M A N U A L

D E R E F E R E N C I A

T A L L E R B A S I C O

D E

U N I X

INDICE

1. INTRODUCCION AL SISTEMA OPERATIVO UNIX

Un sistema operativo es el encargado de manejar el hardware y el software del computador. Estas tareas se centran entorno de:

- Gestión de archivos.
- Ejecución de programas.
- Recibir ordenes del usuario.

Esta interacción se hace por medio de una interfaz (Ej: Una línea de textos o elección de menús).

Este manual se centrará en el sistema operativo UNIX, este fue desarrollado en 1969 por Ken Thompson de AT&T Bell Laboratories. Thompson creo este sistema considerando un entorno de investigación, un ejemplo de esto es el típico comentario de los programadores: UNIX es un sistema operativo que hace lo que tiene que hacer y bien, no es una niñera (haciendo alusión a un sistema operativo muy popular).

Con el tiempo los laboratorios Bell distribuyeron el sistema UNIX a instituciones académicas, fueron los estudiantes de estas instituciones los que llevaron este sistema operativo a un nivel de trabajo en la industria. Luego muchas instituciones producto de la distribución de los laboratorios Bell, desarrollaron sus propias versiones, la más importante competencia fue la BSD (Berkeley Software Distribution) de la Universidad de California y otras series de versiones como Xenix para computadores de Microsoft. Con tantas versiones fue necesario desarrollar un estándar, estos fueron las versiones de AT&T y la BSD.

El primero crea la UNIX System Laboratories y en 1991 desarrolla un sistema estándar llamado sistema 5 que tenía las características del sistema 5 versión 4, BSD version4.3, SunOs y Xenix. Otras empresas crearon la OSF(Open Software Foundation), proponiendo otro estándar, luego AT&T vendió sus derechos a Novell. En estos momentos la versión UNIX para computadores más reconocida es Linux que se caracteriza por ser free.

Otro de los conceptos que se asocian a UNIX es el de sistema portable. Esto se explica de la siguiente forma, UNIX fue creado en lenguaje ensamblador, este lenguaje depende de la máquina, cada modelo de computador tiene un lenguaje ensamblador propio, esto implica, que los programas escritos en un lenguaje ensamblador que pertenece al modelo de una máquina determinada no se podrán ejecutar en otra máquina. Para solucionar este problema se creo el lenguaje de programación C, este permite escribir programas independientes de la máquina, debido a que este lenguaje accede en forma directa al hardware por medio de ordenes generales de programación, lo que hace posible que los programas escritos en este lenguaje puedan ser ejecutados en distintos modelos de computadoras.

- El Shell de UNIX

El Shell proporciona una interfaz entre el NUCLEO y el USUARIO:

El Shell controla recursos como los periféricos (pantalla, impresora, etc.), además recursos del computador como el procesador, tarjetas (sonido, vídeo, etc.). También controla las utilidades (programas de aplicación) que son los programas utilizados por los usuarios (Word, Excel, juegos, etc.), además controla la forma en la cual se almacena y se organiza la información (archivos).

2. COMENCEMOS CON UNIX

UNIX es un sistema multiusuario, es decir, varios usuarios pueden acceder de manera simultanea al sistema para manejar su información. Para mantener la seguridad en la información de los usuarios, el administrador del sistema tiene que crear una ***cuenta UNIX*** para cada usuario, este da un **loginID** (nombre de usuario) y una **password** para que puedan empezar a trabajar. En sistemas con varios usuarios, la password permite prevenir que alguien utilice el login de otro usuario, pues es secreta, para que una password sea válida debe contener un mínimo de caracteres alfanuméricos, este número dependerá del sistema UNIX, el primero de los cuales debe ser alfabético.

A continuación se muestra como se accede a un sistema UNIX, luego de haberse asignado el login y la password al usuario:

En el ***ejecutar*** de la barra de inicio de windows se digita:

telnet jbozo.inf.ucv.cl

aparecerá el siguiente mensaje:

Red Hat Linux release 6.0 (Hedwig)

Kernel 2.2.5−15 on an i686

login:

A continuación se digita el login y se presiona enter:

Red Hat Linux release 6.0 (Hedwig)

Kernel 2.2.5−15 on an i686

*login:**dlazcano***

y aparecerá el siguiente mensaje*:*

Red Hat Linux release 6.0 (Hedwig)

Kernel 2.2.5−15 on an i686

login:dlazcano

password:

Se digita la password (esta no aparecerá en pantalla), al presionar enter, comenzará la sesión en UNIX con la siguiente pantalla:

Red Hat Linux release 6.0 (Hedwig)

Kernel 2.2.5−15 on an i686

login: dlazcano

Password:

Last login: Sat Jan 15 16:41:11 from dial−tnt1−65.chilesat.net

[dlazcano@jbozo dlazcano]$

[dlazcano@jbozo dlazcano]$*** es la línea de comandos y el signo ***$ es el prompt del sistema.

La linea de comandos se divide en:

[<usuario>@<nombre del host> <directorio>]

A continuación se vera como cambiar el password. Esta operación se realiza con el comando **passwd:**

- Ejecutar ***passwd*** en la línea de comandos:

[dlazcano@jbozo dlazcano]$passwd

2) Digitar la antigua password y presionar enter, aparecerá el siguiente mensaje:

Changing password for dlazcano

(current) UNIX password:

3) Digitar la nueva password y presionar enter, el sistema responde:

New UNIX password:

4) Pide retipear la nueva password:

Retype new UNIX password:

Se digita nuevamente (se recuerda que la password no aparecerá en pantalla) se presiona enter y el sistema responde:

passwd: all authentication tokens updated successfully

cuando fue correctamente retipeado.

Ahora como ejercicio salga de UNIX y vuelva a ingresar, para verificar que su nueva password ha sido validada, para esto salga con cualquiera de los siguientes comandos, estos son:

− ***logout***.

−***Ctrl−D***.

−***exit***.

3. EL EDITOR EMACS

Los editores de texto se suelen considerar como procesadores de texto, es decir, programas interactivos orientados a pantalla utilizados para abrir o crear archivos de texto, para introducir o cambiar texto, para imprimir y guardar su contenido. Los editores usados para el procesamiento de palabras soportan normalmente el movimiento del cursor y están orientados a la visualización por pantalla, algunos editores poseen menús desplegables. Pero también se pueden editar textos con programas no interactivos, como los filtros de UNIX que se analizan en los capítulos posteriores.

UNIX tiene muchos editores como VI, JET, etc. Se estudiará el editor Emacs, este es un editor que esta disponible en forma gratuita. Escrito originalmente por Richard Stallman a finales de la decada de los 70, esta disponible para la mayoría de las plataformas. Es extremadamente ampliable y tiene su propio lenguaje de programación, se dice que es el editor más potente de UNIX, su nombre proviene de Editor de Macros.

