 79 Roberto Gómez C.

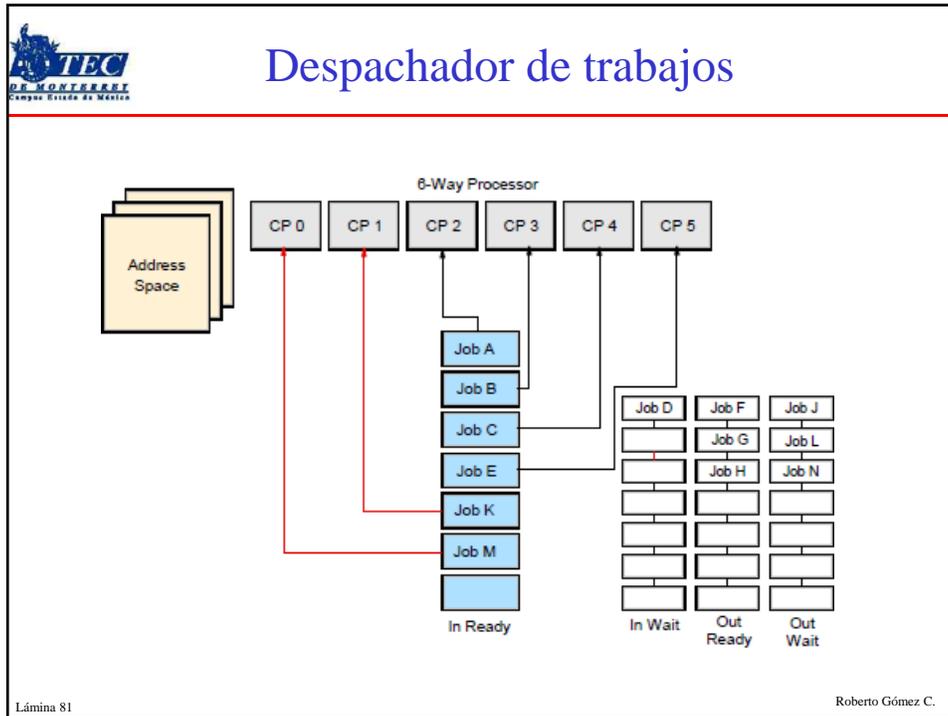


## Tipos colas de espera

- IN-READY
  - En almacenamiento central y esperando a ser despachado.
- IN-WAIT
  - En almacenamiento central y esperando por un evento.
- OUT-READY
  - Lista para ejecutar pero fuera de memoria.
- OUT-WAIT
  - Fuera de memoria y esperado por un evento.

*Solo trabajos en IN-READY puede ser seleccionado para atención*

Lámina 80 Roberto Gómez C.



- 
- The diagram is titled 'Global resource serialization'. It contains a bulleted list of points:
- Multiprogramación, multiprocesamiento requiere coordinación acceso a recursos.
  - Componente *Global Resource Serialization* procesa peticiones para recursos de programas corriendo en Z/OS.
  - Serializa acceso a recursos para proteger su integridad.
  - Cuando un programa solicita acceso a un recursos re-utilizable, el acceso puede ser solicitado como exclusivo o compartido.
    - Cuando se otorga acceso compartido a un recurso, no es posible otorgar un acceso exclusivo.
    - Cuando se otorga acceso exclusivo a un recurso, se debe esperar a que se libere para tener acceso a él.
- Lámina 82 Roberto Gómez C.



## Enqueuing

---

- Se lleva a cabo por las macros ENQ y DEQ.
- Para dispositivos corriendo entre varios sistemas z/OS es necesario usar las macros RESERVE y DEQ.
- En ENQ y RESERVE un programa especifica los nombres de uno o más recursos y solicita control compartido o exclusivo sobre estos.
  - Si los recursos se van a modificar debe solicitar acceso exclusivo.
  - Si no se van a modificar debe solicitar acceso compartido.
- Si el recurso no esta disponible el sistema suspende al programa solicitante hasta que el recurso este disponible.
- Cuando el recurso ya no es requerido se usa el macro DEQ para liberar dicho recurso.

Lámina 83 Roberto Gómez C.



## Locking

---

- Un candado (lock) es un campo que indica si un recurso esta siendo usado y quien lo usa.
- Dos tipos de locks:
  - Globales: para recursos relacionados con más de un espacio de direcciones.
  - Locales: recursos asignados a un espacio de direcciones en particular.
- Para usar un recurso protegido por un candado, una rutina debe solicitar el candado del recurso.
  - Si no esta disponible, la acción tomada por el solicitante depende si el candado es un *spin lock* o un *suspend lock*.

Lámina 84 Roberto Gómez C.



## Spin vs suspend

---

- Spin lock
  - Si no esta disponible, el solicitante continua probando el candado hasta que se libere.
  - Tan pronto como se libere el candado el solicitante puede obtener el candado y el recurso.
  - La mayor parte de los candados globales son de este tipo.
  - La entidad que cuenta con este tipo de candado debe ser deshabilitado para la mayor parte de las interrupciones.
    - Si es interrumpida es posible que nunca libere el candado.
- Suspend lock
  - Si no esta disponible, el solicitante se retrasa hasta que el candado este disponible.
  - Otro trabajo es atendido en el procesador hasta que el candado se libere.
  - Todos los candados locales son de este tipo.

Lámina 85 Roberto Gómez C.



## Prevención deadlock

---

- Para evitar deadlocks los candados son organizados jerárquicamente.
  - Un procesador o rutina solo puede solicitar candados de jerarquía mayor a los candados que actualmente tiene.
- Ejemplo deadlock
  - P1 cuenta con candado A y necesita candado B.
  - P2 cuenta con candado B y necesita candado A.
  - Lo anterior no puede ocurrir ya que los candados deben adquirirse de acuerdo a una secuencia jerárquica
    - Candado A precede a candado B en jerarquía
    - P2 no puede solicitar candado A mientras cuente con candado B
    - Debe liberar candado B, solicitar A y una vez que cuente con este solicitar B.

Lámina 86 Roberto Gómez C.



## Resumen características z/OS

- Uso espacio direcciones para asegurar aislamiento de áreas privadas.
- Diseñado para asegurar integridad de datos, a pesar de que tan grande sea la población de usuarios.
- Puede procesar un gran número de trabajos en lote concurrentes, con balanceo de carga automático.
- Permite incorporar seguridad en aplicaciones, recursos y perfiles de usuarios.
- Proporciona facilidades de recuperación extensivas, haciendo que el sistema sea re-inicializado muy pocas veces.

Lámina 87 Roberto Gómez C.



## Resumen características z/OS

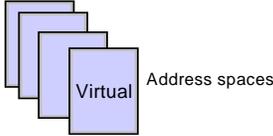
- Puede manejar cargas de trabajo mixtas.
- Puede manejar configuraciones grandes de E/S que incluyen 1000s de drivers de disco, librerías de cinta, impresoras, redes de terminales etc.
- Puede ser controlado desde una o mas terminales de operadores, o desde APIS que permiten la automatización de funciones rutinarias de operación.
- Interfaz operador es crítica para z/OS
  - Proporciona información sobre status del sistema, mensajes para situaciones de excepcionales, control de flujo de trabajo, y permite al operador manejar situaciones de recuperación inusual.

Lámina 88 Roberto Gómez C.



## Resumen facilidades de z/OS

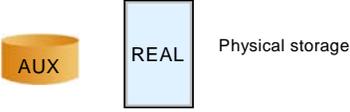
---



Virtual Address spaces



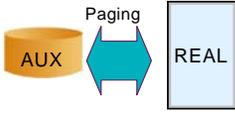
Operator communication



AUX REAL Physical storage



Reliability, availability, and serviceability



AUX REAL Paging



Data integrity

Lámina 89

Roberto Gómez C.



