Tutorial breve de UNIX para las prácticas de la asignatura "Fundamentos de Informática" .

José Otero Rodríguez

Índice general

Capítulo 1

Generalidades.

En esta asignatura se empleará un computador Alpha corriendo la versión de UNIX OSF1 V5.1A de la casa Compaq. Seguramente ha trabajado antes con ordenadores, pero probablemente nunca ha trabajado con uno conectándose de forma remota, utilizando una conexión de red. Durante las prácticas de esta asignatura, se conectará al ordenador `centauro.aulario.uniovi.es`, desde un PC común o desde una terminal gráfica, un conjunto de teclado, monitor y ratón que sirve para interactuar con el ordenador al que se conecte. Puede conectarse también desde su domicilio, si dispone de conexión a Internet, desde un Cyber-café o desde un ordenador de una red de otra universidad, campus o de una empresa privada. Todas esas redes están interconectadas, de modo que se puede llegar desde cualquiera de ellas hasta la red del Aulario de Viesques en donde se conecta Centauro. En la figura 1.1 se pueden ver gráficamente estas situaciones. En cualquier caso, lo primero que necesita es un nombre de usuario y una clave. Ambas cosas las proporciona el Servicio de Informática tras cumplimentar un impreso y acreditar de estar matriculado en esta asignatura. Pueden pasar unos días hasta que pueda usar el ordenador, es necesario realizar una serie de tareas y repetirlas para muchos compañeros suyos, de modo que no es inmediato.
Un aspecto importante es que la clave y/o nombre de usuario puede ser distinto del que tenga para la realización de prácticas en las salas de PC's. Cuando disponga de nombre de usuario, clave y el administrador del sistema lo haya configurado adecuadamente, podrá usar el ordenador. En este momento se puede decir que "tiene cuenta" en el sistema. Esto implica que puede conectarse al ordenador, identificándose con su nombre de usuario y que puede ejecutar programas en el mismo así como guardar información en

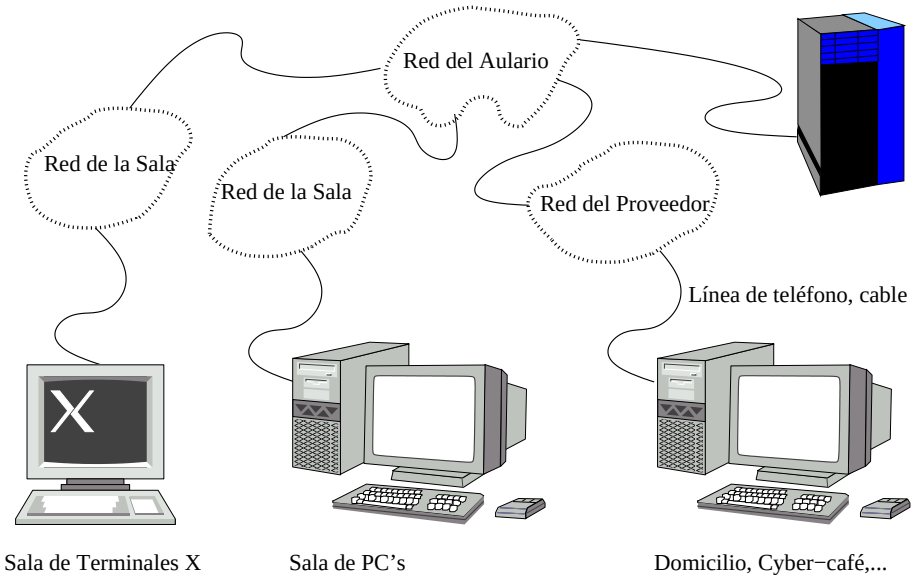

Figura 1.1: Distintas formas de trabajar en el ordenador de prácticas.

ficheros, utilizar el correo electrónico, navegador de Internet, etc. Al proceso de conexión e identificación se le suele denominar "entrar en la cuenta".
Su clave es privada y por lo tanto no debe compartirse con nadie, tampoco es conveniente apuntarla o utilizar como clave cosas como el día de su cumpleaños, su nombre de pila, apellidos,... en general cualquier cosa fácil de adivinar. Lo ideal es mezclar letras mayúsculas, minúsculas, cifras y otros caracteres. Para evitar que alguien le espíe y descubra su clave, mientras la escribe, no aparecerá en la pantalla, por así decirlo, escribirá a ciegas.
Si alguien se apodera de su clave, entra en su cuenta y realiza cualquier actividad ilegal, usted será el responsable, a no ser que se demuestre que la clave le fue sustraída de un modo ilícito. La clave inicial que le da el administrador es válida sólo una vez, en cuanto entre en la cuenta, tendrá que cambiarla usted mismo mediante una serie de órdenes que se detallarán en las secciones siguientes. Esta nueva clave será válida un cierto tiempo, el que determine el administrador.
Si alguna vez no puede entrar en su cuenta, puede ser debido a varias circunstancias, entre ellas equivocarse de clave. La misma secuencia de caracteres escritos con mayúsculas o con minúsculas no son la misma clave, preste atención a la tecla que bloquea las mayúsculas. Si se equivoca tres veces, la cuenta

se desactiva. Esto se hace por cuestiones de seguridad, para que alguien no pueda intentar una y otra vez adivinar su clave. Para activar la cuenta de nuevo deberá acudir al Servicio de Informática, no perderá ninguna información, ni ficheros ni correos electrónicos.

Capítulo 2

Uso de las terminales X.

Las terminales X son dispositivos que constan de un monitor en color, teclado, ratón y una serie de circuitos que le permiten conectarse a un ordenador remoto así como mostrar texto y gráficos en el monitor. Por sí misma, una terminal X no puede realizar ninguna tarea. Su única misión es servir de "interlocutor" entre el usuario y el ordenador remoto.
Cuando una terminal X está libre (no hay nadie usándola), muestra algo similar a lo que se ve en la figura 2.1. En esa figura se observa el texto "Please enter your username" y debajo aparece un campo dispuesto para realizar precisamente eso, introducir su nombre de usuario, en general será su número de matrícula antecedido de los caracteres "zz". Para entrar en su cuenta, haga clic en ese campo y escriba su nombre de usuario (la apariencia será similar a lo que se muestra en la figura 2.2, a continuación pulse el botón con el rótulo "OK". En ese momento la pantalla cambiará y se mostrará algo similar a la figura 2.3, se pide que se introduzca la clave. En este punto, es importante recordar que la clave se escribirá a ciegas, para hacer más difícil que alguien le espíe. Cuando acabe de escribir la clave, pulse "OK". Si ha introducido correctamente su nombre de usuario y clave, entrará en su cuenta.
El aspecto de la pantalla será similar al que está acostumbrado si ha usado alguna vez un ordenador, se verá un escritorio con una serie de iconos y botones (véase 2.4). Si es la primera vez que entra en el sistema, estarán abiertas una serie de ventanas con aplicaciones útiles, como la ayuda o un explorador de ficheros.
En el centro de la barra de tareas que está en la parte inferior de la pantalla se pueden ver cuatro botones. Estos cuatro botones permiten cambiar entre "espacios de trabajo", en cada uno de ellos se pueden tener abiertas las ven-

tanas que se deseen, de modo que es como si el monitor fuese cuatro veces más grande.
A la izquierda de esos botones se puede ver un icono que representa un papel y un lápiz. Encima se puede ver una flecha que apunta hacia abajo. Si se pulsa en esa flecha se despliega una barra de herramientas con más iconos, como se aprecia en la figura 2.5. Esta barra de herramientas permite el acceso a dos aplicaciones útiles para la realización de las prácticas, la terminal (el icono representa un dispositivo parecido a la propia terminal X) y el editor de texto.
La terminal es una aplicación que simula un dispositivo distinto a la terminal X, una terminal de texto. Este tipo de dispositivos no muestra gráficos, sólo texto. Si pulsa en ese icono, aparecerá una ventana como la que se ve en la figura 2.6.
En dicha ventana aparece el texto `centauro.aulario.uniovi.es>` y a continuación, intermitentemente, un rectángulo del tamaño de un carácter y de color destacado contra el fondo. Lo primero se denomina "prompt del sistema", indica que el ordenador está listo para aceptar órdenes escritas, que se denominan comandos. El programa que lee esas órdenes y las procesa se denomina "intérprete de comandos". El rectángulo que aparece de forma intermitente se denomina "cursor" y es el punto de la pantalla de donde parecen surgir los caracteres que se teclean.
La ventana de la terminal de texto puede moverse, redimensionarse, minimizarse o cerrarse, al igual que las de los sistemas operativos que haya usado en cursos anteriores. Para cerrarla puede hacer doble clic sobre el pequeño botón similar a un guión de la esquina superior izquierda. Para minimizarla haga clic una vez sobre el botón de la esquina superior derecha que es casi como un punto. Si minimiza la ventana, el programa que esté ejecutándose en esa terminal de texto no se interrumpe, la ventana se convierte en un icono sobre el escritorio (ver figura 2.14), si hace doble clic en ese icono, la ventana recobra su tamaño. A la derecha de ese botón se encuentra otro más grande, cuadrado, que sirve para maximizar la ventana. Si lo pulsa, la ventana ocupará toda la pantalla, el botón cambia de apariencia y parece estar oprimido. Si lo vuelve a pulsar, retomará la apariencia anterior, en relieve, y la ventana volverá a su anterior tamaño. Para mover la ventana o redimensionarla, el proceso es idéntico al que se sigue en otros sistemas operativos.
Volviendo a la barra de herramientas que se mostraba en la figura 2.5, si pulsa sobre el icono del editor de texto, se abrirá una ventana como la de la figura 2.7. Esta es una aplicación sencilla de edición de textos pero que funciona de