- Comencemos con Emacs

Considerar las siguientes abreviaturas: **C** y **M** corresponden a teclas **Control (Ctrl)** y **ESC (Esc)** respectivamente, el caracter − se considera un carácter de enlace de teclas es decir se interpreta como en el siguiente ejemplo:

C−x se ejecutara como presionar control junto con la tecla **x**.

a) Ejecutemos emacs:

Al iniciar con ***emacs***, se ejecutará el editor de la siguiente manera:

Primero tipear en la línea de comandos:

[dlazcano@jbozo dlazcano]$ ***emacs***

aparecerá la pantalla de recibimiento, presionar enter y aparecerá el siguiente mensaje:

This buffer is for notes you don't want to save, and for Lisp evaluation.

If you want to create a file, visit that file with C−x C−f,

then enter the text in that file's own buffer.

Situarse en la primera línea y borrar el mensaje con C−k (comando para borrar líneas), en este instante se

puede comenzar a escribir.

La segunda forma de ejecutar emacs es con un argumento:

[dlazcano@jbozo dlazcano]$ emacs nombre_archivo

al ejecutar emacs así, se esta creando un archivo en el caso que no exista y si existe lo estará editando.

b) Para terminar la ejecución de emacs:

en forma permanente ejecutar el comando:

C−x C−c

Si los archivos en los cuales se estaba trabajando no han sido grabados, preguntará si desea grabarlos.

Pero también se puede suspender emacs con:

C−z

Y luego volver con el comando ***fg***, la explicación de cómo ejecutar estos comandos se estudiara más adelante.

c) Manejo de archivos y ventanas en emacs:

c.1) A continuación ejecutar emacs, para esto digitar lo siguiente en la línea de comandos:

[dlazcano@jbozo dlazcano]$ emacs

aplicar lo aprendido en la letra a) referente a la ejecución de emacs con este formato. Ahora digitar el siguiente texto:

HOLA MUNDO EMACS

c.2) Guardar el archivo con **C−x C−s** (guardar), aparecerá el siguiente mensaje:

File to save in: ~/

Digitar el nombre del archivo:

File to save in: ~/prueba1

Luego presionar enter y el sistema responde:

Wrote /home/dlazcano/prueba1

Este mensaje confirma el lugar donde quedará guardado, guardar en el directorio ***dlazcano,*** más adelante se estudiará el concepto de árbol de directorios con el cual se podra visualizar y entender mejor este mensaje.

c.3) Crear otro archivo, para esto no será necesario ejecutar otro emacs, se dividirá la pantalla en dos con el siguiente comando: ***C−x 2***

ahora que ya esta dividida la pantalla en dos crear otro archivo en la otra pantalla, para esto trasladarse con el

siguiente comando:

C−x o

Para crear el otro archivo utilizar la siguiente opción de apertura:

C−x C−f

Aparecerá el siguiente mensaje:

Find file: ~/

A continuacion se da nombre al archivo:

Find file: ~/prueba2

El siguiente mensaje indica que se ha creado un archivo nuevo:

(New file)

Este comando se utiliza para abrir archivos, pero su acción por defecto es que si no existe lo crea, a continuación digitar:

HOLA MUNDO UNIX

Guardar el archivo con ***C−x C−s***, como el archivo ya ha sido nombrado solamente guardara lo que se ha escrito en el archivo ***prueba2***, lo cual se vera en el siguiente mensaje:

Wrote /home /dlazcano/prueba2

Para cerrar la ventana que se abrió utilizar el comando ***C−x 0,*** recordar que la ventana que se cerrará es en la que se esta situado**.**

c.4) A continuación se analizará los dos comandos antes mencionados para salir de emacs, primero se analizará el comando: ***C−z***.

Este comando suspende emacs, el sistema responde con el siguiente mensaje:

[1]+ Stopped emacs

[dlazcano@jbozo dlazcano]$

la línea ***[1]+ Stopped emacs*** indica que hay un proceso emacs suspendido, esta opción da la versatilidad de salir del editor hacia la línea de comandos UNIX y luego volver al emacs, para volver al proceso suspendido digitar en la línea de comandos el mandato ***fg*** *(mandato UNIX)*:

[dlazcano@jbozo dlazcano]$fg

Si no se quiere suspender el emacs que se está ejecutando, el mandato a utilizar será:

C−x C−c

Con este mandato se saldrá de emacs y se termina su ejecución, siempre y cuando los archivos hayan sido guardados en caso contrario se mostrara la siguiente pregunta:

Save file /home/ayudantes/dlazcano/prueba2? (y, n, !, ., q, C−r or C−h)

Para ver cada una de las alternativas digitar ***C−h***, para grabar digitar ***y***.

c.5) A continuación se procederá a abrir emacs, considerando que ya se tienen dos archivo creados ***prueba1*** y ***prueba2***.

Como ya sabe podrá hacerlo de dos formas:

a)El caso visto anteriormente:

[dlazcano@jbozo dlazcano]$ emacs

luego, dentro de emacs abrir el archivo, con el comando:

C−x C−f

Al ejecutarlo responderá emacs con el siguiente mensaje:

Find file: ~/

Digitar el nombre del archivo a abrir:

Find file: ~/prueba1

Presionar enter y el archivo será abierto.

b) Otra forma de acceder a un archivo, en este caso ***prueba1***, es dando un argumento como se muestra a continuación:

[dlazcano@jbozo dlazcano]$ emacs prueba1

hay que considerar que con este formato si el archivo no existe, será creado.

c.6) Una vez abierto el archivo, dividir nuevamente la pantalla con el comando: ***C−x 2*** y abrir el archivo **prueba1** en una ventana y el archivo **prueba2** en la otra ***(ver punto c.3).***

A continuación volver a dividir la ventana pero de forma vertical con el comando: ***C−x 3***

Primero situarse en la ventana de ***prueba1*** y ejecutar él comando ***C−x 3***, luego situarse en la ventana de ***prueba2*** y repetir la operación, recordar que para moverse por las pantallas se utiliza el comando ***C−x o***.

La pantalla quedara dividida de la siguiente forma:

A continuación crear dos archivos más dejando solo una ventana de ***prueba1*** y una ventana de ***prueba2***, tomando en cuenta los pasos seguidos en el punto c.3) referente al comando ***C−x C−f*** y al comando ***C−x C−s***, crear los archivos ***prueba3,*** en el digitar:

esta es la prueba 3.

En el archivo ***prueba4*** digitar:

esta es la prueba4.

La pantalla quedara de la siguiente forma:

c.7) A continuación incluir todo el contenido del archivo ***prueba3*** en el archivo ***prueba4***, para esto utilizar el comando: ***C−x i***.

a) Primero ubicarse sobre la pantalla que corresponde al archivo **prueba4**, ejecutar el comando y aparecerá el siguiente mensaje:

Insert file: ~/prueba3

Con esta sentencia se esta indicando que archivo ***prueba3*** sea el insertado en el archivo ***prueba4***.

Este archivo se guarda con otro nombre para esto ejecutar el comando:

C−x C−w

Emacs dará el siguiente mensaje:

Write file: ~/

Digitar el nombre:

Write file: ~/prueba5

Y el nuevo archivo ha sido creado.

c.8) Para terminar usar el comando: ***C−x C−v***.

