Appendice

B

Elementi di informatica

B.1 Struttura e funzionamento di un personal computer

I personal computer sono piccoli calcolatori che possono offrire un'ampia gamma di potenze di calcolo, ma che si definiscono «personal» perché tipicamente sono utilizzati da un unico utente per volta. Per questo motivo, i personal di solito non prevedono limitazioni all'accesso nella forma di definizione di identificativi (*username*) e di parole segrete (*password*).

I personal computer sono in generale composti almeno da: uno schermo, una tastiera e un mouse (che servono per comunicare con la macchina), un supporto elettronico per l'immagazzinamento dei dati (hard disk) e il computer vero e proprio, che comprende: un processore (cpu: *central processing unit*), la memoria principale (RAM: *random access memory*) e la memoria di supporto (ROM: *read only memory*).

Altri componenti del personal possono essere alcune «periferiche»: una stampante, un lettore di dischetti magnetici (unità floppy disk, per l'inserimento e il trasporto di dati tra personal diversi), un lettore di compact disc o DVD, uno scanner (per trasferire testi o immagini da supporto cartaceo a supporto elettronico), un microfono per la digitalizzazione o casse per l'emissione di suoni, macchine fotografiche o telecamere per l'acquisizione di immagini, ecc.

I personal computer possono essere connessi tra loro in reti che consentono la condivisione sia di dati sia di periferiche. Ogni personal computer deve essere dotato di un «sistema operativo», cioè del software necessario al proprio funzionamento e alla gestione delle periferiche a esso connesse. Il sistema operativo è parte integrante della macchina, che senza di esso non riesce neanche ad accendersi. Dotato di un opportuno sistema operativo, il personal offre spesso anche un'interfaccia grafica che consente l'uso della macchina anche a persone non esperte (per esempio tutti i sistemi operativi macintosh e windows).

Sul personal computer si «caricano» ed eseguono, oltre al già citato software di sistema, anche software applicativi di vario tipo (per esempio programmi per la gestione dei testi, di banche dati; programmi di grafica e altri).

B.1.1 Calcolatori multiutente

In generale, i grossi calcolatori multiutente sono estremamente veloci nell'effettuare calcoli, nella gestione dell'INPUT e dell'OUTPUT. Sono cioè spesso molto più veloci degli essere umani che ne richiedono le prestazioni. Per cui gli utenti di un calcolatore lavorano spesso «dividendo il tempo» del calcolatore (*time-sharing*) con altri utenti. Nella maggior parte dei casi, ciò non comporta nessun rallentamento nel lavoro.

Esistono diversi tipi di calcolatori in grado di supportare la multi-utenza, per esempio i calcolatori tipo alpha (Digital), SUN, Silicon Graphics, IBM e altri, con sistema operativo (OS) unix. Un altro sistema, dal punto di vista operativo molto simile allo unix, ma, diversamente da questo, totalmente gratuito, è il sistema operativo linux.

È bene sapere che esistono anche altri sistemi operativi (che però trascendono gli scopi e i limiti di questa appendice) e che esistono calcolatori in grado di funzionare con diversi tipi di sistema operativo.

B.1.2 Struttura di un calcolatore multiutente

I principali componenti di un calcolatore sono l'**unità centrale di elaborazione** (o **cpu**, **c**entral **p**rocessing **u**nit) e la memoria RAM. La cpu si identifica con il processore, che esegue le operazioni aritmetiche o di altro tipo sui dati presenti nella memoria RAM. La potenza di un processore (la sua velocità di calcolo) è legata alla sua frequenza e si misura in MHz (MegaHertz). La RAM è una memoria di lettura-scrittura ad accesso casuale (cioè il processore può accedervi in qualsiasi punto) ed è volatile perché, spegnendo il computer, si azzera. La sua capacità si misura in byte (vedere Glossario). Altre componenti essenziali del calcolatore sono i dispositivi di ingresso (input: terminali, dischi, unità di lettura, rete ecc.) e uscita (output: terminali, stampanti, rete ecc.).

B.1.3 Reti di computer

Ogni calcolatore multiutente o personal computer può essere collegato anche a più calcolatori e la rete che ne risulta è gestita dai sistemi operativi dei calcolatori che la utilizzano. Ognuno di questi calcolatori può quindi essere visto come nodo della rete (net o web, ragnatela).

