

SISTEMAS OPERATIVOS

Linux

2014

Parte B



TRABAJO PRACTICO 3-7

“Cuando se haga popular Linux los virus van a aparecer y se va a infectar igual o peor que Windows”

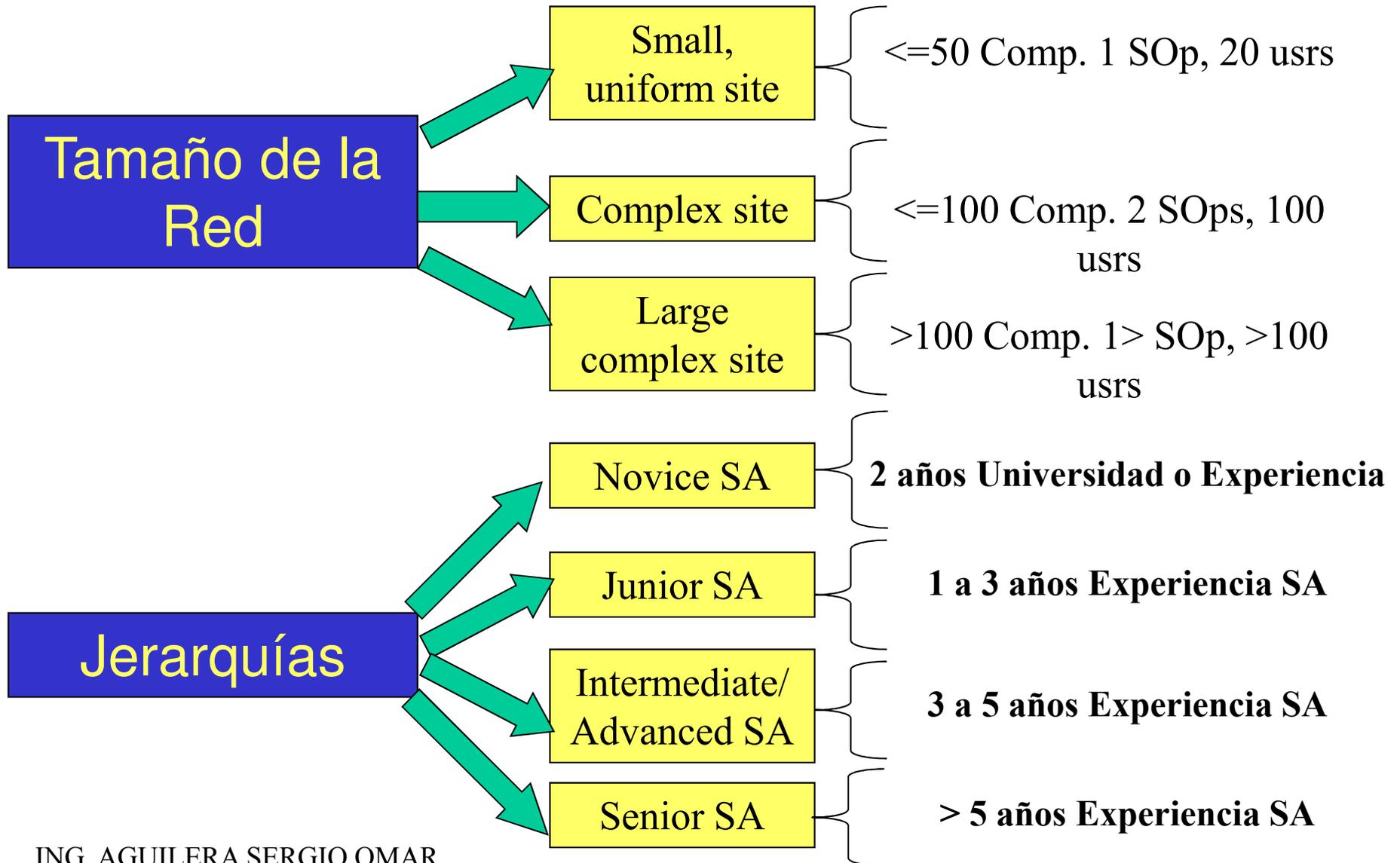
Linux ya es popular en: servidores, routers, firewalls, Dispositivos móviles y ultra portátiles. Sin embargo todavía no tiene virus

CONOCIMIENTOS DE UN ADMINISTRADOR DE SISTEMAS LINUX



- (1) PROGRAMACION –SCRIPTS, C-**
- (2) INSTALACION Y MANTENIMIENTO DE HARDWARE**
- (3) DOCUMENTACION**
- (4) TESTEO DE SISTEMAS**
- (5) DESARROLLO DE HCI (HUMAN COMPUTER INTERFACE)**
- (6) REDES DE COMUNICACION**
- (7) CAPACITACION DE USUARIOS EN LAS APLICACIONES COMUNES DEL MERCADO**
- (8) PACIENCIA Y DIPLOMACIA CON LOS USUARIOS**
- (9) CONOCIMIENTO DE LAS NORMAS CONTRACTUALES Y REGLAMENTACIONES INSTITUCIONALES**
- (10) DETECCION Y SOLUCION DE PROBLEMAS EN LOS SISTEMAS CONECTADOS Y/O EN SERVICIO**
- (11) CAPACIDAD DE EJECUCION DE LAS POLITICAS DE TRABAJO Y ADMINISTRACION DE LOS SISTEMAS DE DATOS CORPORATIVOS**

DEFINICIONES PARA PLATAFORMAS UNIX/LINUX SEGÚN LA SAGE *Certifications*



Actividades de Administrador



1. Encender el Servidor(es) – Controlar la inicialización

2. Usar “dmesg” para ver en pantalla la inicializacion completa.

Luego redireccionar a un archivo, para llevar registros cronológicos

```
# mkdir /var/log/regs_adicionales
```

```
# dmesg > /var/log/regs_adicionales/[ddmmaa].txt Ej. 210408.txt
```

3. Verificar cuando ingresaron los usuarios al sistema (en el tiempo que no estuvo presente)

```
# lastlog
```

Username	Port	From	Latest
root	pts/1	10.0.0.1	Fri Apr 25 19:27:29 – 0300 2008
man			**Never logged in**
proxy			**Never logged in**
backup			**Never logged in**
Irc			**Never logged in**
fedtez	pts/3	10.0.0.1	Fri Apr 25 22:15:15 – 0300 2008

Si hubiera accesos fuera de horario >>> Verificar porque¿?