Este comando sirve para sustituir el archivo actual con otro.

En el ejemplo se remplazará el archivo ***prueba5*** por el archivo ***prueba4***, ejecutar el comando y el mensaje que se verá en pantalla será el siguiente:

Find alternate file: ~/prueba5

Borrar el nombre de ***prueba5*** y digitar ***prueba4***:

Find alternate file: ~/prueba4

Presionar enter y se produce el remplazo.

d) El sistema de ayuda

Para obtener la ayuda de comandos que contienen un string se ejecuta el comando: *C−h*

Aparecerá el siguiente mensaje:

C−h (Type ? for further options)−

Para poder conocer las opciones que tiene el sistema de ayuda teclear: ***?***.

C−h (Type ? for further options)?

En pantalla se desplegaran las opciones que existen, dentro de estas opciones existe la alternativa ***a***, esta opción permite hacer una búsqueda de un string determinado, considere el siguiente ejemplo:

Digitar :

C−h (Type ? for further options)a

Y aparecerá el siguiente mensaje:

Apropos command (regexp):

Digitar **delete** que será el concepto de busqueda:

Apropos command (regexp): delete

Y aparecerá el resultado de esta búsqueda.

Para cerrar la ventana de ayuda ejecutar el comando: C−x 1, estando en la ventana de archivo de trabajo.

e) Tratamiento de errores y ejecución de comandos shell desde emacs

Uno de los errores mas complicados es cuando se elimina un archivo por equivocación, emacs da la posibilidad de recuperar dicho archivo mediante el comando: ***M− x recover−file***

Ejemplo del uso de este comando:

e.1) Primero se hará el alcance de que este comando podrá recuperar el archivo siempre y cuando la eliminación se haya hecho desde dentro de emacs, por lo tanto primero se aprenderá a ejecutar comandos de shell desde emacs.

Para esto existe el comando: ***M−!***.

El ejemplo que se dará será el de la eliminación de un archivo, se ejecutará el comando y aparecerá el siguiente mensaje:

Shell command:

Digitar el comando ***rm*** (se estudiará mas adelante), y el nombre del archivo a eliminar:

Shell command:rm prueba1

Al presionar enter emacs confirmá la buena ejecución del comando con el siguiente mensaje: ***(Shell command succeeded with no output)***

luego ejecutar el comando: ***M x recover−file.***

Emacs responderá con el siguiente mensaje:

Recover file: ~/

Digitar el nombre del archivo a recuperar:

Recover file: ~/prueba1

Emacs responderá con el siguiente mensaje:

Recover auto save file /home/ayudantes/dlazcano/#pruebasegunda#? (yes or no)

Digitar :***yes*** y el archivo es recuperado.

e.2) Comando deshacer:

A continuación se dará un ejemplo para deshacer el borrado de un carácter.

Para esto ejecutar el comando ***C−x u***.

Abrir el archivo ***prueba2***, y al texto:

hola mundo UNIX

*b*orrar la letra X:

hola mundo uni

al ejecutar C−x u volverá a aparecer la letra X, este deshacer funciona como un retroceder, y se retrocederá tantas veces se ejecute el comando.

f) Búsquedas y movimientos en el texto

A continuación se muestra la lista de comandos de:

a) búsquedas en el texto:

C−s	***buscar hacia delante***
C−r	***buscar hacia atrás***
ESC	***acabar la búsqueda incremental***
C−g	***abortar la búsqueda actual***

Usar ***C−s*** o ***C−r*** otra vez para repetir la búsqueda en cualquier dirección.

Si Emacs sigue buscando, ***C−g*** cancelara la parte no hecha de la búsqueda,

de otro modo la aborta toda.

b) Movimiento del cursor en el texto:

Sobre que moverse	Atrás	Adelante
Carácter	C−b	C−f
Palabra	Mb	Mf

Linea	C−p	C−n
Ir a principio de la línea	C−a	C−e
Frase	Ma	Me
Párrafo	M−[	M−]
Pagina	C−x[	C−x]
Ir a principio de archivo	M−<	M−>

Considerando el texto escrito en prueba2:

hola mundo UNIX

Se analizarán los siguientes comandos:

Situarse al principio del archivo con ***M−<.,*** luego con ***C−s*** buscar la palabra ***mundo***. Al ejecutar el comando de búsqueda aparece el siguiente mensaje:

I−search:

Presionar enter y aparece lo siguiente:

Search:

Digitar la palabra a buscar:

Search:mundo

La forma de mostrar que la encontró es ubicar el cursor al final de la palabra.

g) Marcar, suprimir y borrar texto:

A continuación se dará la lista de comandos de:

g.1) Marcar texto:

Para marcar una región de texto (para copiar o borrar, por ejemplo) poner una marca al final de la región de texto a marcar y situarse al principio .

El texto comprendido entre el lugar de la marca y el cursor es la región.

Poner marca aquí	C−SPC
Marcar párrafo	M h
Marcar pagina	C−x C−p
Marcar todo el buffer	C−x h
Intercambiar marca y cursor	C−x C−x

g.2) Suprimir y borrar texto:

Objeto a borra	Atrás	Adelante
Carácter(borrar no suprimir)	Del	C−d

Palabra	Del	Md
Línea	C−k	C−k
Frase	C−x del	Mk

Si previamente se ha marcado una región esta se podrá suprimir. El texto suprimido se guarda para poder copiarlo en otro lugar. Es como un copy + cut

Suprimir región: *C−w*

g.3) Copiar y recuperar

Seleccionar	**C−c SPACE (select)**
Borrar	**C−w (cut)**
Copiar la ultima región borrada	**C−y(copy)**
Recuperar regiones borradas anteriormente	**My (paste)**
Deshacer	**C−x u (undo)**

A continuación considerando el archivo *prueba2* se realiza una muestra de cómo funcionan estos comandos:

HOLA MUNDO UNIX

Nota : considerar como el cursor al símbolo @:

Primer ejercicio:

El primer ejercicio que se realizara será un cortar y pegar para esto situarse con el cursor al final de la palabra que se va a cortar.

HOLA MUNDO UNIX@

Se coloca una marca con el comando ***C−spc*** y luego se situa al principio de la palabra que se va a cortar***:***

HOLA MUNDO @UNIX

A continuación se corta con el comando ***C−w***:

HOLA MUNDO@

Y se traslada el cursor al principio del lugar donde se quiere pegar:

@HOLA MUNDO

Finalmente pegar con el comando ***C−y***:

@UNIX HOLA MUNDO

Segundo ejercicio:

El segundo ejercicio a realizar será un copiar y pegar un párrafo, para esto se situara al principio del párrafo, luego se marcara con el comando ***M−h***:

@UNIX HOLA MUNDO

luego que se marcó el párrafo, se ubica el cursor en el lugar donde queremos hacer la copia y se ejecuta el comando ***C−y***, se pueden hacer tantas copias como veces se ejecute el comando.