## Otros programas para z/OS

---

- Usualmente un sistema z/OS contiene programas licenciados adicionales (software con costo) necesitado para crear un ítem de trabajo práctico
  - Manejadores de seguridad
    - RACF
    - Algunos productos no IBM.
  - Manejadores de base de datos
    - DB2 (relacional)
    - Otros productos de BD (jerárquicas) se encuentran disponibles,

Lámina 90

Roberto Gómez C.



## Otros programas para z/OS

- Compiladores
  - C, COBOL, PL/1
- Utilerías para manejo de transacciones
  - Customer Information Control System (CICS)
  - Information Management System (IMS)
  - WebSphere Application Server para z/OS
- Programas para llevar a cabo ordenamientos de información (sort)
- Otros programas
  - SDSF: System Display and Search Facility

Lámina 91 Roberto Gómez C.



## Middleware para z/OS

- Middleware es algo entre el sistema operativo y un usuario final o aplicaciones de usuario final.
- Middleware proporciona funciones no disponible en el sistema operativo
  - Sistemas bases de datos
  - Servidores Web
  - Message queuing and routing functions
  - Transaction managers
  - Java virtual machines
  - XML processing functions

Lámina 92 Roberto Gómez C.



## Breve comparación z/OS y Unix

- Algunos conceptos son comunes a ambos

Unix	z/OS
Boot	IPL
Archivos	Data Sets
Editores: vi, ed, sed y emacs	ISPF
telnet o rlogin	TSO logon
Proceso, thread	Espacio de direcciones, TCB

Lámina 93 Roberto Gómez C.



## Introducción a TSO/E, ISPF y Unix

### Interactuando con z/OS

Lámina 94 Roberto Gómez C.



## Lo primero es lo primero...

- Una terminal 3270



- Emulador de la terminal 3270
  - VistaTN3270 <http://www.tombrennansoftware.com/order.html>

Lámina 95 Roberto Gómez C.



## ¿Cómo se interactúa con z/OS?

- TSO/E
  - permite usuarios logon a z/OS y usar un conjunto básico de comandos
  - conocido como TSO en su modo nativo
- ISPF
  - proporciona un sistema de menús para acceder la mayor parte de las funciones más usadas de z/OS
- z/OS UNIX shell y utilerías
  - permite usuarios escribir e invocar shell scripts y utilerías, y usa el shell programming language

Lámina 96 Roberto Gómez C.



## TSO/E

---

- Acronimo de Time Sharing Option/Extensions
- Permite a los usuarios crear una sesión interactiva con z/OS
- Proporciona la capacidad de un logon para un usuario y un interfaz prompt para comandos básicos de z/OS
- La mayor parte de los usuarios trabajan con TSO a través de su interfaz de menús ISPF
  - Interactive System Productivity Facility

Lámina 97 Roberto Gómez C.



## TSO

---

- En un sistema z/OS cada usuario cuenta con user ID y un password autorizado para TSO logon
- Durante el TSO logon, el sistema despliega la pantalla TSO logon en la terminal 3270 o el emulador TN3270
- Programadores sistema z/OS pueden modificar el layout y texto del panel TSO logon para cumplir con las necesidades de los usuarios del sistema

Lámina 98 Roberto Gómez C.

### Pantalla de inicio

The screenshot shows a terminal window titled "Vista Session A" with a menu bar (File, Edit, Font, Transfer, Macro, Options, Window, Help) and a toolbar. The main content is a black terminal window with white text. At the top, it says "TCP/IP MSG10 --> SOURCE DATA SET = SYS1.LOCAL.UTAMLST(USS20S14)". Below that is the date "06/06/06" and the word "W E L C O M E T O" followed by the time "08:30:22". A large graphic is formed by the characters "Z", "U", "O", and "S" arranged in a diamond shape. Below the graphic, it says "YOUR TERMINAL NAME IS : SC0TC234" and "YOUR IP ADDRESS IS : 10.128.36.27". The next line is "IBM Academic Initiative zSeries z/OS 1.4.0 Center". Below that is a warning: "\*\*\* NOT FOR COMMERCIAL USE \*\*\*\*" and "z/OS 1.4...z/OS 1.4...z/OS 1.4...z/OS 1.4.....". The final instruction is "==> ENTER 'L ' FOLLOWED BY THE APPLID YOU WISH TO LOGON TO. EXAMPLE 'L TSO' FOR TSO/E OR 'L C001' FOR CICS13 CICS APPLICATION.". At the bottom of the terminal window, it shows "0.0 06/06/06.157.07:33AM.zos.kctr.marist.edu A 24,1".

Lámina 99 Roberto Gómez C.

### Pantalla de inicio: logon

This screenshot is identical to the previous one, showing the same terminal window and content. The only difference is at the bottom of the terminal window, where the text "L TSO\_" is now visible, indicating that the user has entered their logon ID.

Lámina 100 Roberto Gómez C.



## Usando emulador wc3270

---

- Conectándose al servidor

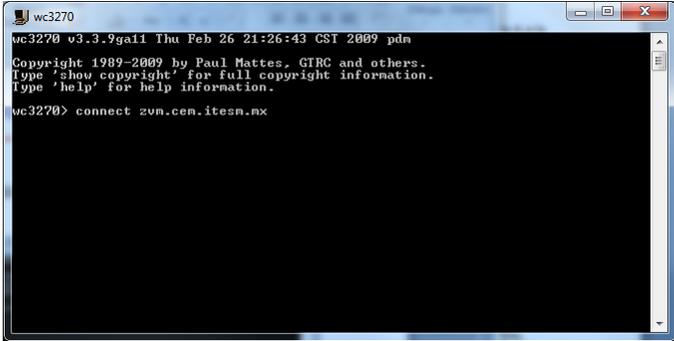


Lámina 101

Roberto Gómez C.



## Definiendo la “interfaz de conexión”

---

- Definir interfaz de conexión, no es necesario el userid ni su contraseña asociada.



Lámina 102

Roberto Gómez C.

 **Pantalla de inicio: logon fase 2**

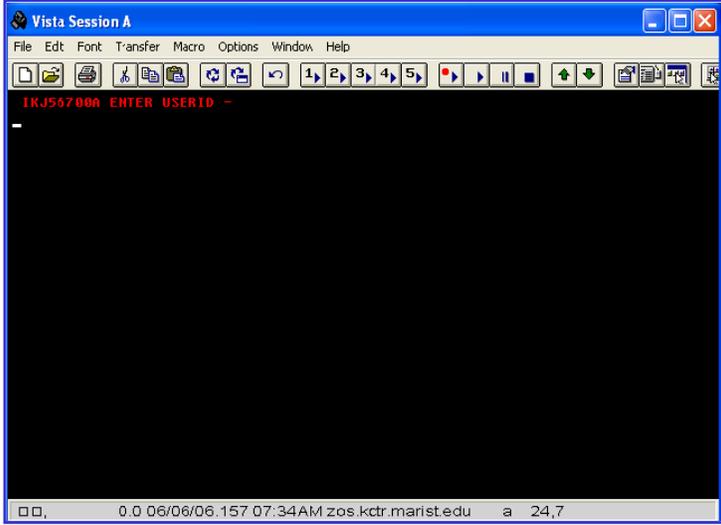


Lámina 103 Roberto Gómez C.