forma muy parecida a otras similares de sistemas operativos conocidos por usted.
Observe que las ventanas se solapan, la ventana que está "encima", tiene un borde de un color más vivo que el resto. Se dice que esa ventana está en "primer plano" o que tiene el foco. Cualquier cosa que tecleemos, aparecerá en o interactuará con esa ventana y no con el resto.
Pinchando en el menú "File", se despliega un menú que permite abrir un fichero (Open), guardarlo (Save), guardarlo con otro nombre (Save As) o cerrar la aplicación (véase figura 2.9).
En la figura 2.8 se muestra el editor con un texto, un sencillo programa en C++. Para guardar este archivo, haga clic en "File" (la pantalla mostrará algo similar a lo que se ve en la figura 2.9) y después en "Save", se pedirá el nombre del fichero, como se ve en la figura 2.10, al acabar pulse "OK". Obsérvese que ahora aparece el nombre del fichero en la parte superior de la ventana (véase 2.11). La siguiente vez que se guarde un fichero, no será necesario que escriba su nombre, a no ser que desee específicamente guardarlo con otro nombre.
En la figura 2.12 se puede ver que las ventanas han intercambiado sus papeles, la que está en primer plano ahora es la de la terminal. Esto se consigue haciendo clic en la porción de la ventana que sobresale "por debajo" de la del editor.
En la misma figura se muestra como se compila el programa y como se ejecuta. Más adelante se explicará en que consiste compilar, en este momento se trata de poner un ejemplo de un comando, es decir una orden que se da al ordenador. El comando en cuestión es `g++` y procesa un archivo, cuyo nombre se escribe a continuación. Para que se ejecute el comando se teclea la orden, en este caso necesita saber cual es el archivo que procesa (por eso se escribe a continuación) y después se pulsa la tecla "return", a veces rotulada "enter". En este caso, el comando produce un programa ejecutable. Para ver su funcionamiento se escribe el nombre (por defecto "a.out") y se pulsa "return", como si fuese un comando más.
En la siguiente línea aparece el texto `g++ hola.cpp -o hola`. Este ejemplo ilustra la cuestión de las "opciones" de un comando, las alternativas que ofrece el comando a su funcionamiento o bien características adicionales. Se identifican por un guión y una letra. Pueden escribirse varias opciones seguidas después de un mismo guión, sin espacios entre ellas. En el caso de la figura 2.12, la opción sirve para cambiar el nombre del programa ejecutable que produce el compilador. Este nombre se escribe después de la opción `-o`.

Figura 2.1: Contenido de una terminal X libre.

Para cerrar el editor de texto pulsar en “File” y después en “Close”, como se muestra en la figura 2.13.
Para salir de la cuenta, pulse en el botón pequeño que hay al lado de los cuatro botones del centro de la barra de herramientas rotulado con la palabra “exit”. Aparecerá algo similar a lo que se muestra en la figura 2.15, pulse en el botón rotulado como “Continue logout” y saldrá de su cuenta.

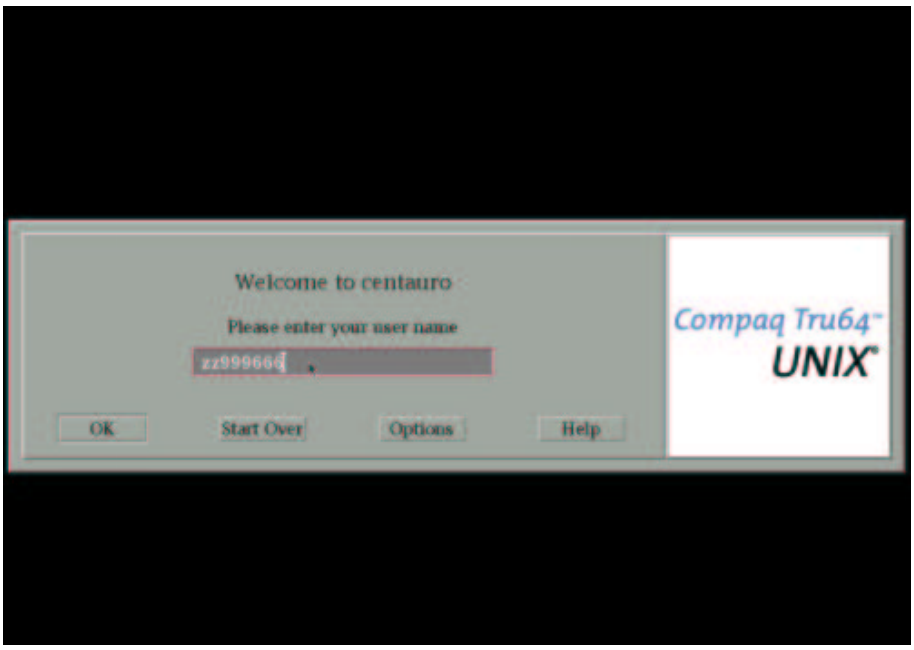

Figura 2.2: Introduciendo el nombre de usuario.

Figura 2.3: Introduciendo la clave.

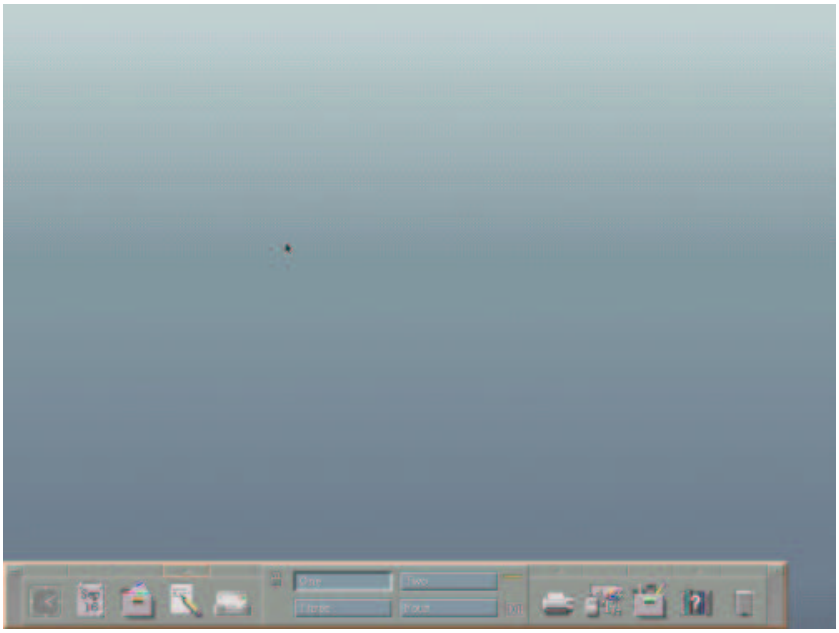

Figura 2.4: El escritorio CDE.

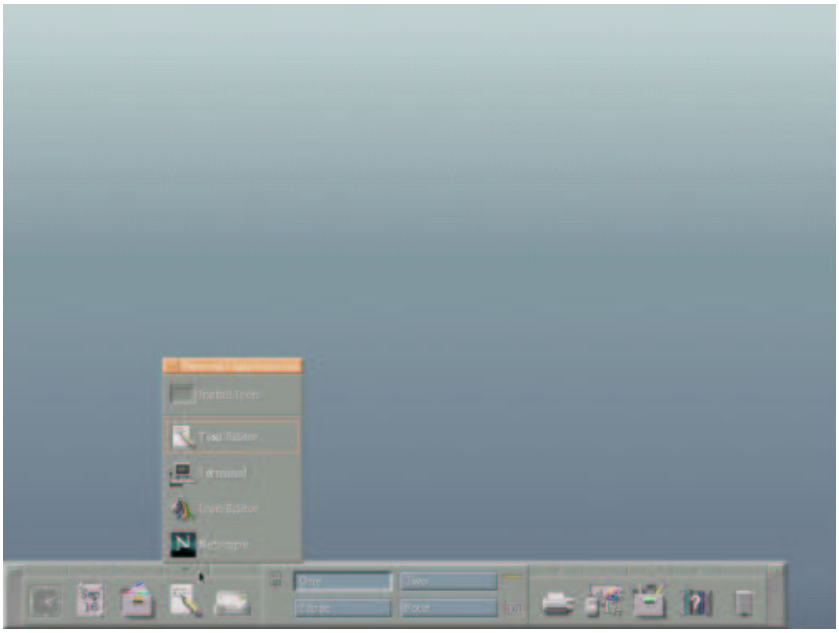

Figura 2.5: Desplegando una barra de herramientas.

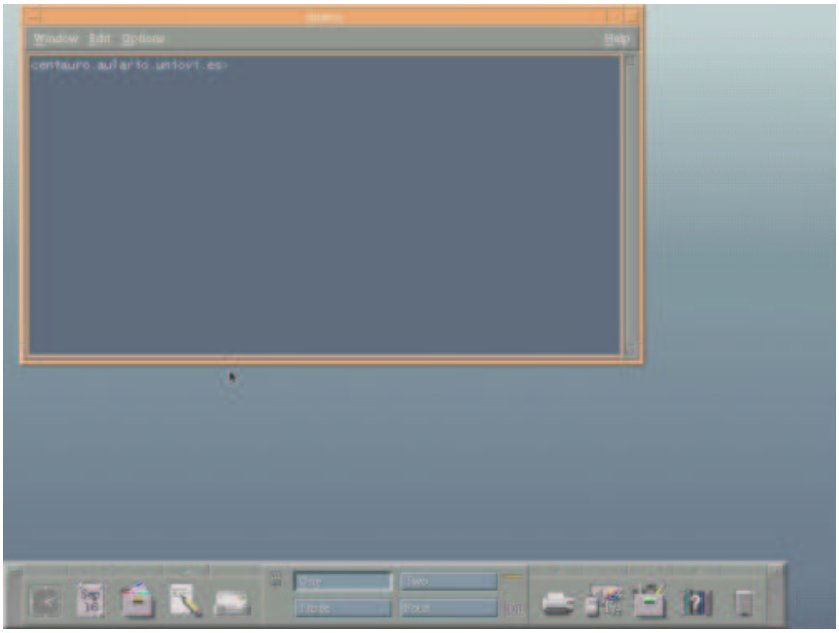

Figura 2.6: Abriendo una terminal de texto.