Tercer ejercicio:

A continuación se realizara el mismo ejercicio anterior, pero se copiara el párrafo en otro archivo, para esto recordar el punto ***c.2)*** que permite abrir dos ventanas en pantalla y abrir otro archivo en la pantalla abierta, este archivo será ***prueba2***, repetir los pasos de marcado del punto anterior, trasladarse hacia la otra pantalla (de ***prueba2***) y realizar la copia con el comando ***C−y***, para deshacer cualquier copia ejecutar el comando ***C−x u.***

h) Substituir string del texto

Para esto se utiliza el comando interactivo: ***M%***

Al remplazar string este comando permite remplazar símbolos letras frases, etc.

Al ser un comando interactivo existen respuestas válidas en modo sustitución

estas son:

Substituye este, vas al siguiente	SPC
Substituye este, quédate aquí	,
Saltar al siguiente sin sustituir	DEL
Substituir todas las ocurrencias siguientes	!
Retroceder a la ocurrencia anterior	^
Acabar la substitución	ESC

A continuación se dará un ejemplo de remplazo para esto se tomará del punto g) el segundo ejercicio, se harán tres copias del texto escrito en el archivo *prueba2*:

HOLA MUNDO UNIX

Por lo tanto antes de empezar el ejercicio se tendrá en pantalla lo siguiente***:***

HOLA MUNDO UNIX

HOLA MUNDO UNIX

HOLA MUNDO UNIX

A continuación remplazar la palabra ***mundo*** por ***universo***, primero situarse al principio del archivo con el comando ***M−<*** para que lo recorra completo, al ejecutar el comando se muestra el siguiente comando:

Query replace:

Se pide la palabra que se quiere remplazar, digitar:

Query replace: MUNDO

El mensaje luego de presionar enter es el siguiente:

Query replace MUNDO with:

Solicita la palabra por la cual queremos remplazar ***mundo***:

Query replace MUNDO with:UNIVERSO

El sistemaresponde con el siguiente mensaje:

Query replacing mundo with UNIVERSO: (? for help)

En este momento se tiene que ingresar las opciones de la lista anteriormente mostrada, en este caso se elegira ***SPC***, que permite sustituir donde se ubica el cursor y luego ir a la siguiente, al presionar sucederá lo siguiente:

HOLA UNIVERSO UNIX

HOLA MUNDO UNIX

HOLA MUNDO UNIX

volverá a preguntar:

Query replacing mundo with UNIVERSO: (? for help)

Ahora elegir el comando ***!*** que permitirá substituir todas las ocurrencias siguientes, al terminar aparecerá un mensaje que indica la cantidad de ocurrencias encontradas:

Replaced 3 occurrences

i) Ejecución de comandos shell desde emacs:

Se pueden ejecutar comandos de shell desde dentro de emacs para esto ejecutar el comando: ***M−!***

Considere como ejemplo la ejecución del comando ***ls***, este se estudiara mas adelante, al ejecutar el comando ***M−!*** vera el siguiente mensaje:

Shell command:

Digitar el comando:

Shell command:ls

Y se dividirá la pantalla en una de ellas se mostrara el resultado del comando ***ls***.

Pero también se puede iniciar una shell en una ventana shell dentro de emacs para esto ejecutar el comando: ***M−x shell***

Al igual que con el comando anterior la pantalla se dividirá en dos, y se podrá desplazar por las pantallas con el comando ***C−x o***, esta ventana de shell se ejecuta normalmente como si se estuviéra en la línea de comandos UNIX.

4 ARCHIVOS EN UNIX<!DOCTYPE HTML SYSTEM "legacy.dtd">

Un archivo es una colección de información que se almacena en un disco o cinta magnética.

Los archivos del sistema son estructuras que los ordenadores utilizan para organizar y almacenar información. Existen 2 tipos de archivos del sistema:

- **Archivos ordinarios:** estos archivos contienen datos, textos y/o programas ejecutables (comandos).
- **Archivos directorios:** estos archivos contienen nombres de archivos. Los directorios no se utilizan para almacenar datos, si no que se utilizan para organizar otros archivos en grupos.

a) Nombres de archivos

Todos los archivos tienen asociado un "nombre de archivo" ; este nombre identifica el archivo y su contenido.

El nombre de un archivo puede tener de 1 a 255 caracteres; pero se pueden utilizar únicamente los siguientes caracteres:

- Letras mayúsculas (A − Z).
- Letras minúsculas (a − z).
- Números (0 − 9).
- Subrayado (_).
- Punto (.).
- Coma (,).

b) Extensión de un archivo

Las extensiones de archivos proporcionan una ayuda para clarificar el contenido del archivo. Van precedidas por un punto (.) en el nombre del archivo.

UNIX utiliza algunas extensiones para realizar ciertas operaciones; por ejemplo, para compilar un programa FORTRAN, el nombre del archivo que contiene las sentencias Fortran, debe tener una determinada extensión (.f). Pero en la mayoría de los casos las extensiones de los archivos son opcionales. Ej:

− Hola.txt

− Program.c

c) Metacaracteres

Son símbolos especiales que realizan alguna operación en algunos comandos. Estos metacaracteres son:

- **Asteriscos (*)** : Se sustituye por cualquier secuencia de caracteres.
- **Interrogación (?)** : Se sustituye por cualquier carácter.
- **Corchetes ([])** :Se utilizan para especificar una lista de caracteres o un rango. Cuando se utilizan para especificar un rango hay que poner el signo − separando el primer carácter y el último del rango.

En la siguiente figura aparecen ejemplos de cómo se utilizan los metacaracteres.

[dlazcano@jbozo dlazcano]$ls t*

test1 test1.dato test2 test3

[dlazcano@jbozo dlazcano]$ls test?

test1 test2 test3

[dlazcano@jbozo dlazcano]$ ls *[13]

states1 test1 test3

d)archivos invisibles

Los archivos cuyos nombres comienzan con un punto (.) se llaman "archivos invisibles" porque normalmente no aparecen cuando se pide el listado de un directorio.

Normalmente los archivos invisibles se utilizan para almacenar información que el sistema utiliza automáticamente.

e)comando *ls*

Lista el contenido de un directorio o archivo. Si ponemos sólo l***s*** se obtiene una lista con el nombre de los archivos; si se quiere obtener más información sobre esos archivos se utilizan las opciones del comando, cuya sintaxis general es:

ls [−alsF] *archivo*

−a	Lista además los archivos invisibles (es decir, los que empiezan por punto)
−l	Dá la siguiente información de los archivos : • TIPO DE ARCHIVO : d directorio − archivo ordinario • TIPO DE PERMISOS : r lectura w escritura x ejecución • NUMERO DE ENLACES • NOMBRE DEL PROPIETARIO DEL ARCHIVO • TAMAÑO DEL ARCHIVO (en bytes) • FECHA DE LA ULTIMA MODIFICACION • NOMBRE DEL ARCHIVO NOTA.− ls −l es equivalente a ll

−s	El tamaño del archivo en kilobytes (1024 bytes) precede al nombre de cada archivo.
−F	Añade un / a los archivos directorios y un * a los archivos ordinarios ejecutables. NOTA.− ls −F es equivalente a lf
archivo	Nombre de un archivo o un directorio.