Si hay robo de cuenta >>> inhabilitar la Cuenta

```
Abro /etc/passwd >>> # fedtez:x:2:3:felipe:/bin/bash
```

Actividades de Administrador



4. Verificar la actividad del Sistema. Voy a

`/var/log/messages`

Uso `less` – `grep` – `cat` (combinándolos)

ej. `#cat messages | grep xinetd` (controlo que pasa con el Server xinetd)

5. Grabar `/var/log/messages`, por día y luego borrar el original.

6. Verificar actividad de cada servicio. Ubicando los archivos de control en `/var/log` y en un Subdirector `.../samba`

7. Verificar actividad de cada File System: `/temp`, `/var`, `/usr/bin`, `/etc`

Usamos el `mc` (mignate commander). Uso los comandos `df` (disk free) & `free` para ver el Espacio en HD y RAM.

8. Verifico procesos en ejecución.

`# ps -ax` (controlo y elimino lo que no sirva y ocupa memoria)

9. Abrir la última tty y usar `top`. Esto me lista los procesos en ejecución y su actividad En tiempo real. Controlar cada 15 min. Eliminar los procesos dormidos.

10. Es recomendable reiniciar todos los servidores para que se Limpie la memoria, limpie Las colas, y haga un reinicio en limpio.

`# /etc/rc.d/init.d/xinetd restart`

ADMINISTRACION DE USUARIOS



Composición de una cuenta

Unix/Linux:

- (1) Login name (nombre del simbólico del usuario)
- (2) Password (palabra clave de acceso a la cuenta)
- (3) UID (Identificador de Usuario)
- (4) GID (Identificador de Grupo)
- (5) Directorio “home”: /home/pepe
- (6) Shell de trabajo : tsh, bash, etc
- (7) Usuario de mail, uso de alias
- (8) Casilla de mail: /usr/mail; /var/mail
- (9) Archivos de configuración e inicio de sesión

ADMINISTRACION DE USUARIOS



Cuentas Especiales:

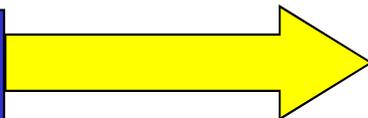
Username	UID	Proposito
root	0	Super usuario. Lo usa el Admin del Sistema. Sin restricciones
daemon	1	Dueño de los demonios del sistema.
bin	2	Dueño de los ejecutables standard provistos por el SOp.

ADMINISTRACION DE USUARIOS



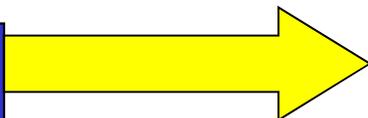
Archivos de Configuración de Cuentas de usuario:

/etc/passwd



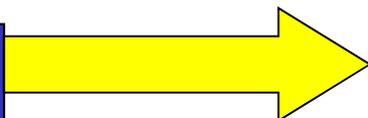
Login name - password encriptado (si no hay /etc/shadow - UID - GID - Campos de comentarios (GCOS) - /home/pepe - login shell

/etc/shadow



Listado de password encriptados

/etc/group



Información del sistema de Grupos, incluye:

Group name - GID - Group members

ADMINISTRACION DE USUARIOS



Formato del /etc/passwd:

```
<user>:<encrip>:<UID>:<GID>:<información>:<home>:<shell>
```

- <user>** → Nombre del usuario
- <encrip>** → Clave encriptada (sino hay /etc/shadow
- <UID>** → Identificado de Usr, para el kernel
- <GID>** → Ident. de Grupo al que pertenecemos
- <Info>** → Información complementaria
- <home>** → Directorio del usr
- <shell>** → Shell de inicio de la sesión

ADMINISTRACION DE USUARIOS



Configuración
del
/etc/passwd

```
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:
daemon:x:2:2:daemon:/sbin:
adm:x:3:4:adm:/var/adm:
lp:x:4:7:lp:/var/spool/lpd
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:
uucp:x:10:14:uucp:/var/spool/uucp:
operator:x:11:0:operator:/root:
games:x:12:100:games:/usr/games:
gopher:x:13:30:gopher:/usr/lib/gopher-data:
ftp:x:14:50:FTP User:/home/ftp:
nobody:x:99:99:Nobody:/:
gdm:x:42:42:./home/gdm:/bin/bash:
Xfs:x:100:233:X Font Server:/etc/X11/fs:/bin/false:
Trax:x:500:0:Bill Gates:/home/trax:/bin/bash
```

ADMINISTRACION DE USUARIOS



Creación de Usuarios:

```
/usr/sbin/adduser; /usr/sbin/useradd; /usr/bin/passwd
```

```
# adduser pepe  
# passwd pepe  
[type the password for the user pepe]
```

Baja de Usuarios:

```
/usr/sbin/userdel
```

```
# userdel [-r] [login]
```

ADMINISTRACION DE USUARIOS



Proceso interno de agregado de Usuarios:

- (1) Modificación del `/etc/passwd`, agregado de los datos propios
- (2) Adjudicación de clave de usuario. Uso del comando “`passwd usr`”
- (3) Creación del directorio `/home/usuario`, para nuestro usuario creado

ADMINISTRACION DE USUARIOS



Formateo de un floppy:

Formato Linux-ext2

- a) `mkfs.ext2 [opt] /dev/fd0`
- b) `Mke2fs [opt] /dev/fd0`

Nota: `/dev/fd0` con 666

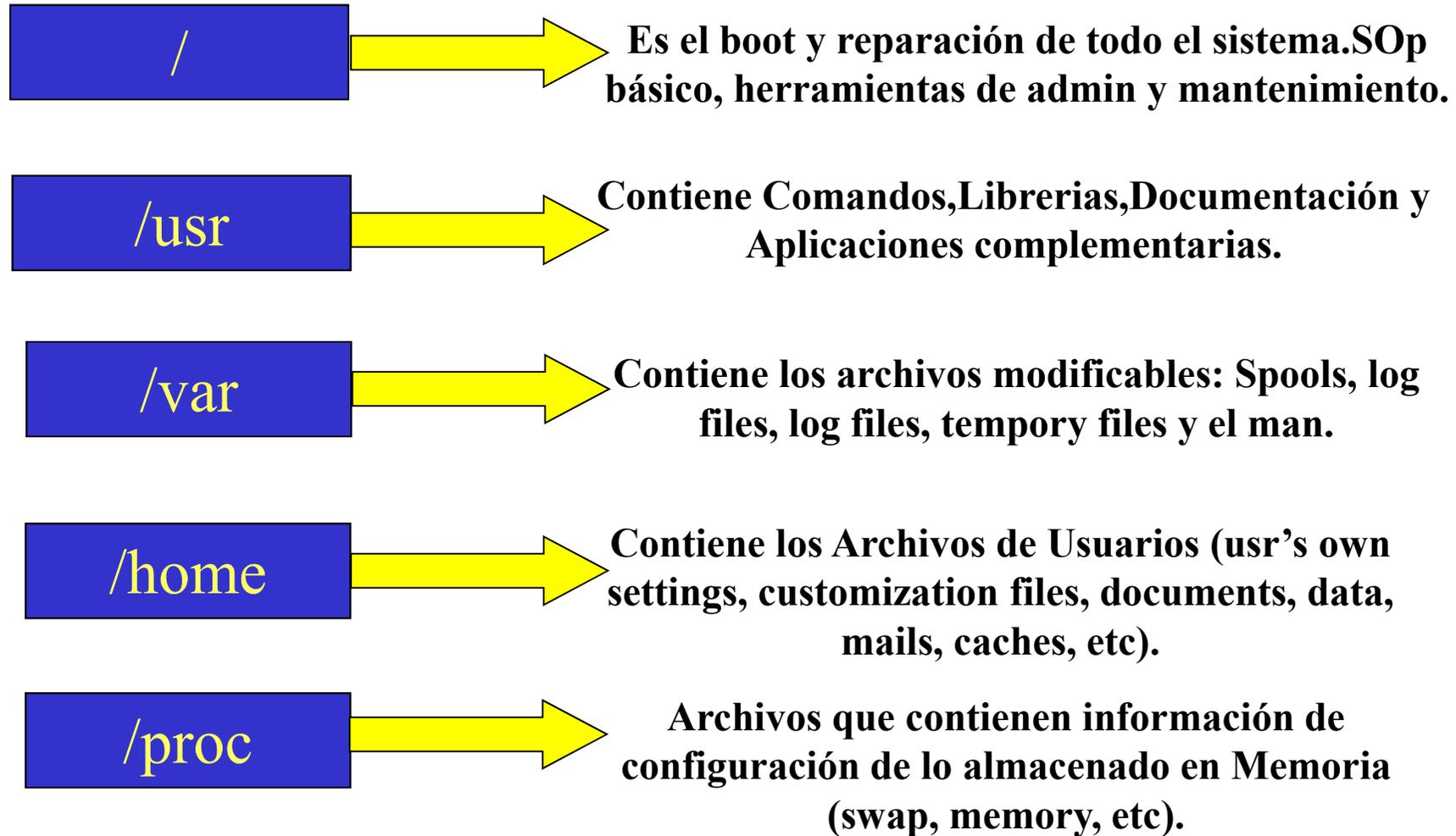
Formato msdos

- a) `mkfs.msdos [opt] /dev/fd0`
- b) `mkdosfs [opt] /dev/fd0`
- c) `mformat a:`

ADMINISTRACION DE USUARIOS



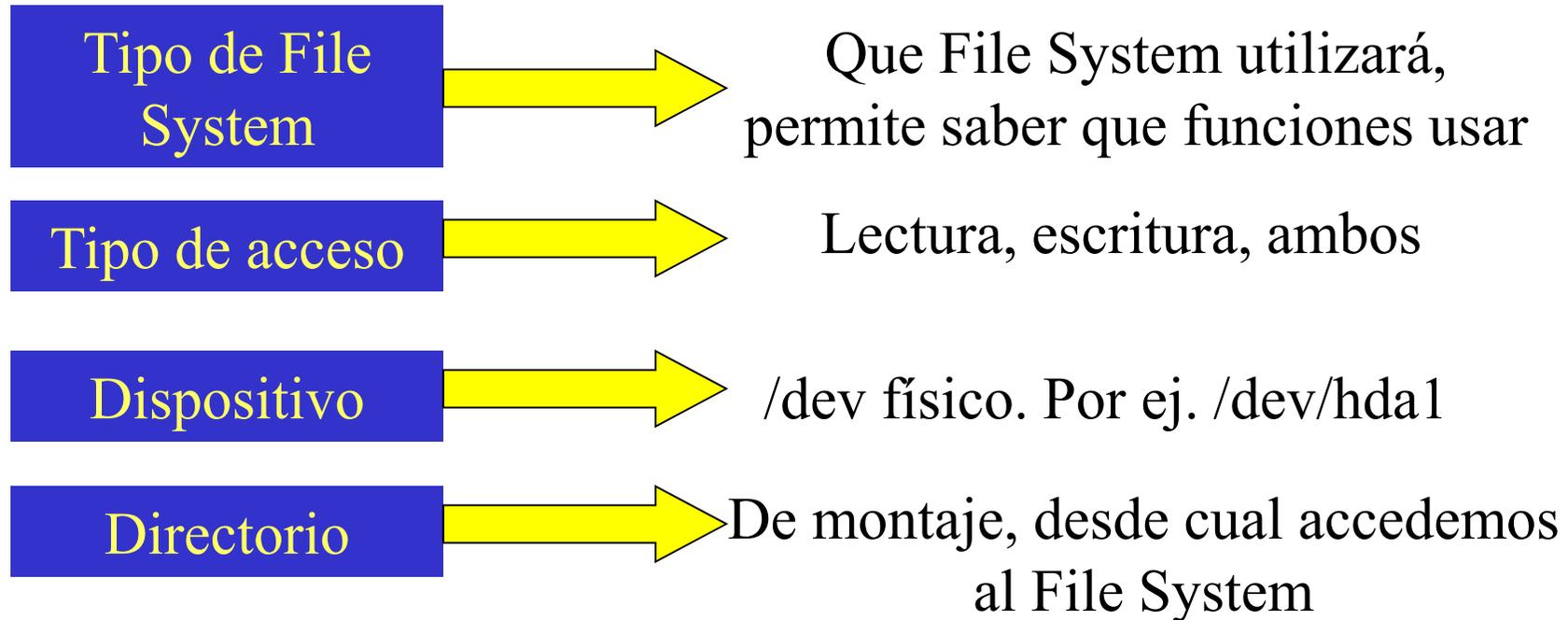
Files Systems de Linux



ADMINISTRACION DE USUARIOS



Montaje de Unidades en Linux:



```
# mount -t <File System> <Dispositivo> <Directorio>
```

```
# umount <Directorio>
```

ADMINISTRACION DE USUARIOS



Montaje de Unidades en Linux:

```
# mount -t iso9660 /dev/hdb /mnt/cdrom
```

```
# mount -t auto /dev/fd0 /mnt/floppy
```

```
# mount -t vfat /dev/sda4 /mnt/zipdrive
```

```
# mount -t vfat /dev/hda1 /mnt/dosdrive
```

(Permite que distintos usrs hagan r/w en la partición DOSWin)

```
# mount -t vfat -o user,rw,exec,umask=000 /dev/hda1 /mnt/dosdrive
```

(No olvidar)

```
# chmod a+rx /mnt/floppy
```

ADMINISTRACION DE USUARIOS



Archivo /etc/fstab (contenido):