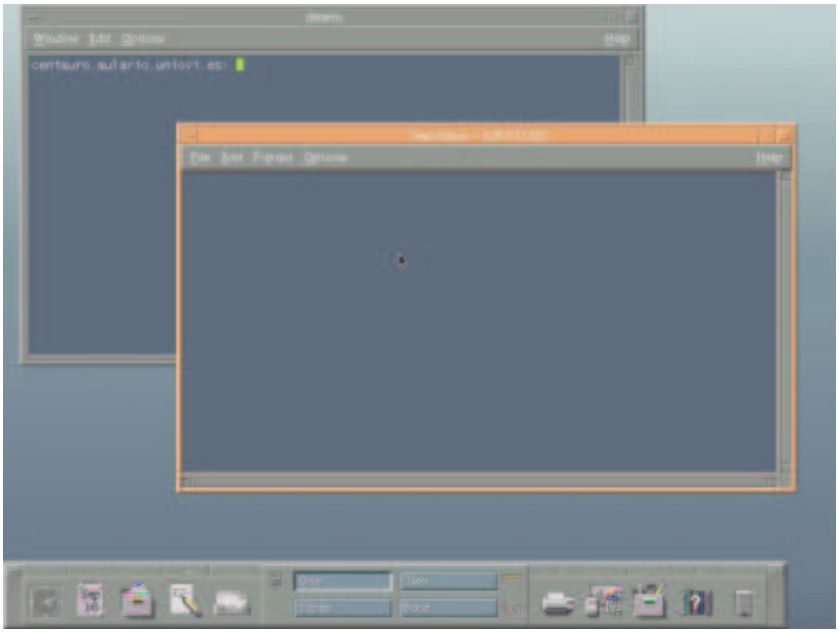

Figura 2.7: Un editor de texto sencillo.

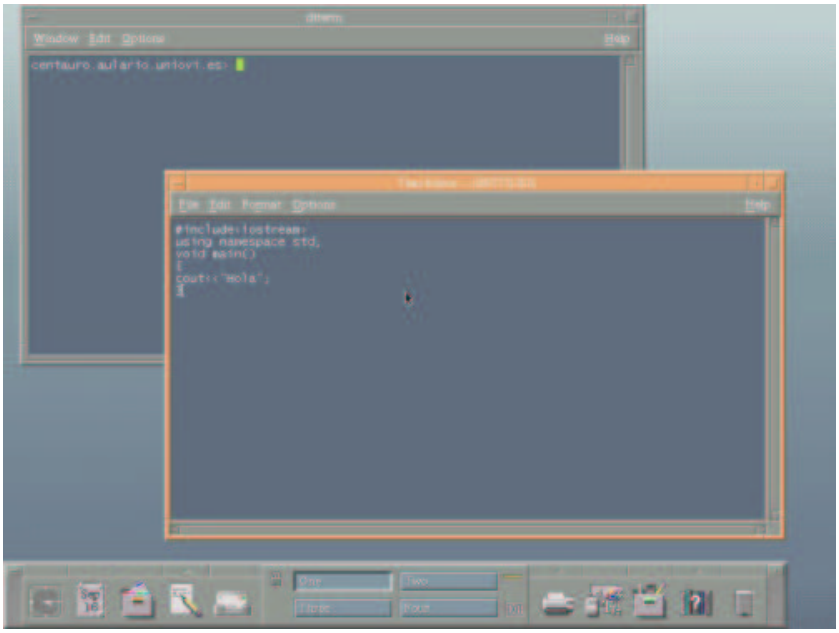

Figura 2.8: El editor con un programa en C++.

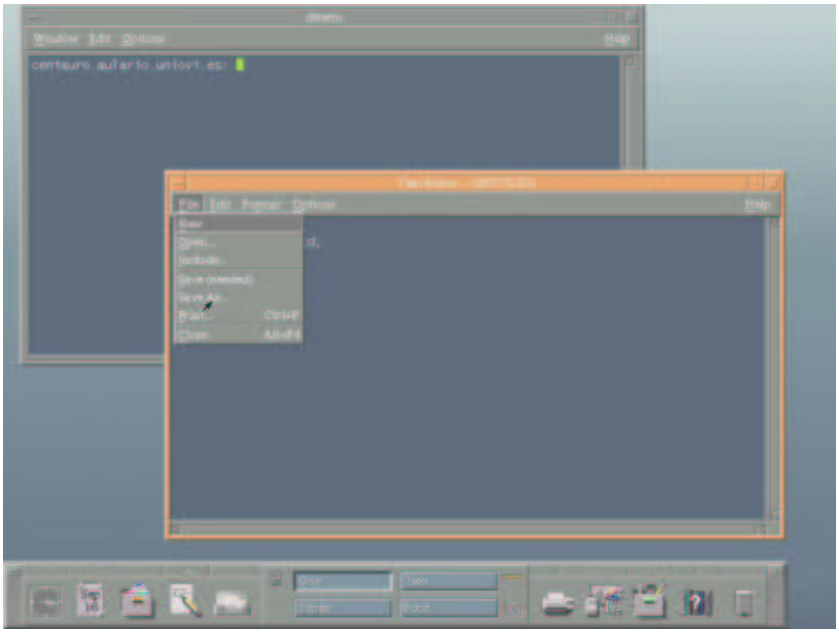

Figura 2.9: El menú File.

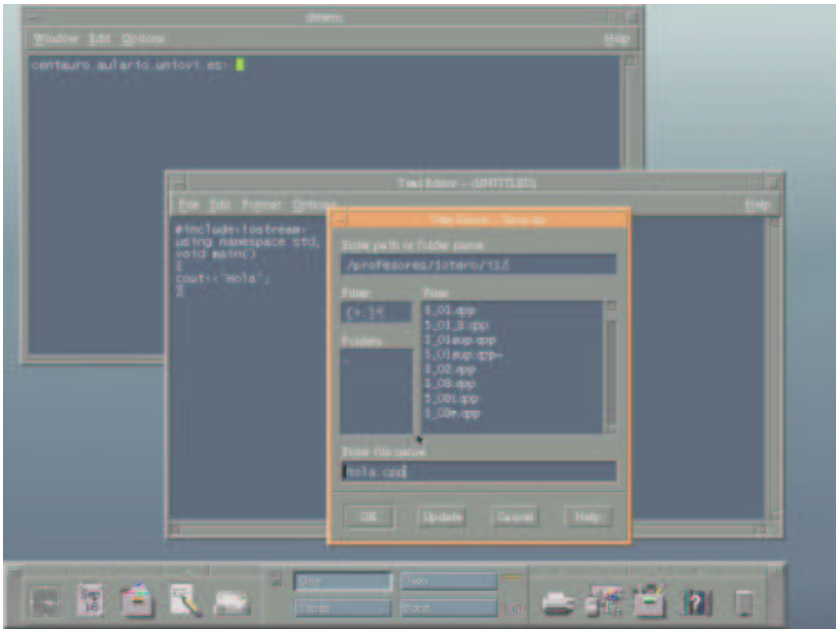

Figura 2.10: Guardando el fichero.

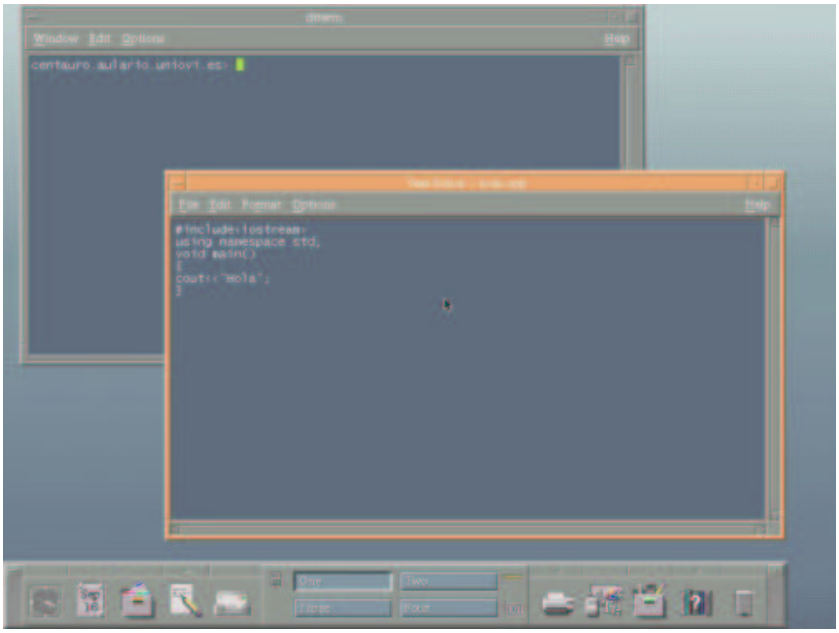

Figura 2.11: El nombre del fichero aparece en la ventana.