 **Pantalla de inicio: logon fase 3**

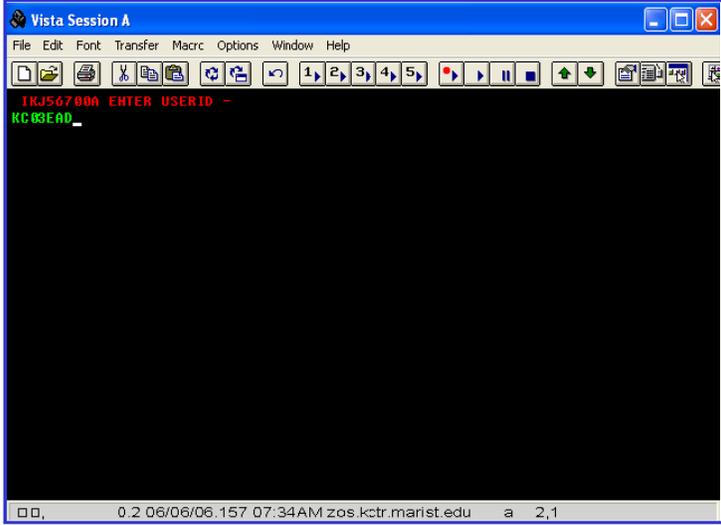


Lámina 104 Roberto Gómez C.

 **TSO/E logon screen**

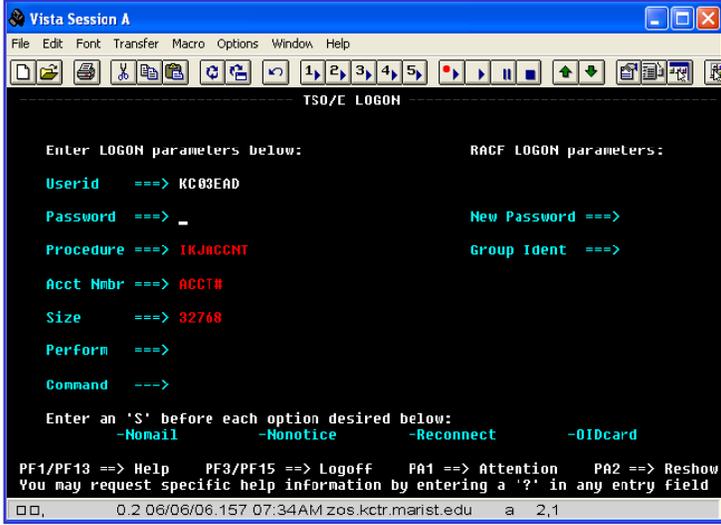


Lámina 105 Roberto Gómez C.

 **Adentro del sistema**

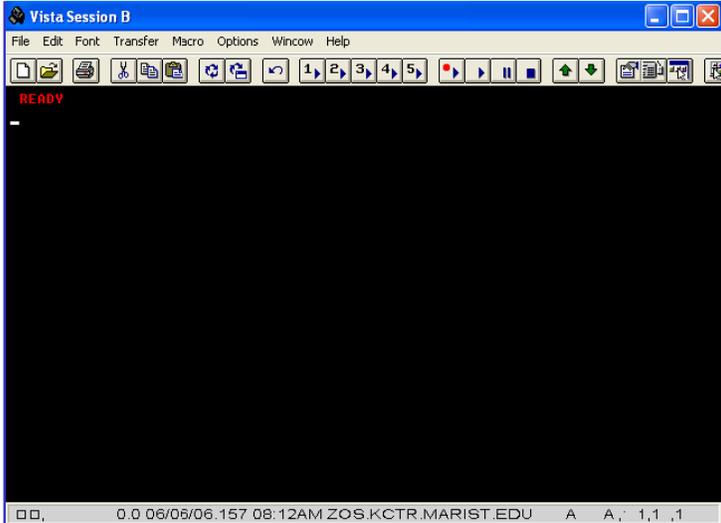


Lámina 106 Roberto Gómez C.



## Usando comandos TSO en modo nativo

---

- Usualmente, ISPF proporciona la interfaz para TSO.
- Sin embargo, TSO incluye un conjunto limitado de comandos básicos independiente de ISPF y otros programas
- Usar TSO de esta forma se conoce como usar TSO en modo nativo
- Cuando alguien firma a TSO, el sistema z/OS responde desplegando el prompt READ, y espera por entrada (similar al prompt de DOS)



Lámina 107

Roberto Gómez C.

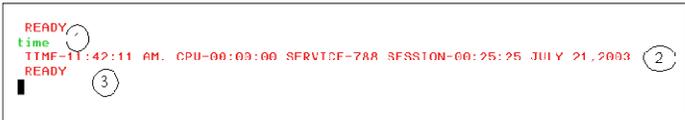


## El prompt READY del TSO

---



TSO esta listo para aceptar comandos  
Cursor, donde se introducen los comandos



- 1 - Se introduce un comando (como un prompt DOS)
- 2 - TSO despliega la salida del comando y
- 3 - TSO esta listo para aceptar nuevos comandos

Lámina 108

Roberto Gómez C.

Ejemplo de “ayuda” de la “ayuda”

Lámina 109 Roberto Gómez C.

Terminología de archivo de datos

- Archivos z/OS se conocen como data sets
- Antes de escribir datos en ellos, se debe reservar espacio para ello en el disco.
- Usuario debe especificar la cantidad de espacio así como el formato de este.
- Existen muchos terminos asociados cuando se asigna un data set
  - Volume serial
    - nombre seis caracteres de un disco o de un volumen de cinta, por ejemplo TEST01
  - Device type
    - Un modelo o tipo de dispositivo de disco, como el 3390

Lámina 110 Roberto Gómez C.



## Términos usados asignación data set

---

- **Organization**
  - El método de procesamiento de un data set, como secuencial
- **Record format**
  - Los datos son almacenado en registros, de tamaño fijo o variable
- **Record length**
  - La longitud (numero de caracteres) en cada registro
- **Block size**
  - Si los registros se encuentran contiguos para ahorrar espacio, esto especifica la longitud del bloque en caracteres
- **Extent**
  - Una asignación de espacio para mantener los datos.
  - Cuando el “extent” primario se llena, el sistema operativo automáticamente asignanara más extents, llamados secundarios
- **Space**
  - Espacio en disco es asignado en unidades llamadas bloques, tracks o cilindros

Lámina 111
Roberto Gómez C.



## Ejemplo uso TSO

---

- prompt READY del TSO logon

```

ICH700011 ZPROF  LAST ACCESS AT 17:12:12 ON THURSDAY, OCTOBER 7, 2004
ZPROF  LOGON IN PROGRESS AT 17:12:45 ON OCTOBER 7, 2004
You have no messages or data sets to receive.
READY
```

- Asignando un dataset desde la línea de comandos del TSO

```

READY
alloc dataset(zschol.test.cnt1) volume(test01) unit(3390) tracks space(2,1)
reclm(0) track(80) dsorg(ks)
READY
11steds
ENTER DATA SET NAME -
  zschol.test.cnt1
  ZSCHOL.TEST.CNT1
  --RECFM=LRECL-BLKSIZE=DSORG
  F      80      80      PS
  --VOLUMES--
  TEST01
  READY
```

Lámina 112
Roberto Gómez C.

 **Usando comandos TSO nativo para hacer un sort de datos**

```
READY
ALLOCATE DATASET(AREA.CODES) FILE(SORTIN) SHR
READY
ALLOCATE DATASET(*) FILE(SORTOUT) SHR
READY
ALLOCATE DATASET(*) FILE(SYSOUT) SHR
READY
ALLOCATE DATASET(*) FILE(SYSPRINT) SHR
READY
ALLOCATE DATASET(SORT.CNTL) FILE(SYSIN) SHR
READY
CALL 'SYS1.SICELINK(SORT)'
```

```
ICE143I 0 BLOCKSET SORT TECHNIQUE SELECTED
ICE000I 1 - CONTROL STATEMENTS FOR Z/OS DFSORT V1R5
          SORT FIELDS=(1,3,CH,A)

201 NJ
202 DC
203 CT
204 Manitoba
205 AL
206 WA
207 ME
208 ID
***
```

Lámina 113 Roberto Gómez C.