#	<device>	<mountpoint>	<filesystemtype>	<options>	<dump>	<fsckorder>
	/dev/hda2	/	ext2	defaults		
	/dev/hda3	/usr	ext2	defaults		
	/dev/sda1	/home	ext2	defaults		
	/dev/hdb	/mnt/cdrom	iso9660	user,noexec,nodev,nosuid,ro,noauto		
	/dev/fd0	/mnt/floppy	vfat	user,noexec,nodev,nosuid,ro,noauto		
	None	/proc	proc	defaults		
	/dev/hda4	/swap	swap	defaults		
	/dev/hda1	/mnt/dos	vfat	exec,dev,nosuid,rw,auto		

Nota: defaults: rw,suid,dev,exec,auto,nouser,async

ADMINISTRACION DE USUARIOS



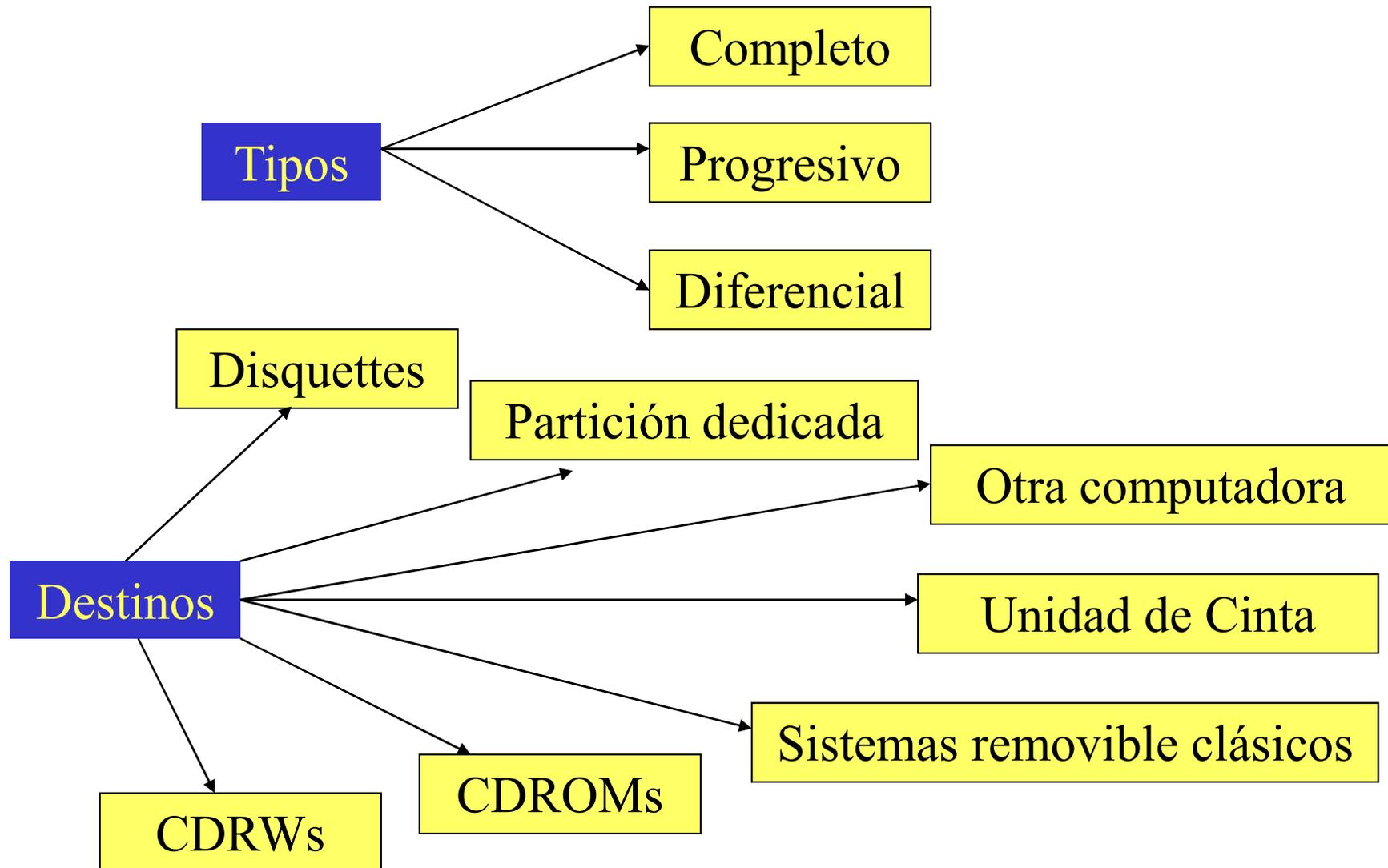
Creacion de un floppy de boot:

```
[/home/pepe]$ /sbin/mke2fs /dev/fd0 #format en ext2
[/home/pepe]$ [ -d /fd ] || mkdir /fd # Si no existe /fd lo crea
[/home/pepe]$ mount /dev/fd0 /fd # monta el floppy en /fd
[/home/pepe]$ cp /boot/boot.b /fd # Copia el kernel de Linux y
[/home/pepe]$ cp /vmlinuz /fd # archivo de boot
[/home/pepe]$ echo image=/fd/vmlinuz label=linux | /sbin/lilo -c - -b
/dev/fd0 -i /fd/boot.b -c -m /fd/xx # Activa el kernel en el floppy
[/home/pepe]$ umount /fd #Desmonta el floppy
[/home/pepe]$ rmdir /fd # borra el /fd
```