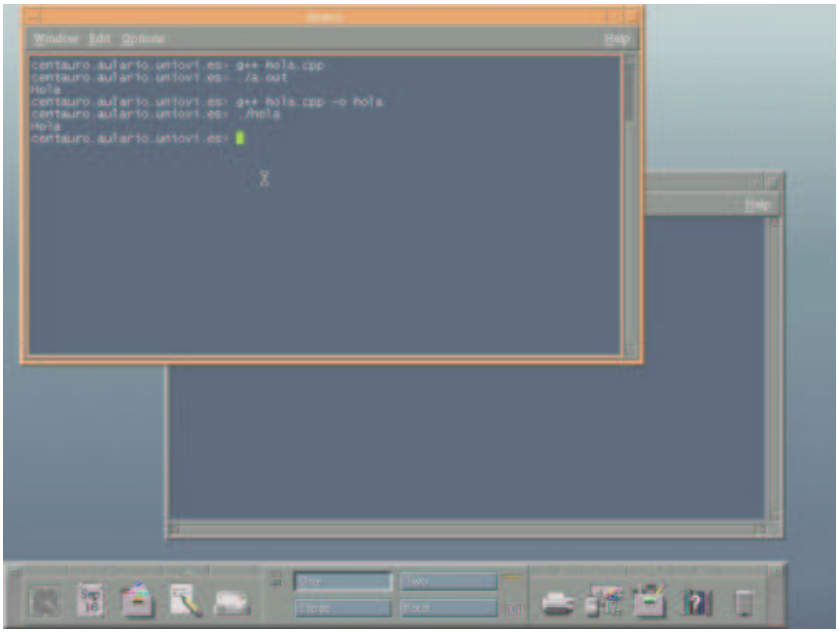

Figura 2.12: Paso de una ventana a primer plano y ejecución de comandos.

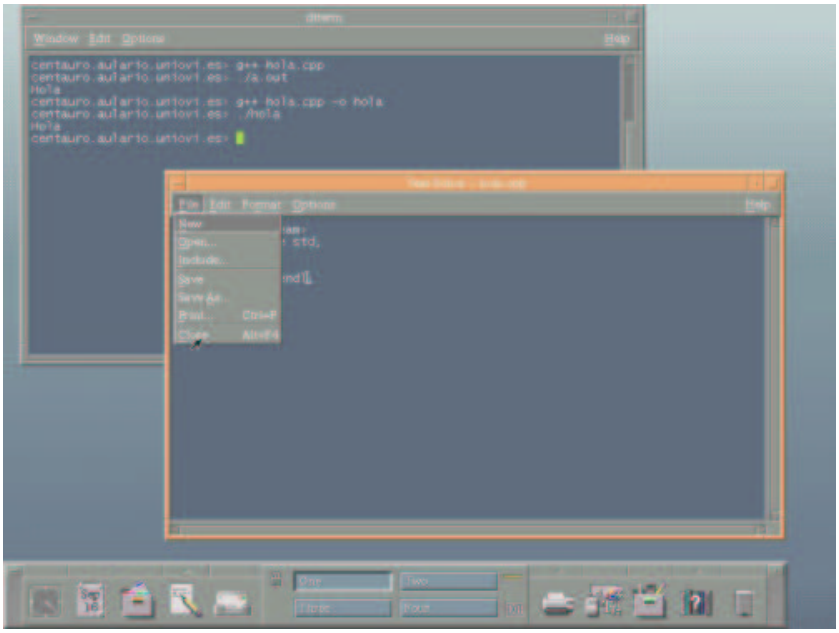

Figura 2.13: Cerrando el editor de texto.

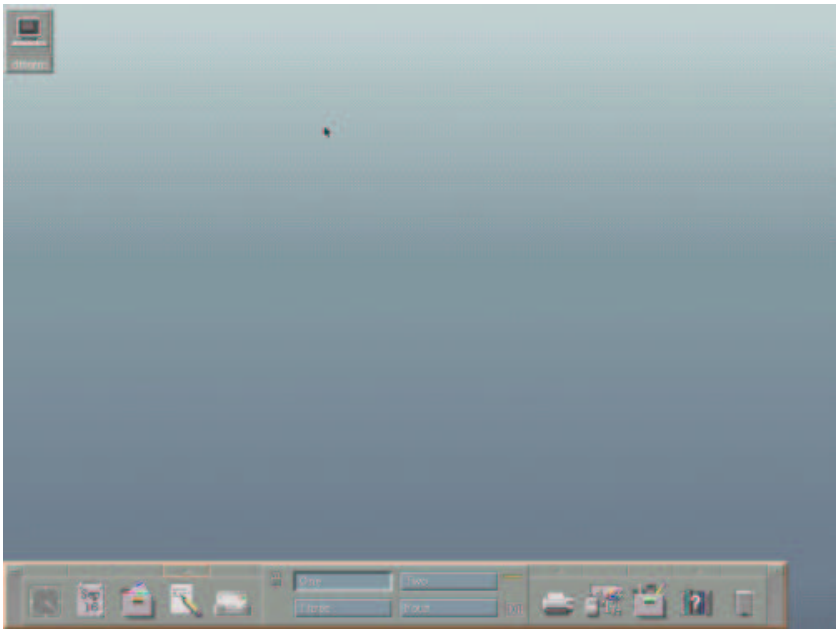

Figura 2.14: Ventana de la terminal de texto minimizada.

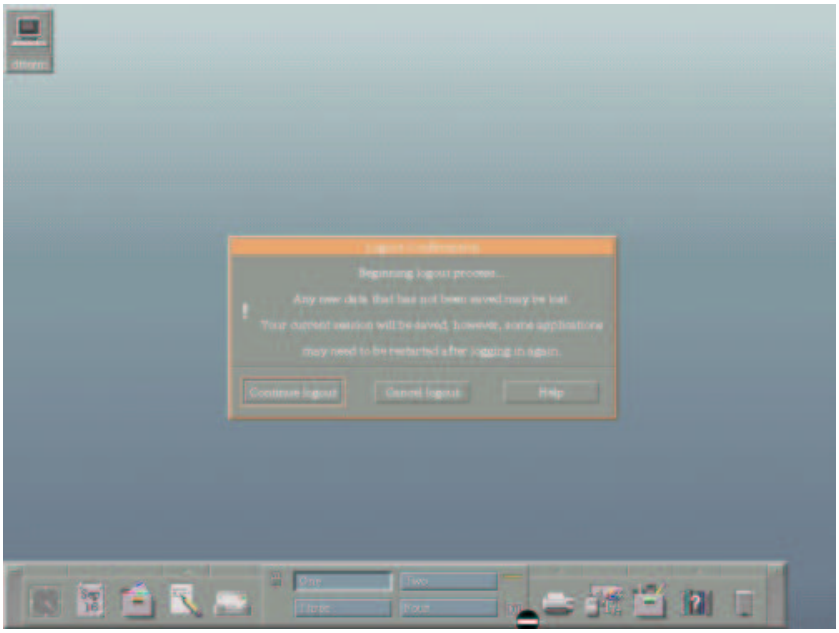

Figura 2.15: Saliendo de sesión.

Capítulo 3

Acceso mediante telnet.

En la mayor parte de los sistemas operativos existe instalado un programa telnet que permite conectarse a un ordenador remoto. En esta sección se estudiará el proceso para los sistemas operativos de Microsoft, aunque para otros sistemas operativos el proceso sería similar.
Para acceder al programa telnet, pulse en "inicio " y después en "ejecutar". En el cuadro de diálogo que aparece escriba `telnet centauro.aulario.uniovi.es` (ver 3.1), recuerde que esta ventana debe de estar en primer plano y que después debe hacer clic en el espacio que hay delante de la palabra "Abrir".
Finalmente pulse "Aceptar", se cierra la ventana anterior y se abre otra como la de la figura 3.2.
El cursor aparecerá intermitentemente a continuación de la palabra "login". Escriba su nombre de usuario y pulse return, después escriba su clave, recuerde que lo hará a ciegas, no aparecerá en la pantalla, y pulse return de

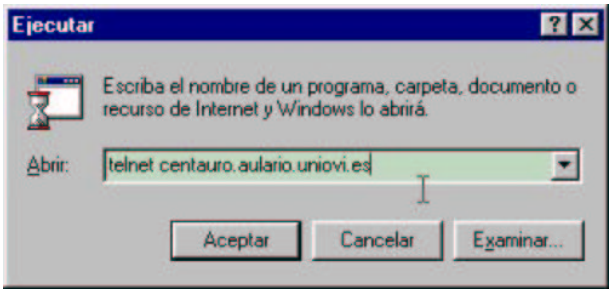

Figura 3.1: Menú ejecutar preparándose para lanzar el programa telnet.

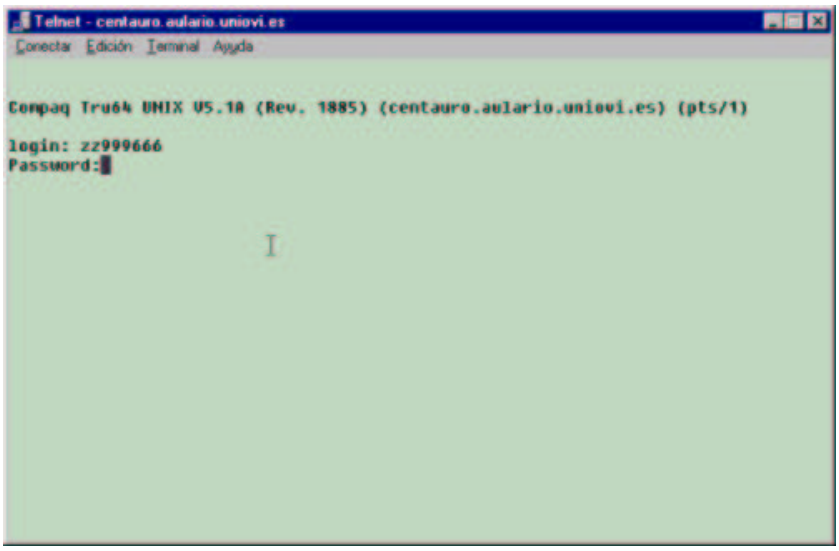

Figura 3.2: Petición de nombre de usuario y clave.

nuevo. La ventana cambiará y mostrará algo parecido a la figura 3.3. Desde este momento, esa ventana se comporta igual que una terminal de texto.