 **Programando usando TSO/E**



CLIST  
REXX Exec  
Command Processor

Lámina 114 Roberto Gómez C.



## ¿Qué es CLIST (Command List)?

- CLIST es un lenguaje interpretativo de alto nivel que permite a un usuario trabajar más eficientemente con TSO/E
- Permite manejar cualquier número de tareas
- Ya que es interpretativo, CLISTs son fáciles de probar no requieren un compilador

execute      correct <any> errors      re-execute

Lámina 115 Roberto Gómez C.



## Que se puede hacer con CLIST

- Escribir programas estructurados, realizar E/S, manejar excepciones y atender interrupciones.
- Operaciones lógicas y aritméticas sobre datos numéricos.
- Funciones de manejo de strings para procesar caracteres.
- Puede realizar tareas rutinarias (p.e. asignación de datasets)
- Proporciona aplicaciones interactivas usando ISPF

Lámina 116 Roberto Gómez C.



## Ejemplo

---

- Un archivo llamado AREA.COMMD

```

ALLOCATE DATASET(AREA,CODES)      FILE (SORTIN)           SHR
ALLOCATE DATASET(*)                FILE(SORTOUT)          SHR
ALLOCATE DATASET(*)                FILE(SYSOUT)           SHR
ALLOCATE DATASET(*)                FILE(SYSPRINT)         SHR
ALLOCATE DATASET(*)                FILE(SYSIN)            SHR
CALL 'SYS1.SICELINK(SORT)'
```

- Y para ejecutarlo:

```
EXEC 'CLIST AREA.COMMD'
```

Lámina 117 Roberto Gómez C.



## ¿Qué es REXX Exec (a.k.a. execs)?

---

- El REXX es lenguaje de alto nivel interpretativo que permite escribir programas de forma clara y estructurada
- Puede realizar numerosas tareas como invocar programas escritos en otros lenguajes.
- Realiza E/S y procesa datos aritméticos y de caracteres.
- Escribir aplicaciones interactivas usando ISPF

Lámina 118 Roberto Gómez C.



## CLIST vs REXX

---

- CLIST solo se ejecuta en un ambiente TSO/E mientras que REXX puede ejecutar en cualquier espacio direcciones MVS.
- Ambos ofrecen procesamiento tipo scripts
- Ambos son interpretativos, no compilados (aunque REXX puede ser compilado)
- Algunos usuarios z/OS escriben funciones directamente en forma de programas REXX o CLISTS
- Programación CLIST es única a z/OS, mientras que el lenguaje REXX es usado en varias plataformas.



Lámina 119

Roberto Gómez C.



## Ejemplo de REXX Exec

---

```

File Edit Edit_Settings Menu Utilities Compilers Test Help
-----
EDIT   KETTNER.REXX.CLIST(MYOMVS) - 01.00      Columns 00001
00072
Command ===>                               Scroll ===> PAGE
***** ***** Top of Data *****

000100 /* REXX OMVS */
000200 p = prompt("on"); /* dont suppress prompting */
000300 "omvs sessions(3) noshareas";
000400 x = prompt(p); /* restore original prompt state */
000500 return;
***** ***** Bottom of Data *****
    
```

Lámina 120

Roberto Gómez C.



## TSO/E Command Processor

- TSO/E proporciona comandos que permiten realizar una amplia variedad de tareas.
- Se puede definir un Command Processor para realizar la definición y mantenimiento de datasets y escribir y programar programas.
- Puede escribir un command processor para reemplazar o añadir al TSO un conjunto de comandos.
- Un command processor es un programa cuyo control es otorgado por el TMP (Terminal Monitor Programa) cuando un usuario teclea un comando en una terminal.
- El TMP proporciona la interfaz entre terminales de usuarios y el command processor proporcionando muchos servicios.

Lámina 121 Roberto Gómez C.



## ISPF Overview

- Acrónimo de Interactive System Productivity Facility.
- ISPF es una interfaz de menús para interacción con usuarios con el sistema z/OS
  - El ambiente de ISPF es ejecutado desde el TSO nativo.
- ISPF proporciona utilerías, un editor y aplicaciones ISPF al usuario.
  - Un usuario ISPF tiene acceso completo a la mayor parte de las funciones del sistema z/OS.

Lámina 122 Roberto Gómez C.



## Navegando a través los menús ISPF

---

- Para acceder ISPF bajo TSO, el usuario introduce un comando desde el prompt ready para desplegar el ISPF Primary Option Menu.
- Se puede acceder a ayuda en línea de cualquiera de los paneles ISPF (presionar la llave PF1).
- ISPF incluye un editor de texto un browser y funciones para localizar archivos y realizar otras funciones de utilerías.

Lámina 123
Roberto Gómez C.



## Primera vista ISPF

---

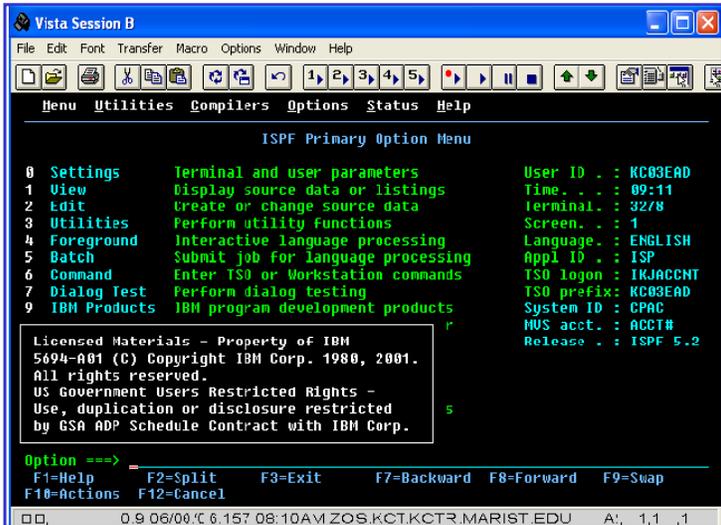


Lámina 124
Roberto Gómez C.



## Segunda pista ISPF

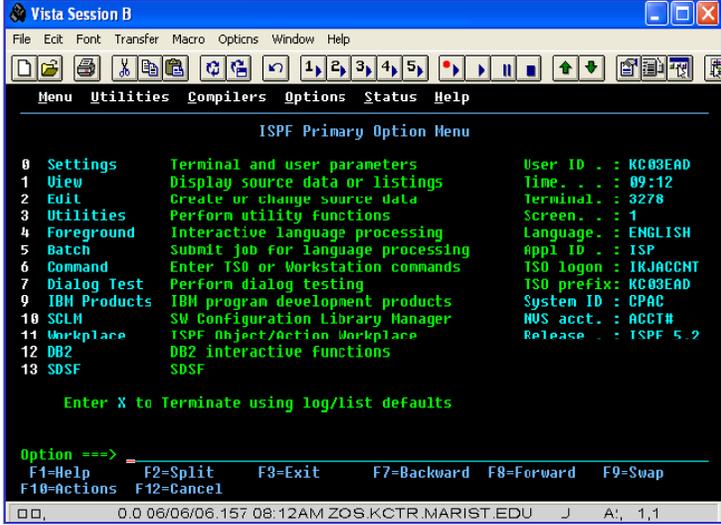


Lámina 125
Roberto Gómez C.