Estas opciones se pueden combinar para obtener la información que queramos al mismo tiempo; por ejemplo, ***ls −sF***, dará la lista de los archivos en la que el nombre de cada archivo va precedido por su tamaño (en kilobytes) y va seguido de un slash (/) en el caso de que sea un directorio o de un asterisco (*) en el caso de que sea un archivo ejecutable.

5 MANEJO DE ARCHIVOS EN UNIX

a) edición de archivos

Los comandos que se utilizan para ver un archivo son: **cat** y **more**.

*comando **cat**:*

El comando **cat** (*concatenate*) se utiliza para visualizar por pantalla el contenido de uno o más archivos. Cuando se especifica más de un archivo, cat los edita uno detrás de otro. La sintaxis del comando es:

cat [−ns] *archivo(s)*

−n	Numera las líneas.
−s	Elimina las líneas en blanco.
archivo(s)	Nombre o nombres de los archivos que se van a editar.

El comando cat no pagina, es decir no controla la cantidad de lina que van apareciendo por pantalla, entonces se utiliza:

- ***CTRL−S*** parar la pantalla.
- ***CTRL−Q*** para continuar con la edición.

El comando cat permite también concatenar archivos; para ello se pondría:

cat archivo1 archivo2 ... archivo n

entonces une los archivos archivo1 archivo2 ... y lo almacena en el archivo n.

NOTA

Si se utiliza por equivocación el comando cat sin ningún argumento, intenta leer de la pantalla, por lo que no sale el prompt del sistema (se queda como colgada); entonces hay que poner:

CTRL−C

por lo que debemos de tener cuidado con esto.

Como el comando cat no pagina, cuando queramos ver un archivo que es muy largo, es aconsejable utilizar el comando ***more***.

*comando **more**:*

El comando **more** se utiliza para ver archivos por la pantalla; la principal diferencia con cat es que se puede controlar el número de líneas que aparecen en pantalla, utilizando las teclas siguientes:

- Con la "***barra espaciadora***" se avanza una página.
- Con la tecla de ***return*** se avanza una línea.
- Con la tecla DEL ó q se sale de la edición.

La sintaxis de este comando es:

more [−cd] [+número de líneas] [+/path] *archivo(s)*

−c	Edita pantalla a pantalla.
−d	Número de líneas que se van a editar.
+número de líneas	Número de la línea a partir de la cual se va a editar.
+/path	Path correspondiente al archivo que se va a editar.
archivo(s)	Nombre o nombres de los archivos que se van a editar.

Por ejemplo:

[dlazcano@jbozo dlazcano]$**more** −c1 +2 datos

editará 1 líneas, empezando por la 2, del archivo llamado datos.

b)Comandos para manipular archivos

Se pueden manipular archivos creándolos, borrándolos, cambiando su nombre o cambiando su contenido.

Los comandos que se utilizan para manipular archivos son: **cp**, **rm** y **mv**.

comando ***cp*** :

El comando **cp** se utiliza para copiar archivos. Su sintaxis es:

cp [−i] *archivo entrada archivo destino*

−i	Origina que el comando requiera una confirmación, en el caso de que el archivo destino ya exista; es decir, pregunta si se desea hacer la copia.
archivo entrada	Nombre del archivo que se va a copiar.
archivo destino	Nombre del archivo en el que se va a copiar el contenido del archivo de entrada

En el caso de que el archivo destino ya exista, lo sobre−escribe, esto significa que se escribirá la información que se esta copiando sobre la información antigua perdiéndose esta última, por lo que es recomendable utilizar la opción −i para que pida la confirmación y así evitar posibles errores. Por ejemplo:

[dlazcano@jbozo dlazcano]$ cp −i datos datos.new

comando **rm**:

El comando rm se utiliza para borrar archivos. La sintaxis de este comando es:

rm [−i] *archivo(s)*

−i	Origina que el comando requiera confirmación para ejecutarse.
archivo(s)	Nombre o nombres de los archivos que se van a borrar.

La opción −i se debe de utilizar para pedir confirmación antes de proceder al borrado.

comando **mv**:

El comando mv se utiliza para renombrar archivos; es decir, el contenido del archivo no cambia, sólo cambia el nombre; o para mover archivos entre directorios (se verá en el capítulo siguiente). La sintaxis del comando es:

mv [−i] *archivo entrada archivo destino*

−i	Origina que el comando requiera una confirmación, en el caso de que el archivo destino ya exista; es decir, pregunta si se desea hacer la copia.
archivo entrada	Nombre del archivo que se va a copiar.
archivo destino	Nombre del archivo en el que se va a copiar el contenido del archivo de entrada. En el caso de que el archivo destino exista, lo "machaca".

6 MANEJO DE DIRECTORIOS EN UNIX

La estructura del conjunto de todos los directorios del sistema, es una estructura de árbol invertido, como se puede apreciar en la figura siguiente. Un directorio equivale a abrir una rama dentro del árbol.

Observemos que:

- Los directorios pueden contener otros directorios, archivos ordinarios o estar vacíos.
- Un archivo ordinario es siempre el último archivo en un path (camino).
- El primer directorio de la estructura es el directorio raíz; todos los demás archivos y directorios parten de él. El directorio raíz se designa con un nombre especial, /. Ningún otro archivo puede tener este nombre.

En el sistema UNIX, todos los archivos forman parte de la jerarquía. Cualquier archivo de esta estructura es parte de una red de directorios conectados. Esta red de directorios, junto con el nombre de un archivo particular, constituye el **pathname** para un archivo.

Cada archivo se identifica con un único pathname, que describe su localización con respecto a los otros directorios.

Se puede especificar un nombre de archivo utilizando pathnames absolutos o relativos:

- Un *pathname absoluto* especifica la localización de un archivo desde el directorio raíz. Por lo tanto, todos los pathnames absolutos deben de empezar con un slash (/).
- Un *pathname relativo* especifica la localización de un archivo con respecto al directorio en que se está trabajando, en lugar del directorio raíz, por lo que no empiezan con un slash (/).

El punto (.) se refiere al directorio en que se está, y dos puntos (..) se refiere al directorio anterior.

A continuación se muestra un ejemplo del árbol de directorio de UNIX:

El *home directory* es un subdirectorio del directorio raíz en el que se entra cada vez que se hace *login*; es donde van a residir los archivos del usuario. Normalmente tiene el mismo nombre que el nombre de usuario o también se pueden agrupar usuarios bajo un concepto común. Por ejemplo:

home\<***usuario***>

− usuario: daniel

home***daniel***

o tambien podria ser:

home\<***grupo***>

− grupo: ayudantes

home***ayudantes***

a) Crear directorios: comando mkdir

El comando **mkdir** (*make directory*) se utiliza para crear nuevos directorios. Su sintaxis es:

mkdir [path] *directorio*

path	Cuando el directorio no se quiere crear en el que se está, hay que indicar el path de donde se quiere crear.
directorio	Nombre del directorio que se va a crear.