Para salir de la cuenta, escriba `exit` y pulse return, tal y como se muestra en la figura 3.4, aparecerá una ventana como la de la figura 3.5, con un único botón "Aceptar", púlselo.

Si se desea volver a entrar en la cuenta, pulse en el menú "Conectar" de la ventana de telnet y a continuación en "Sistema remoto ...", como se muestra en la figura 3.6 o bien en el nombre del ordenador si aparece en la lista del final del menú. Si hace lo primero, aparecerá un cuadro de diálogo (ver figura 3.7) en el que podrá escribir el nombre del ordenador al que desea conectarse o bien despliegue la lista y haga clic en el nombre del ordenador, si ya se había conectado con anterioridad a él. Esto se realiza a continuación de donde dice "Nombre de host". Después pulse en el botón "Conectar", a partir de este momento el proceso es como el que se describió al principio.

Si no desea volver a conectarse, cierre el programa telnet pulsando en "Conectar" y después en "Salir" o bien pulsando en el botón de la parte superior de la derecha que cierra la ventana.

Se pueden abrir tantas ventanas con el programa telnet como se desee, basta con repetir el proceso utilizado para abrir la primera.

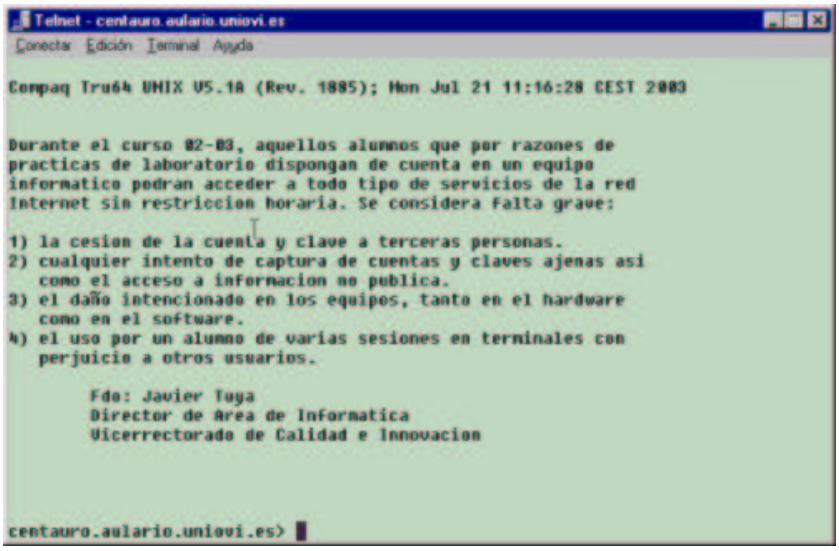

Figura 3.3: Ventana de telnet después de entrar en la cuenta.

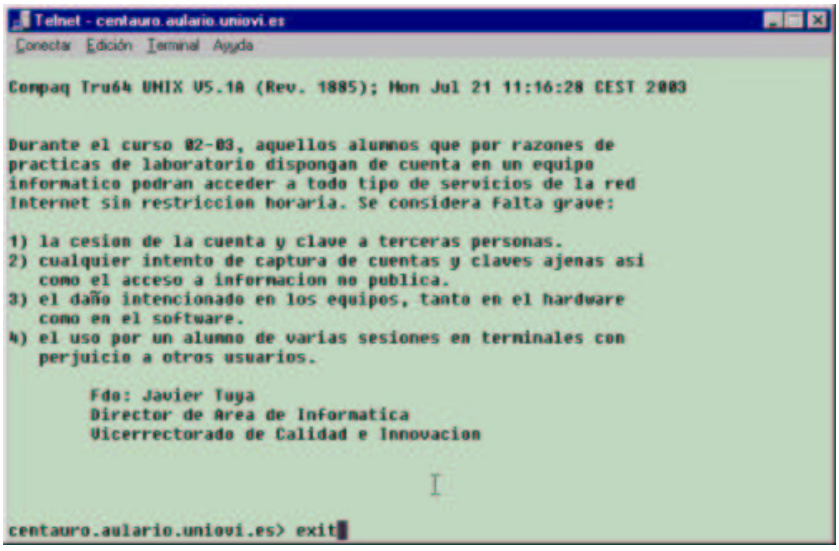

Figura 3.4: Preparándose para salir de la cuenta.

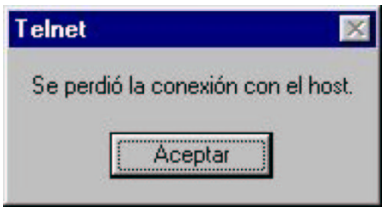

Figura 3.5: Ventana informando de que se ha salido de la cuenta.

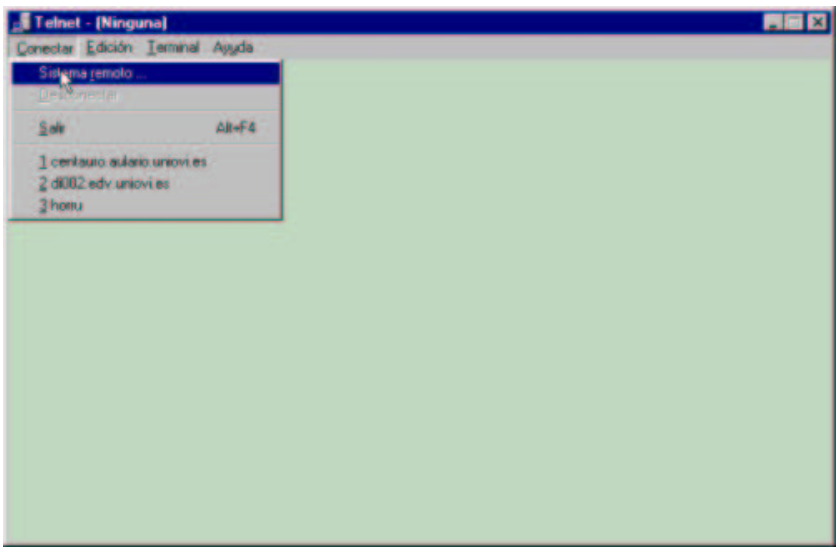

Figura 3.6: Conectándose utilizando el menú "Conectar".

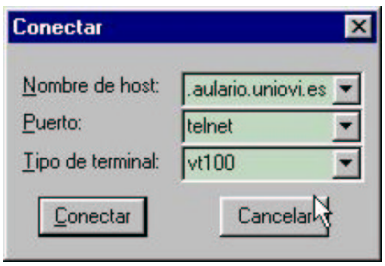

Figura 3.7: Especificando el ordenador remoto.

Capítulo 4

Directorios.

Al igual que en cualquier sistema operativo de la familia Microsoft, en el Unix que se utiliza en las prácticas, se puede crear una estructura de directorios para almacenar en ellos, de una forma ordenada, los ficheros que vaya creando, programas, información que se baje de Internet, que alguien le envíe por correo electrónico, etc.
Para los lectores con menor experiencia, existen dos analogías con objetos del mundo real que permiten entender que es un directorio y un fichero.
Una posible analogía es la carpeta clasificadora con la que algunos de ustedes acuden a clase. Esta carpeta esta dividida en subcarpetas, cada una de las cuales contendrá hojas de papel con apuntes, por ejemplo. Pero también es posible que una subcarpeta contenga otras dos (o más), por ejemplo para separar los ejercicios de una asignatura de la teoría de la misma asignatura. Pues bien, las hojas serían el análogo de los ficheros y las carpetas el análogo de los directorios. Como cada carpeta sólo puede estar contenida en otra, se puede pensar que, las divisiones sucesivas clasifican los ficheros de una forma que se puede representar como un árbol, en donde el tronco surge de un directorio raíz y este se va dividiendo sucesivamente, de la misma forma que las ramas de un árbol se bifurcan. Debido a esto, muchas veces se representan los directorios por un árbol, que crece de forma horizontal, de izquierda a derecha. En el ejemplo siguiente, desde el directorio `tmp` "cuelgan" varios directorios y de alguno de ellos otros más.

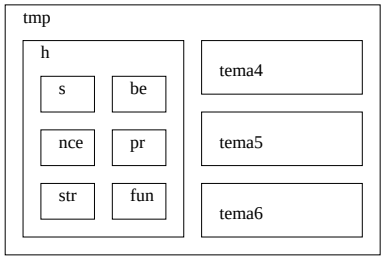

Figura 4.1: Representación de una estructura de directorios como cajas conteniendo cajas.

```
tmp
|-- h
|   |-- s
|   |-- be
|   |-- nce
|   |-- pr
|   |-- str
|   `-- fun
|-- tema4
|-- tema5
`-- tema6
```

Otra posible analogía es pensar en la estructura de directorios como una caja conteniendo una serie de objetos y además otras cajas, como se muestra en la figura 4.1.

En Unix, un directorio se especifica concatenando los nombres del directorio y los de los directorios que lo contienen (de derecha a izquierda), separados por un carácter `/`, y así sucesivamente hasta llegar al directorio raíz, representado por un carácter `/`. Por ejemplo `/tmp/h/fun`. Esto es el nombre completo del directorio `fun`, no siempre es necesario especificar el nombre completo de un directorio.