## Estructura general de los paneles ISPF

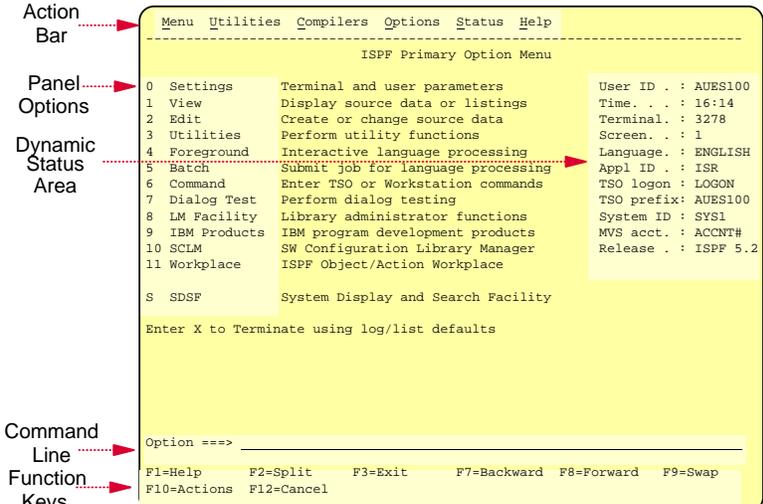


Lámina 126
Roberto Gómez C.



## Funciones comunes proporcionadas en los menús ISPF

---

**Action Bar**  
 Menu Utilities Compilers Options Status Help

-----

**Point-and-Shoot**

0	<b>Settings</b>	Terminal and user parameters
1	<b>View</b>	Display source data or listings
2	<b>Edit</b>	Create or change source data
3	<b>Utilities</b>	Perform utility functions
.		

**Option Number**

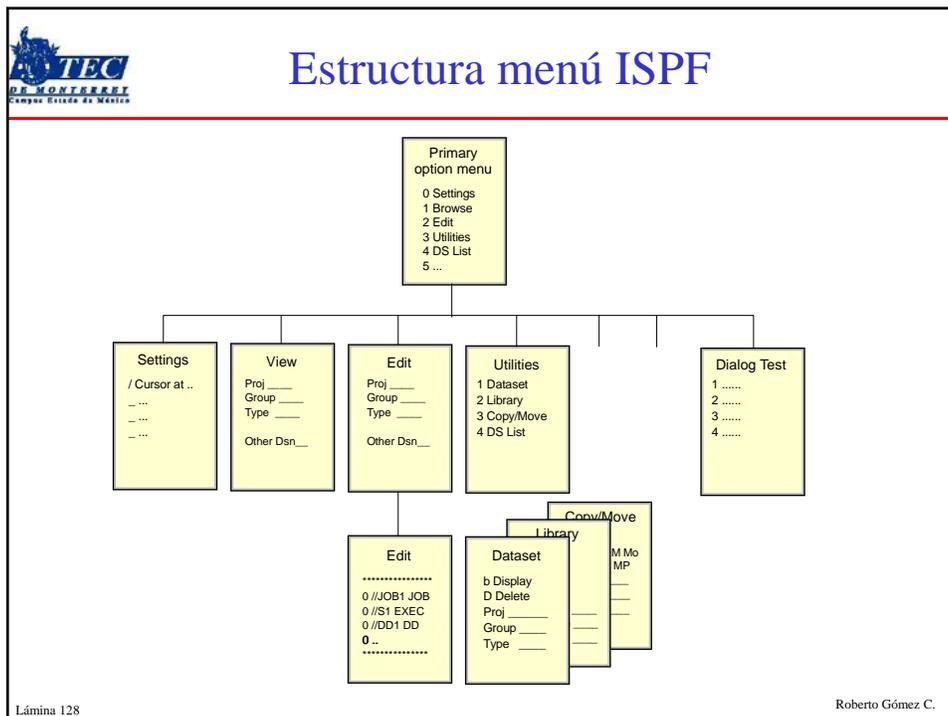
0	Settings	Terminal and user parameters
1	View	Display source data or listings
2	Edit	Create or change source data
3	Utilities	Perform utility functions
.		

Options ===> 3

**Function Keys**

F1=Help	F3=Exit	F7=Bkwd	F8=Fwd
F10=Actions	F11=Retrieve	F12=Cancel	

Lámina 127 Roberto Gómez C.





## Mapeo del teclado

---

Función	Llave
Enter	Ctrl (lado derecho)
Exit, end o ret	PF3
Help	PF1
PA1 o Attention	Alt-Ins o Esc
PA2	Alt-Home
Movimiento cursor	Tab o Enter
Clear	Pause
Página arriba	PF7
Página abajo	PF8
Scroll izquierdo	PF10
Scroll derecho	PF11
Reset locked keyboard	Ctrl (lado izquierdo)

Lámina 129
Roberto Gómez C.



## Primary Option Menu

---

```

2  Menu Utilities Compilers Options Status Help
-----
ISPF Primary Option Menu
3
0 Settings      Terminal and user parameters      User ID . . : USERID
1 View          Display source data or listings    Time. . . : 13:54
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 : ISPF
7 Dialog Test   Perform dialog testing             TSO prefix: USERID
9 IBM Products  IBM program development products   System ID : ISD1
10 SCLM         SW Configuration Library Manager   MVS acct. : IBMGSA
                                     Release . . : ISPF 5.5

License material - Property of IBM log/List defaults
All Rights Reserved .

Option ==>
F1=Help      F2=Split  F3=Exit    F/=Backward F8=Forward  F9=Swap
F10=Actions  F12=Cancel
    
```

1 Primary Options    2 Action Bar    3 Dynamic Status Area

Note: Some ISPF POM panels may likely provide copyright acknowledgement  
Depress ENTER to remove

Lámina 130
Roberto Gómez C.

**Primary Option Menu**

Menu Utilities Compilers Options Status Help

ISPF Primary Option Menu

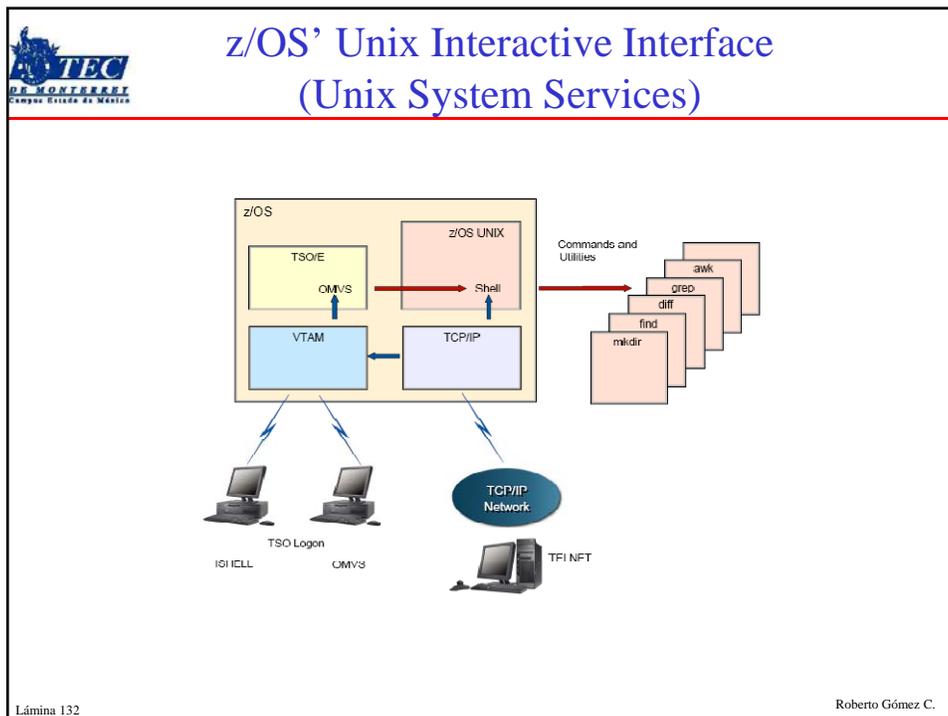
0 Settings	Terminal and user parameters	User ID . . : USERID
1 View	Display source data or listings	Time . . . : 13:54
2 Edit	Create or change source data	Terminal . : 3278
3 Utilities	Perform utility functions	Screen . . : 1
4 Foreground	Interactive language processing	Language . : ENGLISH
5 Batch	Submit job for language processing	Appl ID . . : ISR
6 Command	Enter ISO or Workstation commands	ISO logon : ISPF
7 Dialog Test	Perform dialog testing	ISO prefix: USERID
9 IBM Products	IBM program development products	System ID : IS01
10 SCLM	SW Configuration Library Manager	MVS acct. : IBMGSA
11 Workplace	ISPF Object/Action Workplace	Release . . : ISPF 5.5