ADMINISTRACION DE USUARIOS



Resguardo (Backup):



INTERFAZ DE ADM. DE ARCHIVOS



a. Apertura y Cierre de Archivos (open y creat)

`int open (const char, *pathname, int flags)`

`int open (const char, *pathname, int flags, mode_t mode)`

`int creat(const char, *pathname, mode_t mode)`

equivale a:

`open(pathname,
O_CREAT|O_TRUNC|O_WRONLY, mode)`

b. Cierre de Archivos.

c. Lectura/Escritura.

d. Truncamiento.

ADMINISTRACION DE USUARIOS



Comandos de Backup:

Comando	Plataforma	Características
dump/restore	BSD Systems	Imagen de resguardo. Para multiples volumenes.
tar	Todo Unix/Linux	Bkup archivo por archivo. En general no soporta multiples volumenes.
cpio	AT&T systems	Idem anterior, pero soporta multiples volumenes.

ADMINISTRACION DE USUARIOS



dump/restore:

```
# dump [options [ arguments]] file system  
# dump [options [ arguments]] filename
```

```
Ej: # dump 0dsdfu 54000 6000 126 /dev/rst2 /usr  
(Bkup del /usr sobre una cinta de 2.3 Gb, 8mm,  
conectada desde /dev/rst2)
```

```
# restore -irRtx [modifiers] [filenames]
```

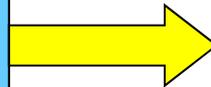
ADMINISTRACION DE USUARIOS



Comandos de Backup:

```
# tar -function [modifier] device [files]
```

function



c: Crea un nuevo bckup

r: Reemplaza, los nombre de los files son creados al final.

t: Tabla, lista los archivos en backup y sale por stdout.

u: Update, para backup incremental

x: Extract, permite restablecer un backup.

```
# tar cvf /hom/joe/backup.tar /home/joe/data
```

ADMINISTRACION DE USUARIOS



modifier



v: Verbose, reporta que esta haciendo con un listado de los archivos en backup.

w: Muestra las operaciones a realizar, y espera confirmación.

f: File, nombre del archivo a usar para el bkup. Normal o /dev (/dev/rdisk/5h, floppy de alta densidad)

m: Modify, no hacer restore de la hora de modificación o restore (extract).

t: Print del listado de archivos en backup.

o: Ownership, usa el UID o GID de quien hace el backup.

```
# tar -wvf temp.tar
```

```
# tar -xf temp.tar hello.dat
```

```
# tar -cv /dev/rtm0 /home
```

ADMINISTRACION DE USUARIOS



Comandos de Backup:

```
# cpio -options [>][<] device [files]
```

Options



o: Copia de salida. Stdin por defecto.

B: Bloquea I/O a 5120 k por registro. Util para backup en cinta.

i: Copia de entrada. Extract de la entrada standard.

t: Crea tabla de contenido de la entrada.

```
# ls /home |cpio -o > /dev/fd0
```

```
# cpio -it < /dev/fd0 >bkup.indx
```

```
# find /home -mtime 1 -type f -print | cpio -oB > /dev/fd0
```

ADMINISTRACION DE USUARIOS



Programación de Actividades:

at: Ejecuta un listado de comandos en un tiempo determinado.

batch: Idem, pero en un tiempo determinado por el Sop.

cron: Idem, pero lanza ejecución periódica

```
# at-qlr [cola hora] [ids-jobs] batch
```

Formato	Accion
at hh:mm	Programa job en 24h.
at hh:mm month day year	Idem pero varios y años.
at -l	Lista jobs programados
at-r job_id	Cancela el job con número job_id