El usuario puede crear directorios, añadir carpetas, esto se realiza mediante el comando `mkdir` *nombre*, donde nombre es el nombre del directorio que se quiere crear. Este comando toma como referencia el directorio desde donde se invoque (en seguida se comenta como se mueve uno por el árbol de directorios). Imagine que se invoca el comando desde el directorio `/tmp/tema4` escribiendo `mkdir ejemplos`. Se crearía otra carpeta dentro de `tema4` de

nombre `ejemplos`, de modo que el árbol quedaría:

```
tmp
|-- h
|   |-- s
|   |-- be
|   |-- nce
|   |-- pr
|   |-- str
|   `-- fun
|-- tema4
|   `-- ejemplos
|-- tema5
`-- tema6
```

El usuario puede moverse por el árbol de directorios, mediante el comando `cd` (**c**hange **d**irectory). En este punto es conveniente introducir dos convenciones que hacen más fácil designar el nombre de un directorio. En primer lugar, el ordenador conoce en que directorio ejecuta el usuario los comandos. El usuario puede referirse a ese directorio con un sólo carácter, el punto '.', si le resulta más cómodo identifíquelo como "aquí" o "actual". Por otra parte, como cada subdirectorio "cuelga" de un único directorio, se permite referirse al directorio por encima de '.' como '..'. De esta forma, desde el directorio `/tmp/tema4/ejemplos`, uno se puede referir a `/tmp/tema4` como "`..`". Análogamente, el directorio `/tmp/h` se puede designar como `../../h`. De alguna forma se describe al ordenador como llegar al directorio, '..' significa "subir", de modo que lo anterior, equivale coloquialmente a "sube una vez, sube otra vez, baja a `h`". Por esa razón a veces se le llama al directorio identificado de esa forma como "path", camino en Inglés. En concreto, si se utiliza el '.' o el '..' se dice que es el path relativo (al directorio en el que se esté) y cuando se especifican todos los directorios hasta el raíz se denomina path absoluto. Explicados estos conceptos, decir que para usar el comando `cd` se escribe el comando y a continuación el path al directorio al que se quiere cambiar, de forma absoluta o relativa. Por ejemplo `cd /tmp/h` cambia al directorio `h` independientemente de donde se esté. Si se está en `/tmp/tema6`, se puede cambiar al mismo directorio escribiendo `cd ../h`.
Por último, el movimiento por el árbol de directorios afecta sólo a la terminal de texto desde donde se efectúe, de modo que el cambio de directorio en una

de ellas no implica el cambio en otras que puedan estar abiertas. Sin embargo la creación de directorios o archivos se percibe desde cualquier terminal que esté abierta.
Es necesario explicar *donde* se pueden crear los directorios. En un ordenador corriendo un sistema operativo de Microsoft, el usuario tiene la certeza de que sus ficheros se guardan en el disco duro del ordenador, quizás en un directorio predeterminado por el sistema, por el administrador o por uno que cree el usuario. En el caso del ordenador utilizado en práctica, cada nueva cuenta se crea de modo que el usuario sólo puede crear o almacenar ficheros y crear subdirectorios en uno predeterminado, generalmente `/practicas/zzxxxxxx` en donde `xxxxxx` será el número de matrícula de cada uno. Por defecto, cada vez que entra en sesión o abre una terminal de texto, el directorio actual será el que se ha comentado.
Tanto si usa las terminales X como si usa el programa telnet desde un PC ordinario, todos los programas y demás ficheros que cree estarán en el disco duro del ordenador remoto. Debe de pensar en la terminal como medio de comunicación entre usted y el ordenador remoto, sirve para enviar órdenes y para observar el resultado de estas. Más adelante se explicará como enviar o recibir ficheros desde un PC al ordenador en el que se realizan las prácticas.

Capítulo 5

Ficheros de texto, edición con joe.

En este curso las prácticas consistirán fundamentalmente en la escritura de programas en C++. Un programa en C++ es un texto con órdenes escritas siguiendo la sintaxis de este lenguaje. Para editar un programa en C++ puede usar el editor de texto del CDE, como se explicó en secciones anteriores, pero sólo si usa una terminal X (o un programa que la simule en un PC). Si quiere editar un programa en C++ usando exclusivamente el programa telnet, deberá utilizar un programa que funcione en una terminal de texto. Uno de estos programas es `joe`. Para crear un fichero de texto en el que se va a guardar un programa en C++, escriba `joe nombre.cpp` (pulse return), donde nombre es una secuencia de letras, números y otros caracteres y `.cpp` es la extensión del fichero, que es siempre la misma. Por ejemplo para editar un programa que se llame "hola.cpp" se escribirá `joe hola.cpp` y después se pulsará return, la pantalla cambiará y mostrará algo similar a lo que se ve en la figura 5.1.

Si se escribe el texto, la apariencia será la de la figura 5.2, nótese que en la parte superior de la pantalla se muestra información sobre la fila y la columna en la que está el cursor (el cuadrado negro), esto es de gran utilidad para localizar líneas concretas cuando se está depurando un programa (corrigiendo errores).

Este editor de texto se maneja exclusivamente con el teclado. Los comandos que normalmente sería accedidos con el ratón, en este caso se ejecutan mediante combinaciones de teclas. Todas las combinaciones se realizan manteniendo pulsada la tecla "control" o "ctrl", después y sin soltar esta tecla,

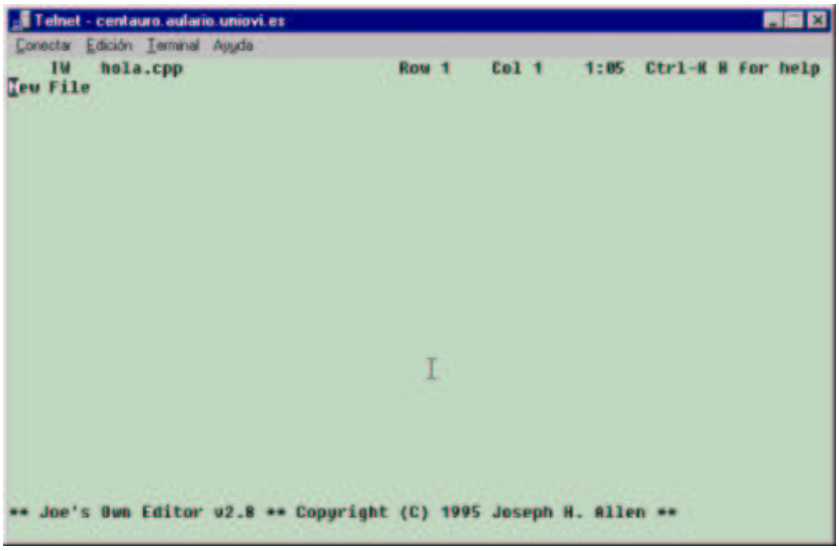

Figura 5.1: Creación de un fichero de texto con joe.

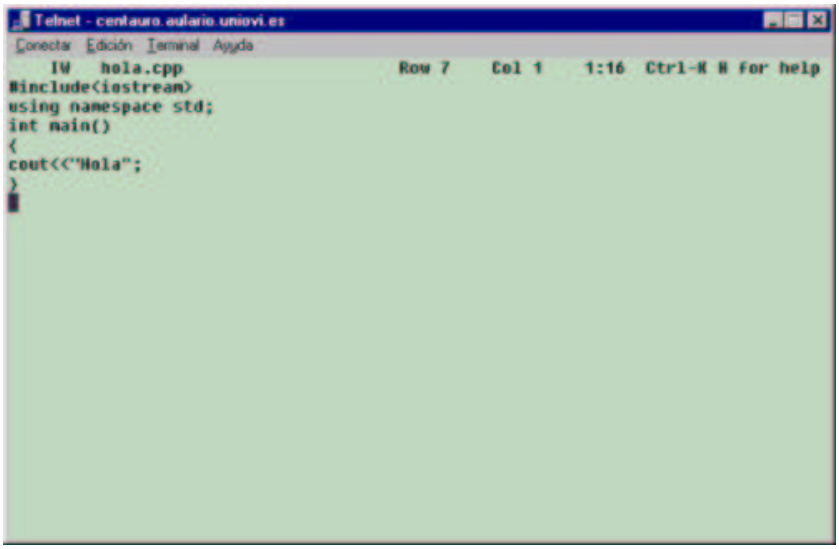

Figura 5.2: "joe" con el fichero.

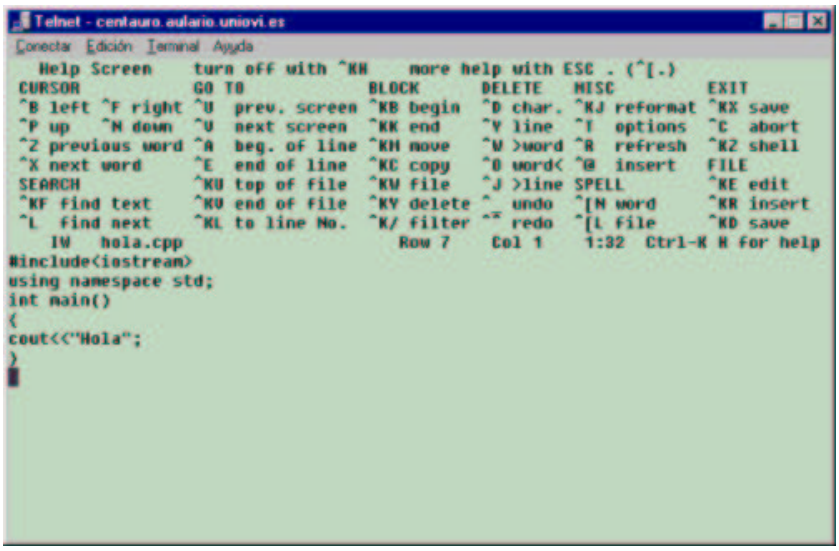

Figura 5.3: "joe" con la ayuda.

se pulsan otras. Existen muchas combinaciones de teclas. La primera que se verá es ctrl+k+h, en el programa viene indicada como `^kh`. Esto quiere decir que se pulsa la tecla ctrl y sin soltar, primero se pulsa 'k', se libera esta y después se pulsa 'h'. Si pulsa esa secuencia de teclas se mostrará algo similar a la figura 5.3, como se ve, aparece una ayuda sobre las combinaciones de teclas. El texto que aparece no se guarda con el fichero, para que desaparezca se teclea la misma secuencia de teclas que se utilizó para desplegarlo.