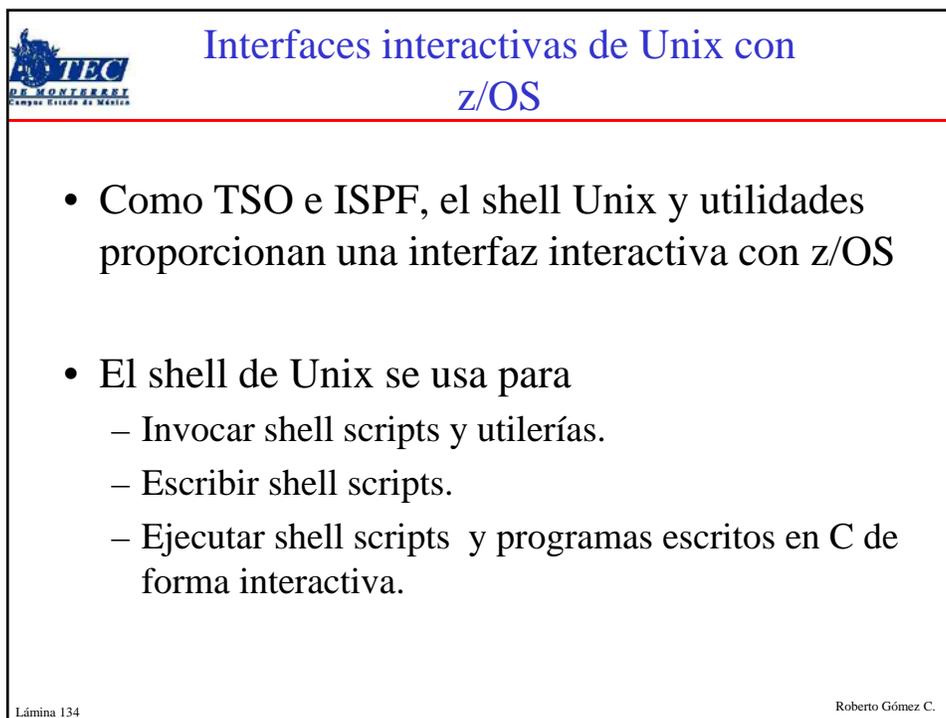
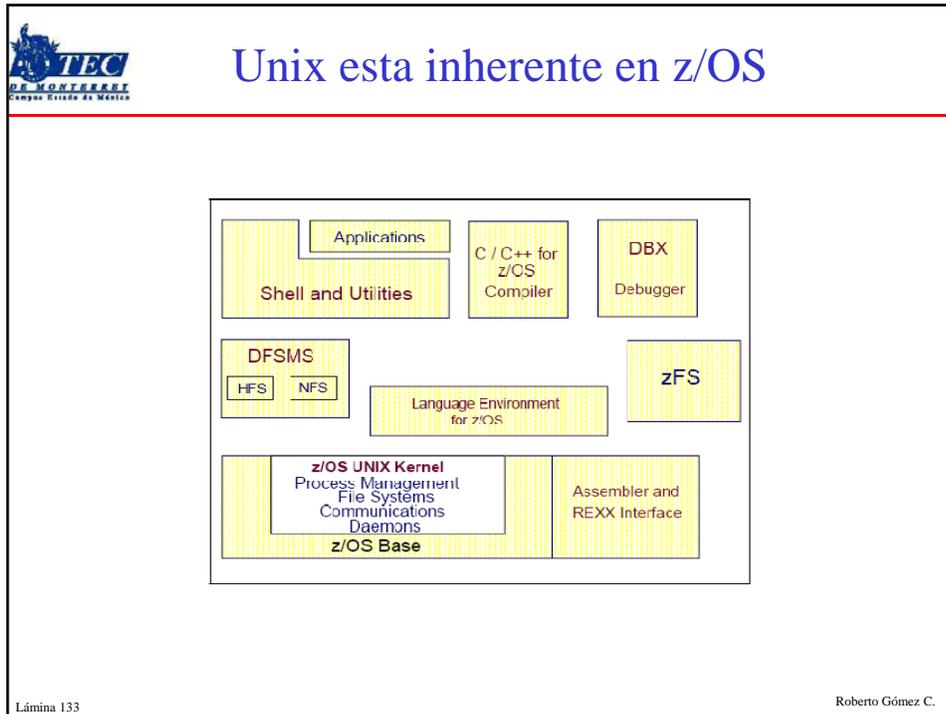
Enter X to Terminate using Log/List defaults

Option ==> \_\_\_\_\_  
 F1=Help F2=Split F3=Exit F7=Backward F8=Forward F9=Swap  
 F10=Actions F12=Cancel

1 Primary Options 2 Action Bar 3 Dynamic Status Area

Lámina 131 Roberto Gómez C.







## Invocando el shell de Unix

---

- Se puede invocar el shell de Unix en cualquiera de las siguientes formas
  - Desde una terminal 3270 o una estación de trabajo corriendo un emulador 3270
  - Desde una terminal TCP/IP conectada, usando el rlogin y comandos telnet
  - Desde TSO introduciendo el comando OMVS o el comando ISHELL

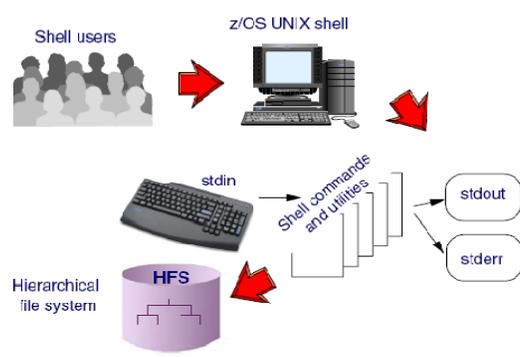
Lámina 135
Roberto Gómez C.



## Unix Shell

---

- z/OS Unix shell basado en el shell del Unix System V
- Algunas características de Unix Korn Shell
- Diferencias entre un comando y una utilería



The diagram illustrates the Unix shell architecture. At the top left, 'Shell users' (represented by a group of people icons) send input to the 'z/OS UNIX shell' (represented by a computer monitor and keyboard icon). Below this, a keyboard icon is labeled 'stdin'. A central box labeled 'Shell commands and utilities' is connected to 'stdin' and 'HFS' (Hierarchical File System, represented by a cylinder icon with a tree structure). The shell also outputs to 'stdout' and 'stderr' (represented by rounded rectangular boxes).