ADMINISTRACION DE USUARIOS



Programación de Actividades:

```
# crontab -led [-u usuario]
```

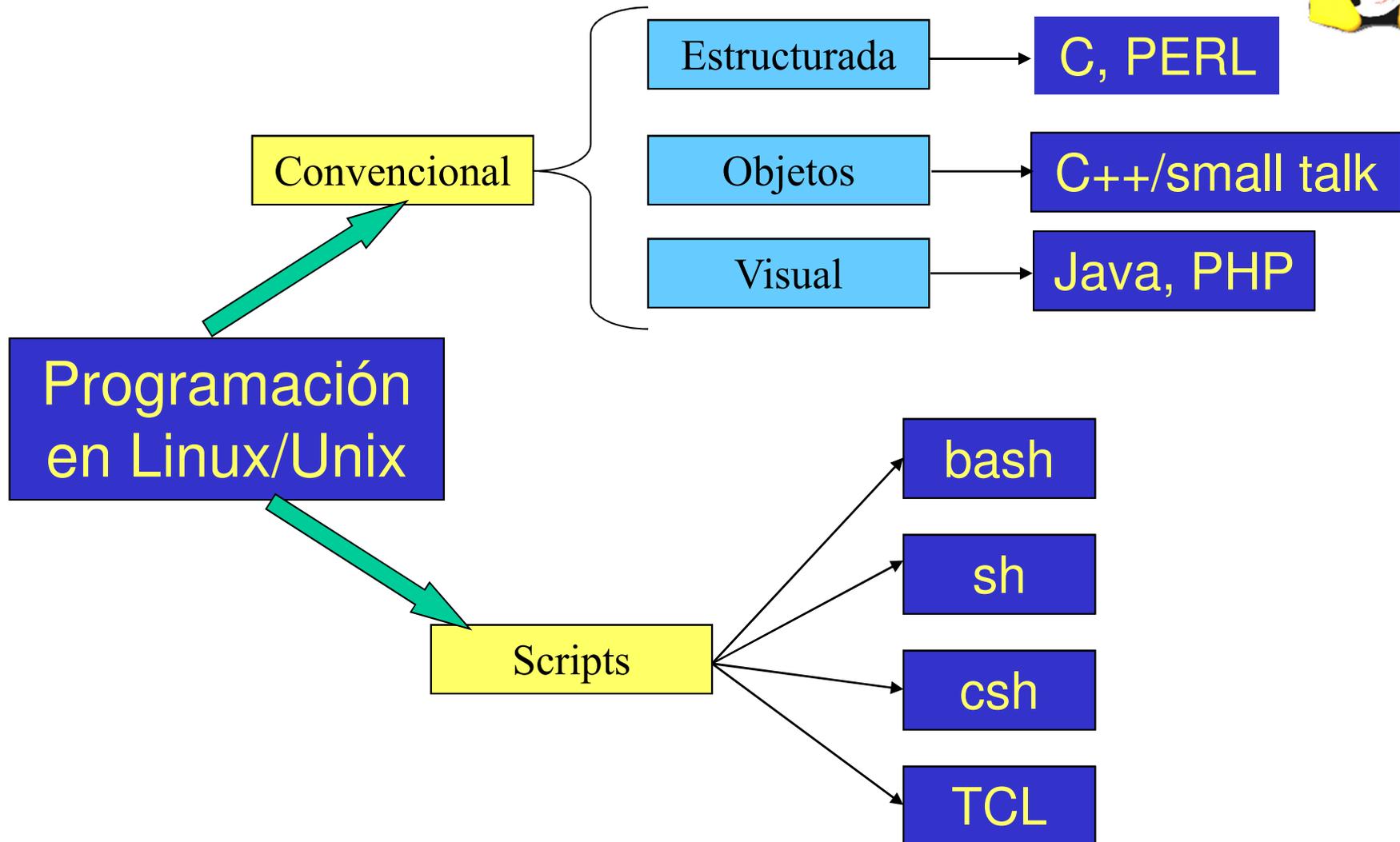
Argumento	Descripción
-l	Lista a crontab propio
-e	Edita crontab. Usa el vi.
-ed	Borra un crontab de usuario
-u usuario	Un file crontab particular de usuario. Solo para el /

```
M H D m d cmd
```

```
30 7 * * 01 sort /usr/wwr/sales/week | mail -s "Fin de"
```

```
30 01 * * * tar cvfz /dev/fd0 /home > home index
```