A continuación se muestran las combinaciones de teclas en modo texto para que se puedan examinar con mayor comodidad.

```
CURSOR           GO TO            BLOCK        DELETE   MISC       EXIT
^B left ^F right ^U  prev. screen ^KB begin  ^D char. ^KJ reformat ^KX save
^P up   ^N down  ^V  next screen  ^KK end    ^Y line  ^T  options  ^C  abort
^Z previous word ^A  beg. of line ^KM move   ^W >word ^R  refresh  ^KZ shell
^X next word     ^E  end of line  ^KC copy   ^O word< ^@  insert   FILE
SEARCH           ^KU top of file  ^KW file   ^J >line SPELL        ^KE edit
^KF find text    ^KV end of file  ^KY delete ^_ undo  ^[N word     ^KR insert
^L  find next    ^KL to line No.  ^K/ filter ^^ redo  ^[L file     ^KD save
```

Como se ve, las combinaciones de teclas están organizadas por categorías, se estudiarán de arriba a abajo y de izquierda a derecha.

- CURSOR: relacionadas con el movimiento del cursor en las cuatro direcciones posibles. Generalmente funcionan también las teclas de mo-

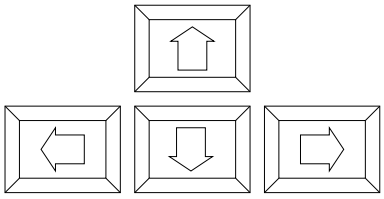

Figura 5.4: Teclas de movimiento del cursor.

vimiento del cursor de la terminal o del pc. Se reconocen por estar dispuestas como se muestra en la figura 5.4.

- SEARCH: relacionadas con la búsqueda de una cadena de caracteres en el texto.
- GOTO: sirven para moverse por el texto. "prev. screen" se mueve una página hacia atrás, "next screen" se mueve una página hacia adelante. De las demás combinaciones, una de las más útiles es la que se mueve a una línea determinada.
- BLOCK: se utilizan para mover o copiar bloques de texto. El funcionamiento de estas teclas es ligeramente distinto de las combinaciones que permiten hacer lo mismo en los sistemas operativos de Microsoft, de modo que se explicará con mayor detalle. En primer lugar se marca el principio (`^KB`) y después el final (`^KK`) del bloque que se va a mover o copiar, se destacará con los colores invertidos. Después se posiciona el cursor en la parte del texto a donde se quiere mover o copiar el texto. Si se copia (`^KC`), el bloque original permanece inalterado, si se mueve (`^KM`), desaparece de su ubicación original y aparece en la nueva posición.
- DELETE: de estas teclas, se destaca como deshacer cambios, otras operaciones pueden realizarse con las teclas habituales. La combinación de teclas para deshacer cambios (revertir una operación de movimiento o copia de texto, por ejemplo) es `^_`. Como el guión bajo se obtiene a su vez mediante una combinación de teclas (por ejemplo en un PC con shift o mayúsculas y la tecla del guión), será necesario mantener pulsadas tres teclas simultáneamente, en primer lugar ctrl, sin soltarla pulsar mayúsculas y de nuevo sin soltar ninguna de ellas, pulsar la tecla del guión.

- EXIT: sirven para salir del programa, guardando los cambios ^KX o descartándolos ^C.
- FILE: se usan para abrir un fichero, insertar un fichero en otro o para guardarlo sin salir del editor.

Capítulo 6

Comandos básicos de Unix.

Hasta ahora se han visto sólo tres comandos de Unix, `g++`, `cd`, `mkdir`. De estos tres, en realidad `g++` es una aplicación que puede estar instalada o no, pero no es un comando. A continuación se verán algunos comandos de unix que serán de utilidad para la realización de las prácticas.

- `passwd`: sirve para cambiar la clave de acceso. De forma automática se ejecutará la primera vez que entre en la cuenta. Es necesario conocer la clave actual para poder cambiarla. Se pedirá que introduzca la nueva dos veces, para dificultar el que la teclee mal de forma inadvertida y que posteriormente no pueda acceder al ordenador. Además tendrá que seleccionar una de cuatro alternativas:

```
centauro.aulario.uniovi.es> passwd
Old password:
Last successful password change for zz999666: Wed May 21 17:07:25 CEST 2003
Last unsuccessful password change for zz999666: NEVER


Do you want (choose one option only):

   1  Pronounceable passwords generated for you
   2  A string of characters generated for you
   3  A string of letters generated for you
   4  To pick your password

Select ONE item by number: 4

You have selected:
To pick your password

New password:
```

```
Re-enter new password:
centauro.aulario.uniovi.es>
```

La opción 1 genera automáticamente una clave pronunciable, la 2 una cadena de caracteres aleatoria, la 3 una cadena de letras aleatoria y la 4 permite introducir la clave que deseemos. Usted seguramente seleccionará la opción 4, aún así, el administrador del sistema puede configurarlo de modo que si la clave es demasiado corta o sencilla, se pida una distinta. Recuerde que en ningún momento se verán las claves, las tecleará a ciegas.

- `ls`: muestra el contenido de un directorio, los ficheros y los subdirectorios. Posee varias opciones útiles, como por ejemplo -l, que muestra mas información que simplemente el nombre, como tamaño, fecha de creación, etc. La opción -a muestra los archivos ocultos, aquellos que empiezan por un '.'.

```
centauro.aulario.uniovi.es> ls
a.out      hola.cpp   hola.cpp~  otro
centauro.aulario.uniovi.es> ls -a
.          ..         a.out      hola.cpp   hola.cpp~  otro
centauro.aulario.uniovi.es> ls -l
total 32
-rwxr-xr-x   1 zz999666   practicas     22288 Sep 18 13:15 a.out
-rw-r--r--   1 zz999666   practicas        69 Sep 18 13:16 hola.cpp
-rw-r--r--   1 zz999666   practicas        68 Sep 18 13:14 hola.cpp~
drwxr-xr-x   2 zz999666   practicas      8192 Sep 18 16:32 otro
centauro.aulario.uniovi.es> ls -al
total 48
drwxr-xr-x   3 zz999666   practicas      8192 Sep 18 16:32 .
drwxr-x--x  17 zz999666   practicas      8192 Sep 18 13:04 ..
-rwxr-xr-x   1 zz999666   practicas     22288 Sep 18 13:15 a.out
-rw-r--r--   1 zz999666   practicas        69 Sep 18 13:16 hola.cpp
-rw-r--r--   1 zz999666   practicas        68 Sep 18 13:14 hola.cpp~
drwxr-xr-x   2 zz999666   practicas      8192 Sep 18 16:32 otro
```

El texto anterior muestra varios ejemplos del uso de `ls`, en primer lugar sin opciones, se muestra el nombre de los ficheros, sin más. En segundo lugar, con la opción -a, se muestran dos archivos ocultos '.' y '..'. En tercer lugar se utiliza la opción -l, de modo que se muestra información adicional, entre otras cosas el propietario del archivo (`zz999666`), la fecha de creación, el tamaño en bytes,... .En la columna más a la izquierda distinguimos los archivos de los directorios, los directorios

tienen una 'd' en esa columna. Finalmente se utilizan ambas opciones simultáneamente, de modo que se muestran los archivos ocultos, también con información adicional. En este momento es oportuno explicar la noción de "comodín". Un comodín es un carácter que representa un carácter cualquiera o una combinación cualquiera de longitud arbitraria. El símbolo '?' representa cualquier carácter, el '*' a cualquier combinación de caracteres. Se emplean con los comandos de Unix para aplicarlos a varios ficheros al mismo tiempo o a aquellos cuyo nombre coincide con un patrón, que se especifica usando comodines.

```
centauro.aulario.uniovi.es> ls
a.out      hola.cpp   hola.cpp~  otro
centauro.aulario.uniovi.es> ls h*
hola.cpp   hola.cpp~
centauro.aulario.uniovi.es> ls ?.*
a.out
centauro.aulario.uniovi.es> ls *.cpp
hola.cpp
centauro.aulario.uniovi.es>
```

En el ejemplo anterior ls muestra en primer lugar todos los archivos. En segundo lugar, aquellos que comienzan por 'h'. En tercer lugar los que comienzan por una sola letra, después llevan un '.' y finalizan en cualquier cosa. Finalmente muestra los que terminan en ".cpp".

- `rm`: sirve para borrar ficheros y/o directorios.

```
centauro.aulario.uniovi.es> ls
a.out      hola.cpp   hola.cpp~  otro
centauro.aulario.uniovi.es> ls
a.out      hola.cpp   hola.cpp~  otro
centauro.aulario.uniovi.es> rm a.out
centauro.aulario.uniovi.es> ls
hola.cpp   hola.cpp~  otro
centauro.aulario.uniovi.es> rm -r otro
centauro.aulario.uniovi.es> ls
hola.cpp   hola.cpp~
centauro.aulario.uniovi.es>
```

En el ejemplo anterior se muestra como borrar ficheros y directorios. Para borrar ficheros se escribe el nombre del comando, un espacio, el nombre del fichero y se pulsa return. Para borrar directorios el proceso es similar, pero hay que añadir la opción -r. Los comodines pueden usarse con `rm`, pero con precaución, ya que pueden borrarse más cosas de las que se desean.