Lámina 136
Roberto Gómez C.



## Interfaces Unix con TSO

---

z/OS UNIX  
(z/OS Shell)  
OMVS command



- UNIX interface
- POSIX 1003.2
- Command interface

UNIX experienced user

ISPF Shell  
(ISHELL)  
ishell command



- ISPF based
- Menu interface

TSO experienced user

Lámina 137 Roberto Gómez C.



## OMVS Shell

---

```

IBM
Licensed Material - Property of IBM
5647-A01 (C) Copyright IBM Corp. 1993, 1998
(C) Copyright Mortice Kern Systems, Inc., 1985, 1996.
(C) Copyright Software Development Group, University of Waterloo, 1989.

All Rights Reserved.

U.S. Government users - RESTRICTED RIGHTS - Use, Duplication, or
Disclosure restricted by GSA-ADP schedule contract with IBM Corp.

IBM is a registered trademark of the IBM Corp.

$

====>

```

Enter any  
Unix command

↙

```

====>

```

RUNNING

ESC=⌘ 1=Help 2=SubCmd 3=HlpRetrn 4=Top 5=Bottom 6=TSO  
7=BackScr 8=Scroll 9=NextSess 10=Refresh 11=FwdRetr 12=Retrieve

Lámina 138 Roberto Gómez C.



## Desplegando directorios y archivos

---

```

$ ls -l
total 7
drwxr-xr-x  2 SMITHA  0          0 Dec  3 04:25 bin
drwxr-xr-x  2 SMITHA  0          0 Nov 19 15:16 doc
-rw-rw-rw-  2 SMITHA  0        250 Nov 17 23:07 etc
-rw-r--r--  2 SMITHA  0          0 Nov 17 23:07 fora
-rw-r--r--  5 SMITHA  0        1605 Dec  3 16:38 port
-rw-r--r--  2 SMITHA  0          0 Nov 17 23:15 script
drwxr-xr-x  2 SMITHA  0          0 Nov 17 23:07 src
drwxr-xr-x 15 SMITHA  0          0 Dec  3 20:37 projecta
$

```

}

```

===> ls -l

```

ESC=␣ 1=Help 2=SubCmd 3=HlpRetrn 4=Top 5=Bottom 6=TSO INPUT  
 7=BackScr 8=Scroll 9=NextSess 10=Refresh 11=FwdRetr 12=Retrieve

Lámina 139
Roberto Gómez C.



## Comandos TSO usados con z/OS Unix

---

- **ISHELL** - invoca el shell ISPF
  - Dirigido a usuarios más familiares con TSO/ISPF que Unix.
  - Proporciona paneles para trabajar con archivos Unix, montando y desmontando sistemas de archivos y administración Unix z/OS.
  - Programadores z/OS pueden hacer mucho de su trabajo bajo ISHELL.
- **OMVS** - comando invoca el shell z/OS Unix
  - Dirigido a usuarios más familiarizados con Unix que con TSO/ISPF.
  - Permite al usuario alternar entre el shell y TSO
  - Programadores Unix encontraran familiar el ambiente del z/OS Unix shell programming.

Lámina 140
Roberto Gómez C.



## ISHELL command (ish)

---

- Buen punto de inicio de usuarios de TSO/ISPF que desean utilizar z/OS Unix.
- Bajo ISHELL, es posible usar códigos de acción para
  - b desplegar un archivo o directorio
  - e editar un archivo o directorio
  - d borrar un archivo o directorio
  - r renombrar un archivo o directorio
  - a mostrar los atributos de un archivo o directorio
  - c copiar un archivo o directorio

Lámina 141
Roberto Gómez C.



## Panel principal de ISHell

---

```

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:  +
/
_____
_____

(C) Copyright IBM Corp., 1993, 2002. All rights reserved.
Command ==>
F1=Help   F3=Exit   F5=Retrieve F6=Keyshelp F7=Backward F8=Forward
F10=Actions F11=Command F12=Cancel

```

Lámina 142
Roberto Gómez C.



## Pull Down Menu Bar - ISH

---

```

File Directory Special file File systems Options Setup Help
-----
BPXWP  1. List directory(L)... Shell
        2. New(N)...
Enter  3. Attributes(A)... t an action bar choice. You can
also  4. Delete(D)... d line.
        5. Rename(R)...
A b1a  6. Copy to PDS(C)... of your working directory.
        7. Copy from PDS(I)... More:
        8. Print(P)
        9. Compare(M)...
        10. Find strings(F)...
        11. Set working directory(W)
        12. File System(U)
-----
Command ==>
F1= Help      F3=Exit      F5=Retrieve  F6=Keyshelp  F7=Backward  F8=Forward
F10=Actions   F11=Command  F12=Cancel
    
```

Lámina 143 Roberto Gómez C.

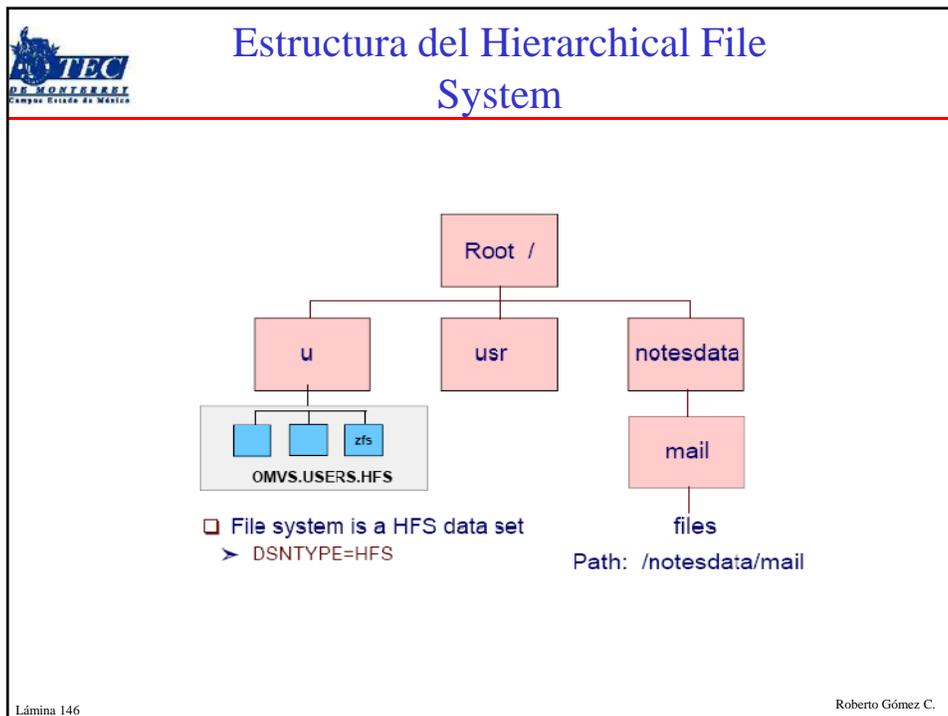
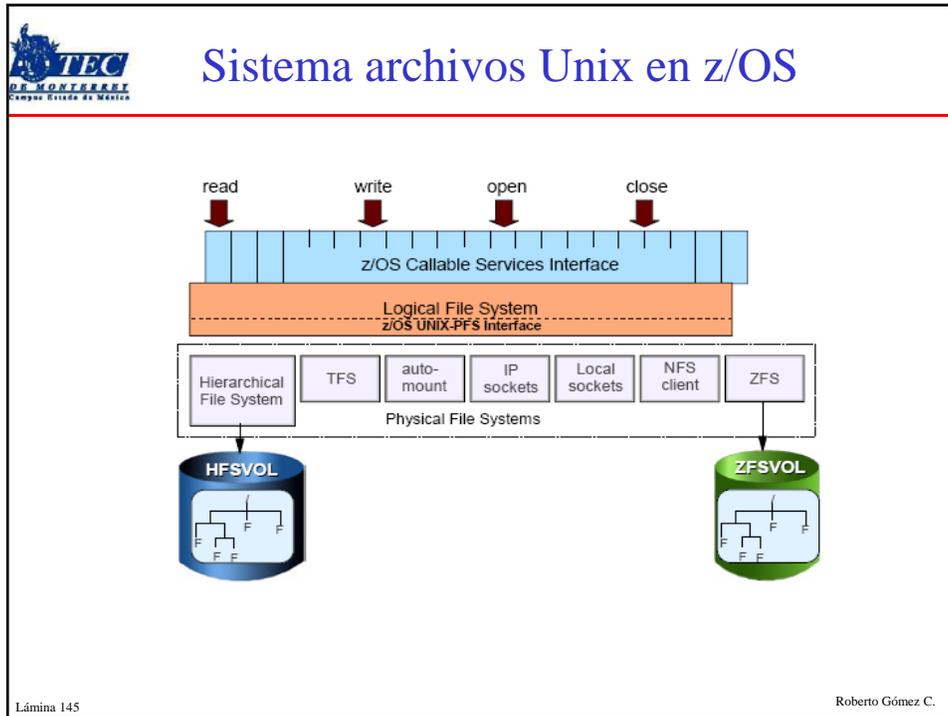