Por ejemplo, si se esta en el directorio ***dlazcan****o* y se quiere crear un directorio de nombre *programa*, entonces se debe digitar:

[dlazcano@jbozo dlazcano]$ **mkdir** programa

b) acceder a un directorio con el comando: cd

El comando **cd** (*change directory*) se utiliza para moverse de un directorio a otro. Su sintaxis es:

cd [directorio] [..] [.] [~] [~nombre usuario]

directorio	Nombre del directorio al que se quiere acceder. Si este directorio no es el inmediatamente siguiente al que nos encontramos hay que indicar el pathname correspondiente, bien sea el absoluto o el relativo.
..	Se refiere al directorio inmediatamente anterior al que nos encontramos.
.	Se refiere al directorio en que estamos.
~	Para ir directamente al **Home directory**.
~nombre usuario	Para ir al **Home directory** del usuario especificado.

Con un mismo comando **cd** se puede avanzar y retroceder en la estructura de árbol, por ejemplo:

[dlazcano@jbozo dlazcano]$* cd *programas

[dlazcano@jbozo programas]$

Considere además un comando muy importante, el comando **pwd** (*print working directory*) dice en qué directorio se esta. El formato es:

[dlazcano@jbozo dlazcano]$pwd

c) Borrar directorios

Para borrar directorios se utilizan los comandos : **rmdir** ó **rm −r**.

comando **rmdir**:

El comando **rmdir** (*remove directory*) se utiliza para borrar un directorio; pero antes de utilizar este comando se deben de borrar todos los archivos que contenga (incluidos los archivos invisibles), es decir, el directorio que se va a borrar tiene que estar vacío.

La sintaxis del comando es:

rmdir opción *directorio*

Opcion	Opciones de eliminación
directorio	Nombre o nombres de los directorios que se van a borrar.

Si el directorio que se va a borra contiene algún archivo, cuando se ejecute el comando rmdir dará un mensaje de error.

comando **rm −r**

En el capítulo anterior se estudio cómo borrar archivos utilizando el comando **rm**, a continuación se vera cómo utilizar este comando para borrar directorios, ya que éstos son un tipo de archivos.

El comando **rm −r** borra recursivamente todos los archivos que existan en el directorio y después borra el directorio. La sintaxis del comando es:

rm −r [−i] *directorio*

−i	Origina que el comando requiera confirmación para borrar cada uno de los archivos contenidos en el directorio.
Directorio	Nombre del directorio que se va a borrar.

Un ejemplo de este comando es:

[dlazcano@jbozo dlazcano]$ rm −r −i aplicaci

d) copiar archivos entre directorios comando: cp

Para copiar archivos entre dos directorios se utiliza el comando **cp** con la siguiente sintaxis:

cp directorio1/archivo1 directorio2/archivo2

directorio1	Nombre del directorio, o path, donde se encuentra el archivo que se va a copiar.
Archivo1	Nombre del archivo que se va a copiar.
Directorio2	Nombre del directorio, o path, donde se va a poner la copia del archivo.
Archivo2	Nombre que se le va a dar a la copia del archivo. Si se omite, la copia tendrá el nombre del archivo original (es decir, archivo1).

Si se omite directorio2, toma por defecto el directorio en el que nos encontramos; entonces la sintaxis de comando será:

[dlazcano@jbozo dlazcano]$ cp directorio1/archivo1 .

e) renombrar archivos de otros directorios comando: mv

Para renombrar un archivo y/o trasladarlo a otro directorio se utiliza el comando **mv**, con formato:

mv directorio1/archivo1 directorio2/archivo2

directorio1	Nombre del directorio, o path, donde se encuentra el archivo que se va a copiar.
Archivo1<=/th	Nombre del archivo que se va a copiar.
Directorio2	Nombre del directorio path, donde se va a poner la copia del archivo.
Archivo2	Nombre que se le va a dar a la copia del archivo. Si se omite, la copia tendrá el nombre del archivo original (es decir, archivo1).

Si se omite directorio2, toma por defecto el directorio en el que nos encontramos; entonces la sintaxis de comando será:

[dlazcano@jbozo dlazcano]$ mv directorio1/archivo1 .

7 EJECUCION EN PRIMER Y SEGUNDO PLANO

Ejecución en primer plano**:** El inicio de un proceso depende del termino de otro.

Ejemplo:

[dlazcano@jbozo dlazcano]$ ***date;who***

/* primero se ejecuta date después who*/

Ejecución en segundo plano: El inicio de un proceso no necesita el termino de otro.

Ejemplo:

[dlazcano@jbozo dlazcano*]$ date & who*

/* who no espera que termine date */

8 CORREO ELECTRONICO

Uno de los programas de correo más sencillo se denomina MAIL.

Este como otros programas funcionan con direcciones de correos electrónicos la sintaxis es la siguiente:

Nombre_de usuario@nombre_sistema.nombre_dominio

Ejemplo:

Dlazcano@jbozo.inf.ucv.cl

A continuación se muestra que papel cumple en el ejemplo cada parte de la dirección de correo electrónico:

Dlazcano = Nombre_usuario

Jbozo = nombre_sistema

Inf.ucv.cl = nombre_dominio

Creación y envío de correo electrónico por medio de mail:

Nota: considerar que se enviara un mail a nuestra cuenta de correo electrónico.

- Introducir la palabra **mail** en la línea de comandos seguida de una dirección de correos , como se muestra a continuación:

[dlazcano@jbozo dlazcano]$mail dlazcano@jbozo.inf.cl

2) El mandato **mail** responderá con el indicador (tema) subject, este dá la referencia del mensaje, introducir una línea sobre el tema:

subject : feliz año 2000

3) Pulsar enter y teclear mensaje:

hola como estas

4) Después de finalizar el tecleo del mensaje se coloca un punto en la línea siguiente o se presiona ***Ctrl−D***.

5) luego aparecerá el mensaje:

Cc:

Que significa, si se quiere mandar copia a otra dirección de correo electrónico.

6) finalmente se retorna la línea de comandos de UNIX.

A continuación se identifica cada paso para enviar un mail en una sesión de UNIX, además haremos el alcance que para enviar un mail a un usuario del mismo sistema basta con digitar su ***login***:

1) [dlazcano@jbozo dlazcano]$ mail dlazcano@jbozo.inf.ucv.cl

2) Subject: feliz año 2000

3) hola como estas

4) .

5) Cc:

6) [dlazcano@jbozo dlazcano]$

Para usar MAIL en la recuperación de correo desde el directorio donde esta guardado, se ejecuta el comando de la siguiente manera:

[dlazcano@jbozo dlazcano]$ mail

si es que hay mensajes aparecerá el siguiente mensaje:

Mail version 8.1 6/6/93. Type ? for help.