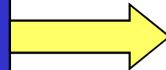
PROGRAMACION DE SHELL



PROGRAMACION DE SHELL

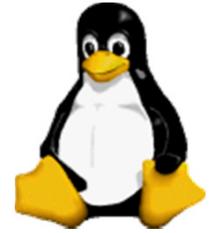


**Bash (Bourne
again shell)**

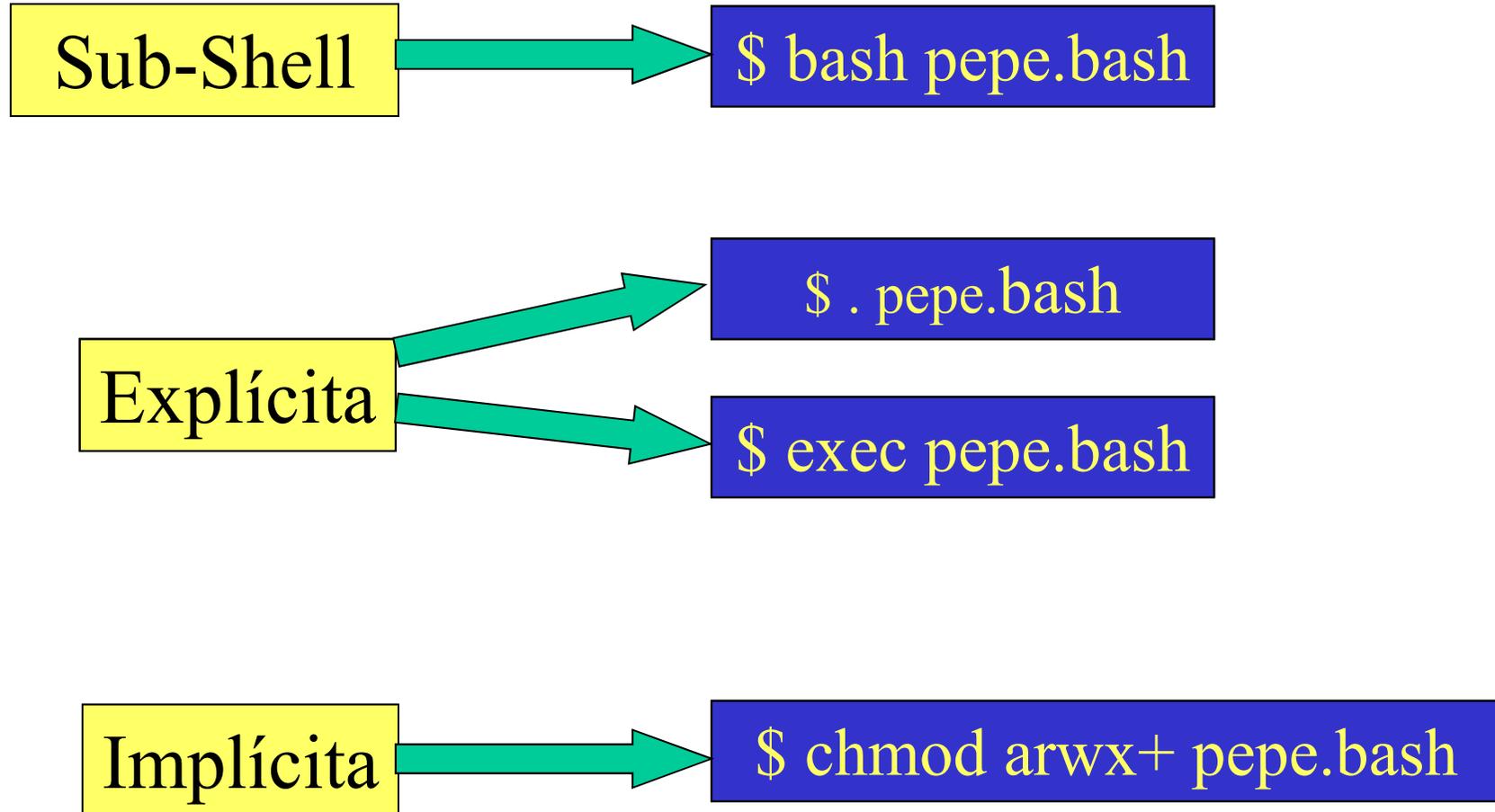


- (1) Es Interpretado. Sintaxis propia.**
- (2) Programas cortos, de texto.**
- (3) Uso principal en la Automatización de tareas de administración de sistemas.**
- (4) Programas top-down. Uso de lógica de C/C++**
- (5) Manejo de Variables tipo string.**
- (6) Estructuras de Control.**
- (7) Set de funciones para la administración de sistemas.**
- (8) Permite el lanzamiento de ejecuciones de programas binarios.**
- (9) Manejo de Menús y Submenús.**
- (10) El código fuente principal no permite el uso de operaciones matemáticas.**

PROGRAMACION DE SHELL



Modos de Ejecución:



PROGRAMACION DE SHELL



Asignación de Valores a variables:

Explícita



```
$ S="Que Tal?"  
$ echo $S  
$ Que Tal?
```

Uso de <read>



```
# Otro ejemplo  
echo "Su nombre : \c"  
read NOMBRE  
echo "Domicilio: \c"  
echo "$NOMBRE -$DOMICILIO"  
$ Que Tal?
```

PROGRAMACION DE SHELL



Primer Script:

```
# ! /bin/bash  
# Ejemplo 1: Hola Mundo!  
echo "Hola Mundo!!!!!!!!!"
```

Paso de Variables:

```
edad=30  
echo $edad #(visualización de la variable por monitor)  
echo" La edad del usuario que tipea es $edad años #(más completa)  
$ La edad del usuario es 28 años #(salida por monitor)
```

PROGRAMACION DE SHELL



Paso de Parámetros posicionales (argumentos):

```
# ! /bin/bash  
# File de Argumentos  
# Ejemplo 2: Paso de argumentos desde línea de comandos  
# Modalidad: ./arg [name] [surname]  
echo "Bienvenido al Prg $0"  
echo "Su nombre es $1"  
echo "Su apellido es $2"  
echo "Colocando todo a continuación es: $*"
```

```
$ ./args Pepe Garcia  
Bienvenido a ./args  
Su nombre es Pepe  
Su apellido es Garcia  
Colocando todo a continuacion: Pepe Garcia
```

PROGRAMACION DE SHELL



Variables especiales internas:

Variables	Comentario
#	Nº de Parámetros en la línea de llamada al Prg.
*	Todos los parámetros posicionales de la línea de orden separados por blancos
?	Valor de retorno de la última orden ejecutada
\$	Nº de Proceso PID del propio guión
!	Nº de Proceso PID de la última orden ejecutada en background

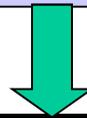
PROGRAMACION DE SHELL



Evaluación de Expresiones:

\$ [expresion]

#Referido a expr. matemáticas



Operación	Comentario
- +	Resta Suma
!	Negación
* / %	Multiplicación División Resto
<< >>	Movimiento de Bits
<= >= <>	Comparaciones
== !=	Igualdad Desigualdad
& ^ &&	AND OR AND Lógico OR Lógico

PROGRAMACION DE SHELL



Operadores para evaluación:

Operador	Comentario
-d	Si archivo es Directorio
-f	Si archivo es archivo común
-r	Si tiene permiso de ejecución activado
-s	Si el largo del archivo es mayor que 0
-w	Si el permiso escritura está activado
-x	Si el permiso de ejecución esta activado

Uso de test (evaluación de sentencias lógicas):

```
test expresion [expresion]
```

PROGRAMACION DE SHELL



Operadores:

export



\$ export <Variables>

readonly



\$ readonly <Variables>

unset



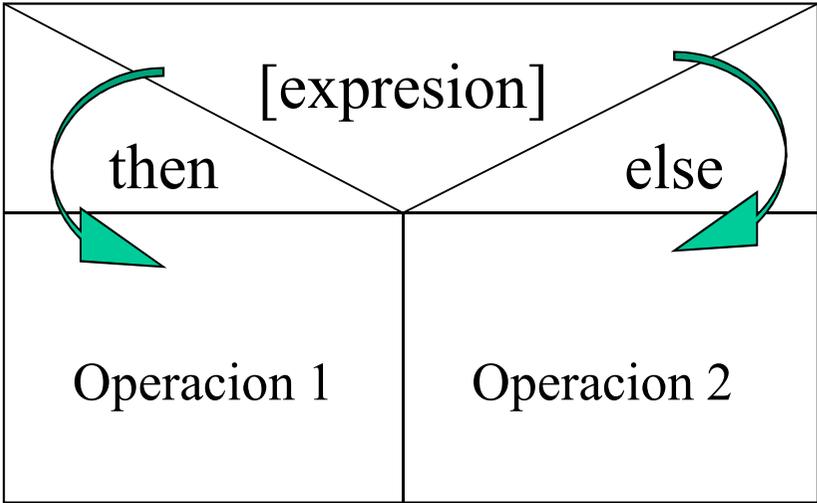
\$ unset Variable



PROGRAMACION DE SHELL

Estructuras Condicionales:

```
if [expresion]
then
Operacion 1
else
Operacion 2
fi
```