- `cp`: copia el contenido de un fichero en otro. Su sintaxis es `cp origen destino`, donde origen es el fichero inicial y destino el fichero en donde se va a hacer una copia del primero.

- `mv`: cambia el nombre a un fichero. Su sintaxis es `mv nombre_inicial nombre_final`.

- `file`: da información sobre el contenido de un archivo, su sintaxis es `file nombre`.

```
centauro.aulario.uniovi.es> file otro
otro:   directory
centauro.aulario.uniovi.es> file *
a.out:  COFF format alpha dynamically linked, demand paged executable or object
 module not stripped - version 3.13-14
hola.cpp:       c program text
hola.cpp~:      c program text
otro:   directory
centauro.aulario.uniovi.es>
```

  Como se observa en el ejemplo anterior, se puede usar con comodines.

- `more`: es un paginador, se suele utilizar en combinación con otro comando para encauzar su salida y mostrarla pantalla a pantalla. Para ir viendo toda la salida de ese otro comando, se pulsará la barra espaciadora. Por ejemplo, en un directorio con muchos ficheros, si se ejecuta `ls`, se verán sólo los últimos según el orden alfabético. Si se encauza la salida con more, se verán los nombres de los ficheros uno a uno. La sintaxis es `comando|more`, el carácter '—' separa a more del comando cuya salida se quiere encauzar.

- `pwd`: muestra el path absoluto del directorio actual. Por ejemplo:

```
centauro.aulario.uniovi.es> pwd
/practicas/zz999666
centauro.aulario.uniovi.es>
```

  En este caso se muestra el directorio por defecto al entrar en la cuenta, denominado también directorio "home" o simplemente "home".

La anterior es una lista exigua de los comandos disponibles, lo estrictamente necesario para poder realizar las prácticas. En la siguiente sección se mostrará cómo utilizar el compilador instalado en el ordenador de las prácticas.

Capítulo 7

Sesión introductoria con el compilador.

Un programa en C++ es un texto que está escrito siguiendo la sintaxis de ese lenguaje. El ordenador no puede procesar directamente la ordenes escritas en ese lenguaje. Previamente es necesario traducirlas al lenguaje que entiende directamente el ordenador. Esto no es necesario que lo realize el programador, sino que existe un programa diseñado específicamente para realizar esta tarea, el compilador. Este programa toma un fichero de texto que contiene un programa en C++ y, si este es correcto, produce un programa ejecutable que realiza las operaciones descritas en ese fichero de texto. A este fichero se le conoce como "fichero fuente" o "fuente". Si el programa incumple alguna de las reglas del lenguaje, el compilador no produce un programa ejecutable, en lugar de eso, informa al programador de los errores cometidos.
En las secciones anteriores se utilizaba un programa muy sencillo para ilustrar el funcionamiento de un editor de texto. El programa es el siguiente:

```
#include<iostream>
using namespace std;
int main()
{
cout<<"Hola"<<endl;
}
```

El programa anterior es correcto, de modo que si se procesa con el compilador, producirá un programa ejecutable:

```
centauro.aulario.uniovi.es> ls
```

```
hola.cpp
centauro.aulario.uniovi.es> g++ hola.cpp
centauro.aulario.uniovi.es> ls
a.out     hola.cpp
centauro.aulario.uniovi.es> a.out
Hola
centauro.aulario.uniovi.es>
```

En el ejemplo anterior se muestra el contenido de un directorio en donde sólo hay un fichero conteniendo el fuente del programa en C++. Si se llama al compilador, como el programa es correcto, produce un ejecutable, por defecto lo llama `a.out`. Obsérvese como la orden `ls` muestra ahora el fichero ejecutable generado por el compilador. Para ejecutar ese programa, se escribe su nombre y se pulsa return. El programa muestra por la pantalla "`Hola`", para que el prompt aparezca en la siguiente línea, es necesario incluir `endl` al final de lo que se desea que aparezca en la pantalla.
Si existiesen errores en el programa, no se hubiera producido un ejecutable y se informaría de los errores al programador. Evidentemente el análisis de los errores que realiza el compilador es linitado, una vez que se encuentra el primero de los errores, el análisis se ve perturbado por este, de modo que puede que informa de errores que no existen y que vienen desencadenados por el anterior. Por ello es conveniente corregir los errores comenzando por el que ocurre antes en el programa.
Imagine que eliminamos el carácter # en la primera línea del programa y guardamos el fichero (puede hacer esto con joe o con el editor de texto del CDE si está en una terminal X). Si borramos el ejecutable que generó antes el compilador e intentamos compilar el programa de nuevo, se produce lo siguiente:

```
centauro.aulario.uniovi.es> ls
a.out     hola.cpp
centauro.aulario.uniovi.es> rm a.out
centauro.aulario.uniovi.es> ls
hola.cpp
centauro.aulario.uniovi.es> g++ hola.cpp
hola.cpp:1: ‘iostream’ was not declared in this scope
hola.cpp:2: syntax error before ‘using’
hola.cpp: In function ‘int main()’:
hola.cpp:5: ‘cout’ undeclared (first use this function)
hola.cpp:5: (Each undeclared identifier is reported only once for each function
    it appears in.)
hola.cpp:5: ‘endl’ undeclared (first use this function)
centauro.aulario.uniovi.es> ls
hola.cpp
centauro.aulario.uniovi.es>
```

Observe la secuencia de órdenes y el resultado. En primer lugar, se borra el fichero `a.out` que había generado la compilación anterior, por ello la orden `ls` sólo muestra el fichero fuente. A continuación se intenta compilar el programa, como es incorrecto, se producen errores. Si a continuación se ejecuta de nuevo la orden `ls`, se comprueba que no se ha generado el ejecutable.
Observe la salida del compilador, aporta información del fichero en donde se producen los errores y de la línea en la que se producen. En la primera línea existe un error, el compilador interpreta la cadena `iostream` como un identificador no declarado, lo cual es incorrecto. A continuación, en la línea dos, se informa de que hay un error de sintaxis *antes*. Esto es muy frecuente, hay que acostumbrarse a leer completamente la descripción del error, ya que de lo contrario es fácil creer equivocadamente que el error esta en esa línea y no en la anterior. Como el error se produce en la primera línea, en donde se incluyen el archivo de cabeceras en donde está definido `cout` o `endl`, aparecen los errores correspondientes en la línea en donde se utilizan, en cuanto se añada el carácter # que se suprimió, estos errores desaparecerán.
Es importante destacar que si no se guarda el fichero modificado y no se compila de nuevo el programa, los cambios que se realicen no tendrán efecto en el programa ejecutable. Imagine que se corrige el error del programa anterior y que no se compila el fichero fuente, no se producirá ningún programa ejecutable, si existía uno procedente de una compilación anterior, no se sobreescribirá.
Suponga que se ha ejecutado el compilador sobre el programa anterior corregido y que se ha producido un ejecutable. Si se modifica el fichero fuente como aparece a continuación,

```
#include<iostream>
using namespace std;
int main()
{
cout<<"Hola"<<endl;
//falta un ; en la siguiente linea
//este programa no compila
cout<<"Adios"<<endl
}
```

al intentar compilarlo se producirán errores y no se obtendrá un programa ejecutable. Nótese como una vez más el error se detecta en una línea determinada pero se produce en la anterior.

```
centauro.aulario.uniovi.es> g++ hola.cpp
hola.cpp: In function ‘int main()’:
hola.cpp:7: parse error before ‘}’ token
```

El que haya habido errores en la compilación no es óbice para que el ejecutable que se obtuvo en la compilación anterior siga presente:

```
centauro.aulario.uniovi.es> a.out
Hola
```

Si se examina el contenido del directorio con `ls -l`, se observará que la fecha o la hora de modificación del ejecutable es *anterior* a la del fichero fuente, por lo tanto el ejecutable no ha podido obtenerse del fuente que existe en este momento.

```
centauro.aulario.uniovi.es> ls -l
total 23
-rwxr-xr-x   1 zz999666   practicas     22400 Sep 19 17:44 a.out
-rw-r--r--   1 zz999666   practicas        95 Sep 19 17:45 hola.cpp
centauro.aulario.uniovi.es>
```

Por todo ello, la secuencia habitual en el proceso de escritura y compilación de un programa es la que se muestra en el diagrama de la figura 7.1. Se edita el fichero fuente, se guardan los cambios y se compila. Si hay errores se vuelve a editar y se repite el proceso. Si no hay errores, se prueba el programa. Si el programa realliza las operaciones especificadas, se ha terminado, si no hay que editar el fichero de nuevo y reiniciar el proceso. Fíjese en que después de editar el fichero fuente es necesario compilar de nuevo y que sólo se prueba el programa si no ha habido errores.

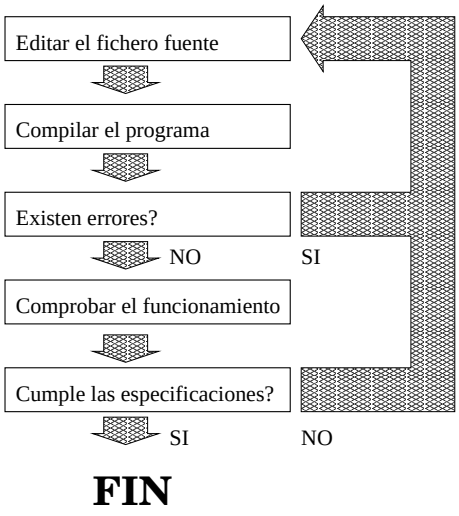

Figura 7.1: Proceso de escritura de un programa.