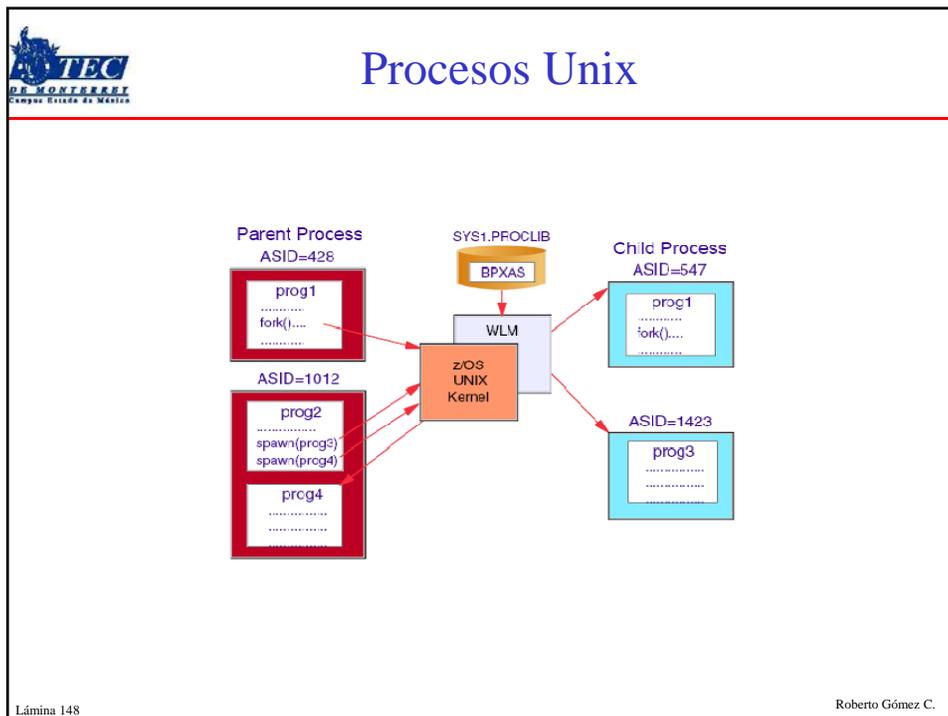
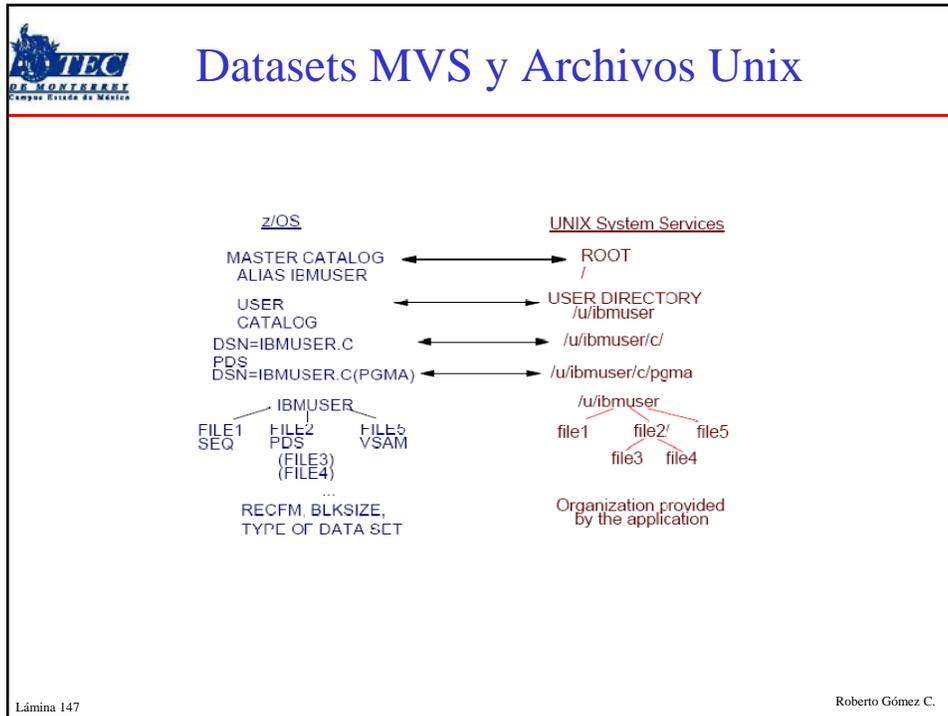
## OMVS command shell session

---

- Se utiliza el comando OMVS para invocar al z-OS Unix shell
- Bajo el shell de Unix, los usuarios pueden
  - Invocar comandos shell o utilidades que soliciten servicios del sistema
  - Escribir scripts shells usando el lenguaje de programación shell
  - Correr scripts de shell y programas escritos en C interactivamente (en foreground), en background o en batch

Lámina 144 Roberto Gómez C.







## login directo al shell

- rlogin
  - cuando el demonio inetd esta activo, se puede hacer un rlogin al shell desde una estación de trabajo
  - para firmar al sistema, utilizar la sintaxis del comando rlogin (remote log in) soportada
- telnet
  - también usa el demonio inetd
  - inetd debe estar activo y configurado para reconocer y recibir las peticiones de entradas de telnet

Lámina 149 Roberto Gómez C.



## z/OS, TSO e ISPF

Roberto Gómez Cárdenas  
rogomez@itesm.mx

Lámina 150 Roberto Gómez C.



## Resumen facilidades z/OS

- Espacios direcciones y almacenamiento virtual para usuarios y programas
- Almacenamiento virtual es respaldado por almacenamiento real y auxiliar.
- Movimiento programas y datos entre almacenamiento real y auxiliar se hace a través de paginación.
- Selección trabajo para ejecución, basado en prioridad y habilidad para ejecutar.
- Conjunto facilidades para manejo de archivos almacenados en disco o cinta
  - operadores usan consolas para arrancar y detener z/OS, introducir comandos y manejar el sistema operativo