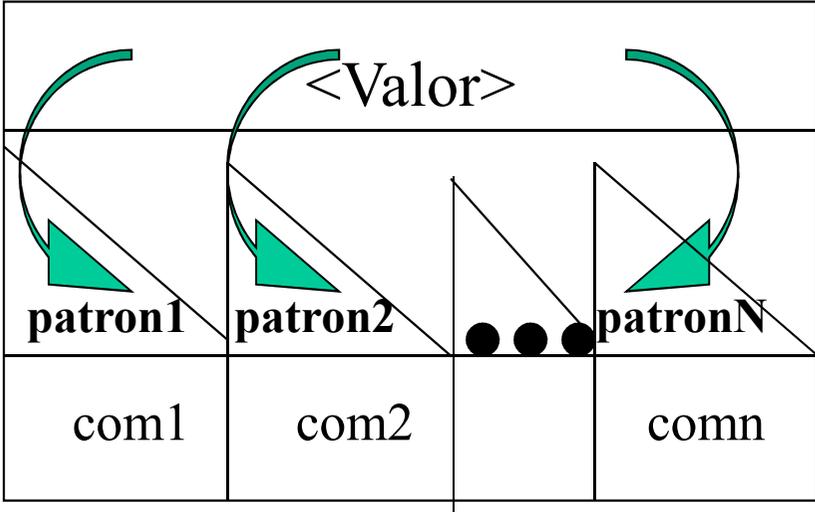
```
...
if [ $valorx = 20 ]
then
echo "Valor es igual a $valorx"
else
echo "Valor es distinto de " $valorx"
fi
...
```



PROGRAMACION DE SHELL

Estructuras Condicionales Múltiples:

```
case <valor> in
    patron1: com1;;
    patron2: com2;;
    patronn:comn;;
esac
```



```
char=$1
case $char in
    [0-9]: echo "Es un dígito";;
    [a-z]: echo "Es una minúscula";;
    [A-Z]: echo "Es una mayúscula";;
esac
```



PROGRAMACION DE SHELL

Estructuras while –do- done:

```
while <comando>
```

```
do
```

```
com1
```

```
com2
```

```
comn
```

```
done
```

```
while <comando>
```

```
com1
```

```
com2
```

```
comn
```

PROGRAMACION DE SHELL



```
while ["opt" != 4]
do
echo "[1] Listar archivos"
echo "[2] Ver directorio de trabajo"
echo "[3] Crear directorio"
echo "[4] salir"
read -p "Ingrese una opcion:" opt
case $opt in
    1) ls;;
    2) pwd;;
    3) read -p "Nombre del Directorio:" DIR
       mkdir $DIR;;
    4) break;;
    *) echo "Opcion invalida, intente de nuevo";;
esac
done
```

PROGRAMACION DE SHELL



Estructuras until-do-done:

```
until <comando>
```

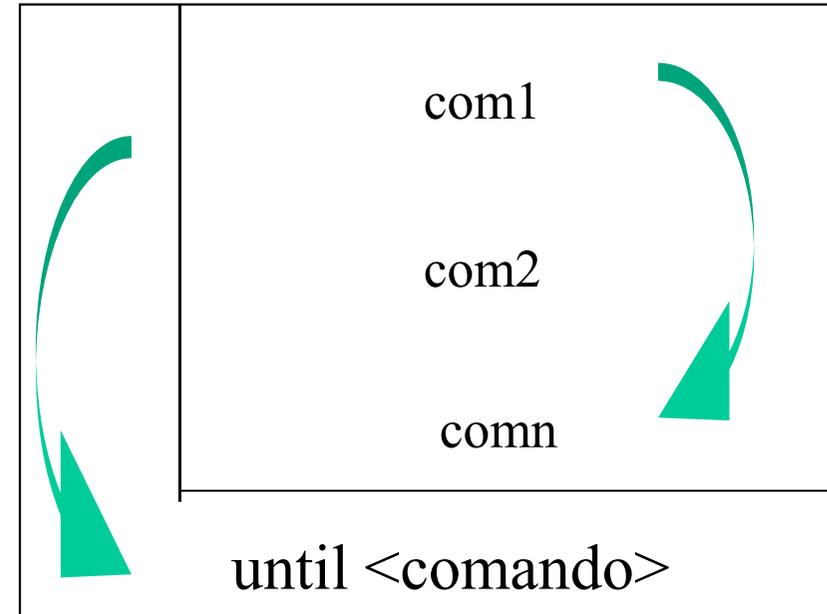
```
do
```

```
com1
```

```
com2
```

```
comn
```

```
done
```



```
usuario=$1
```

```
until who | grep “^$usuario”> /dev/null
```

```
do
```

```
sleep 60
```

```
done
```

```
echo “$usuario esta en el sistema”
```

PROGRAMACION DE SHELL



Estructura for-do-done:

```
For <var> [in <lista>]
```

```
do
```

```
com1
```

```
com2
```

```
comn
```

```
done
```

```
for <var> [in <lista>]
```

```
com1
```

```
com2
```

```
comn
```

```
for x in dosls a: | tr [A-Z] [a-z]
```

```
do
```

```
doscp -m a: $x #copia all files del floppy low density de DOS al dir actual de Unix
```

```
# Transforma los nombres a minúscula
```

```
echo $x
```

```
done
```

BIBLIOGRAFIA DE REFERENCIA



1. Programación en Linux, con ejemplos. Kurt Wall. QUE, Prentice Hall. Madrid. 2000.
2. Sistemas Operativos. 5ta Ed. William Stalling. Pearson Prentice Hall. Madrid. 2006
3. Sistemas Operativos. 7ma Ed. William Stalling. Pearson Prentice Hall. Madrid. 2012
4. Sistemas Operativos Modernos. Andrew. S. Tanenbaum. Prentice-Hall. Interamericana S.A. Madrid, 2009.
5. Unix, Sistema V Versión 4. Rosen, Rozinsky y Farber. McGraw Hill. NY 2000.
6. Linux, Edición especial. Jack Tackett, David Guntery Lance Brown. Ed. Prentice Hall. 1998.
7. El Libro de Linux. Syed M. Sarwar, Robert Koretsky y Syed. A. Sarwar. Ed. Addison Wesley. 2007. España.



**FIN DE TPs PARTE B
DE SOp Linux**