"/var/spool/mail/dlazcano": 1 message 1 new

>N 1 dlazcano@jbozo.inf.u Mon Jan 17 21:26 11/396 "hola"

&

El & que aparece es el que indica que se esta listo para que se ingresen comandos, para ver el mensaje se digita el número que le corresponde en este caso el ***1***.

Mail version 8.1 6/6/93. Type ? for help.

"/var/spool/mail/dlazcano": 1 message 1 new

>N 1 dlazcano@jbozo.inf.u Mon Jan 17 21:26 11/396 "hola"

&1

y se mostrara el mensaje ***1***:

Message 1:

From dlazcano Mon Jan 17 21:26:28 2000

Date: Mon, 17 Jan 2000 21:26:28 −0300

From: Daniel Lazcano <dlazcano@jbozo.inf.ucv.cl>

To: dlazcano@jbozo.inf.ucv.cl

Subject: hola

&

luego para grabar se digita :

&q

aparecerá el siguiente mensaje:

Saved 1 message in mbox

luego en la línea de comandos digitar cualquiera de los siguientes comandos:

Los comandos de mail son:

+	Se mueve al siguiente mensaje y lo lista
−	Se mueve al mensaje anterior y lo lista
?	Imprime la lista de mandatos de mail
R	Responde al autor
D	Elimina el mensaje vigente
H	Reimprime la lista de mensajes
Q	Sale del programa y guarda los mensajes en el archivo del correo predeterminado, mbox.
T	Teclear o listar mensaje vigente
X	Sale del programa y no guarda mensajes en mbox.
N	Pasa al siguiente mensaje y lo lista

El mandato mail también soporta los mandatos de redireccionamiento en el código shell. Esto representa un método sencillo para enviar la salida de programas y archivos grandes con el formato de mensajes. Para usar un método de redireccionamiento se utiliza el mandato mail con la opción −s(subject)seguida de una dirección de correo.

Por ejemplo para enviar una copia del calendario del mes actual, se redirecciona la salida del mandato CAL , como a continuación se muestra:

[dlazcano@jbozo dlazcano]$ cal| mail −scalendario dlazcano@jbozo.inf.ucv.cl

el carácter | representa un tubo es decir la salida del comando cal será la entrada de mail.

para enviar rápidamente un archivo.

[***dlazcano@jbozo dlazcano]$ mail −sarchivo dlazcano@jbozo.inf.ucv.cl***

el **−*s*** es el equivalente al subject.

9 FTP

Para descargar archivos directamente en la red se usa el mandato FTP (file transfer protocol).

Se puede conectar por un nombre:

[dlazcano@jbozo dlazcano]$ ftp jbozo.inf.ucv.cl

o una dirección IP:

[dlazcano@jbozo dlazcano]$ ftp 206.246.150.88

ambos permiten conectar con el mismo computador remoto. Por ejemplo:

[dlazcano@jbozo dlazcano]$ ftp jbozo.inf.ucv.cl

si se conecta satisfactoriamente aparece el siguiente mensaje:

Connected to jbozo.inf.ucv.cl.

220 jbozo.inf.ucv.cl FTP server (Version wu−2.4.2−VR17(1) Mon Apr 19 09:21:53 EDT 1999) ready.

Name (jbozo.inf.ucv.cl:dlazcano):

Solicita el login:

Connected to jbozo.inf.ucv.cl.

220 jbozo.inf.ucv.cl FTP server (Version wu−2.4.2−VR17(1) Mon Apr 19 09:21:53 EDT 1999) ready.

Name (jbozo.inf.ucv.cl:dlazcano):dlazcano

Luego solicitara el password:

331 Password required for dlazcano.

Password:

Se digita el password, recordar que el password no se ve por pantalla, si el password es correcto nos permite la conexión:

Remote system type is UNIX.

Using binary mode to transfer files.

ftp>

se bajara el archivo ***mbox***:

ftp> get mbox

local: mbox remote: mbox

200 PORT command successful.

150 Opening BINARY mode data connection for mbox (345 bytes).

226 Transfer complete.

ftp>

También se puede tener una conexión interactiva con los comandos open y close:

%***[dlazcano@jbozo dlazcano]$ FTP***

FTP> open jbozo.inf.ucv.cl

Luego la conexión se cierra con close.

Mandatos FTP:

!	Ejecuta un mandato shell
Ascii	Especifica descarga de archivo de texto.
Binary	Especifica descarga de archivo binario
Bye	Cierra la conexión abierta y sale de FTP
Put file	Envía un archivo a un Pc remoto
Get file	Descarga el archivo file desde el directorio vigente del Pc remoto
Mget n	Descarga archivos múltiples de acuerdo a un patrón n.
Mput n	Enviar archivos múltiples a un Pc remoto de acuerdo a un patrón n.
Close	Cierra la conexión abierta
Open	Abre una conexión

MANUAL DE REFERENCIA TALLER BASICO DE UNIX

PROYECTO MEJORAMIENTO CALIDAD DE LA EDUCACION

− 1 −

NUCLEO:

Asigna recursos

Y

Controla procesos

SHELL: Interpreta

Y/O

Envía instrucciones al núcleo

INTERFAZ GRAFICA

Y

LINEA DE COMANDOS

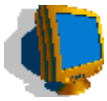
COMPUTADOR

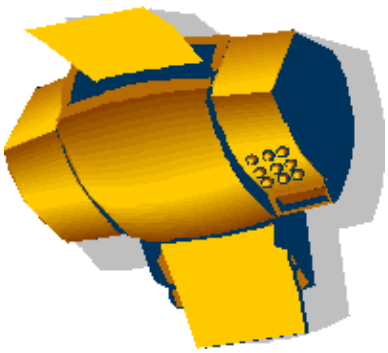
ARCHIVO

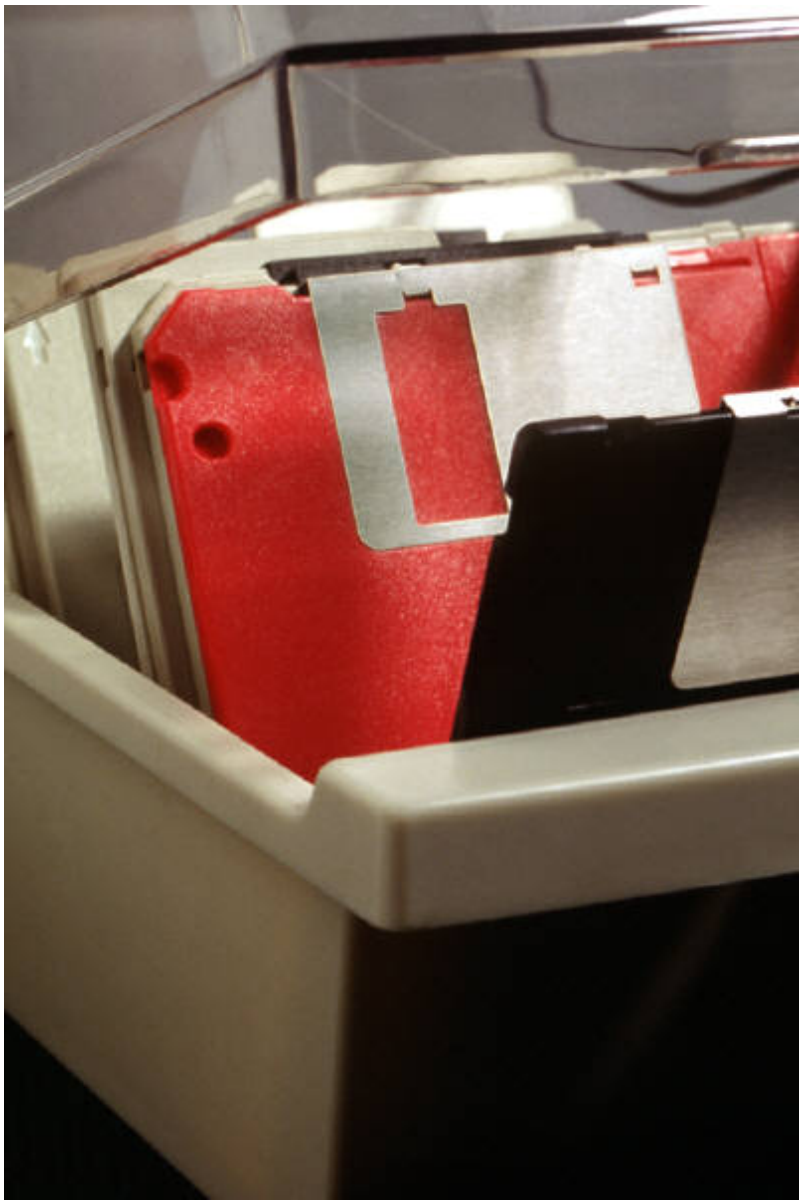
IMPRESORA

UTILIDAD

PANTALLA







hola mundo emacs

−−−−:**−F1 prueba1 (Lisp Interaction)−−L1−−All−−−−−−−−−−−−−−−−−−−−

hola mundo emacs

−−−−:**−F1 prueba1 (Lisp Interaction)−−L1−−All−−−−−−−−−−−−−−−−−−−−

−−−−:**−F1 pruebaprimera (Lisp Interaction)−−L1−−All−−−−−−−−−−−−−−−−−−−−

hola mundo emacs

−−−−:**−F1 pruebaprimera (Lisp Interaction)−−L1−−All−−−−−−−−−−−−−−−−−−−−

hola mundo emacs |hola mundo emacs

|

|

|

−−−−:−−−F1 prueba1 (Fundamental −−−−:−−−F1 prueba1 (Fundamental)

hola mundo UNIX | hola mundo UNIX

|

|

−−−−:−−−F1 prueba2 (Fundamental −−−−:−−−F1 prueba2 (Fundamental)

hola mundo emacs esta es la prueba3

|

|

|

|

−−−−:−−−F1 prueba1 (Fundamental −−−−:−−−F1 prueba3 (Fundamental)

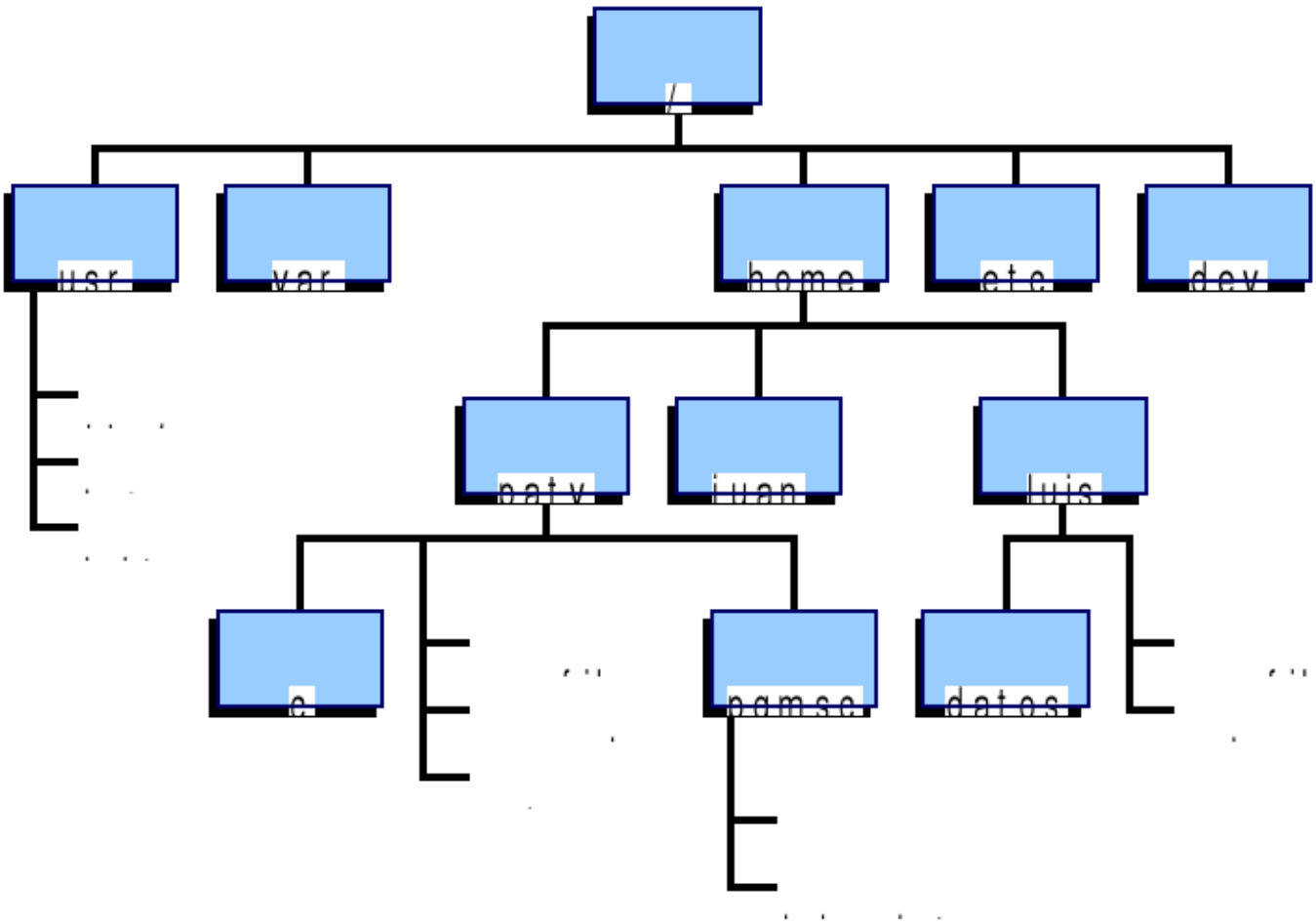
hola mundo UNIX esta es la prueba4|

|

|

|

−−−−:−−−F1 prueba2 (Fundamental −−−−:−−−F1 prueba4 (Fundamental)


/
usr
var
home
etc
dev
paty
juan
luis
a
pamse
datos