Le informazioni possono passare da un nodo all'altro in «pacchetti» utilizzando il protocollo TCP/IP (vedere Glossario) e la velocità di trasferimento di tali «pacchetti» dipende dal tipo di cavo che i dati percorrono (esistono cavi «lenti» e cavi a fibra ottica estremamente più veloci), dallo stato di «affollamento» dei segmenti di rete che si attraversano (la rete viene utilizzata contemporaneamente da tutti gli utenti che ne fanno richiesta; non si perdono dati, ma il loro trasferimento può risultarne rallentato), dalla buona gestione della rete stessa.

Le reti di computer vengono utilizzate per tutte le comunicazioni tra calcolatori, tra cui ricordiamo: la posta elettronica (**e-mail**), il trasferimento di dati e programmi (**ftp**: **f**ile **t**ransfer **p**rotocol), l'utilizzo di un calcolatore *remoto* (con programmi come **telnet**), la navigazione in cerca di programmi, banche dati o altro (per esempio con un **browser** di rete, in modalità grafica).

B.1.4 Cos'è un sistema operativo

Il sistema operativo è il programma che gestisce il calcolatore ed è costituito da sottoprogrammi o *moduli*, che coordinano e svolgono tutte le attività del computer.

La parte *fisica* della macchina, i suoi circuiti, le memorie, le periferiche, si chiama *hardware* (ferramenta). Su questa materia, del tutto passiva, opera il *software* (*soft*, leggero; *hard*, pesante): il programma. Sono i programmi che consentono l'uso e il funzionamento delle parti fisiche della macchina calcolatore.

La dimensione di un sistema operativo è dell'ordine delle centinaia di migliaia di istruzioni (per la descrizione di cos'è un programma e cosa sono le istruzioni di un programma si rimanda più avanti).

Il sistema operativo si occupa di molte diverse funzioni contemporaneamente: registra i nomi e gli indirizzi dei file (vedere oltre) sui dischi e controlla ogni comando che parte dalle tastiere dei terminali o dei personal connessi al computer; gestisce la RAM in modo che ogni processo in corso possa averne *allocata* una porzione sen-

za sovrapposizioni o perdite di tempo; risponde alle richieste degli utenti; controlla l'uso delle risorse; esegue regolari backup (salvataggi) di dati e altre funzioni di manutenzione; gestisce la rete di cui il calcolatore è nodo e così via.

Il limite tra hardware e software è talvolta così vago che le stesse funzioni che in un calcolatore vengono svolte mediante l'hardware, in un altro sono svolte dal software. La linea di demarcazione tra hardware e software è ormai più che altro una vera e propria nuova regione, denominata *firmware*.

I fattori che determinano le scelte tra hardware e software sono essenzialmente due:

1. il tempo (in genere una soluzione software è più lenta di una hardware);
2. il costo (un circuito che realizza un'istruzione complessa è più costoso di un software che sfrutta generici circuiti elementari).

La visualizzazione della struttura di una proteina e la simulazione del suo movimento con programmi di grafica molecolare può avvenire sfruttando un opportuno software che calcola traslazioni e rotazioni. Per esempio, esistono programmi come RasMol e SwissPDBViewer che funzionano proprio come appena descritto (Capitolo 7).

Alternativamente esistono macchine (per esempio Silicon Graphics) dotate di un processore supplementare costruito apposta per ruotare e traslare velocemente oggetti composti da molti atomi. Tali macchine sono più costose, perché hanno un sofisticato *pezzo* in più. Ma sono le migliori per la grafica molecolare.

B.1.5 Cos'è un «file»

Un file è una porzione di spazio su disco in cui è possibile immagazzinare dell'informazione.

Esistono e possono venire usati diversi tipi di file (di dati, di comandi, eseguibili in formato ASCII o binario), come si vedrà in seguito.

Il più semplice tipo di file è quello di testo, che può venire creato, modificato, salvato su disco o cancellato mediante l'uso degli appropriati comandi del sistema operativo.

Ogni file è definito da un *nome* e da un *contenuto.*

ESEMPIO: il file identificato dal nome *sequenza* contiene la sequenza: ATGGTAACGCTGACT; il file **lettera** contiene una lettera di saluti; il file *search* contiene un programma che effettua una ricerca, e così via.

Nella maggior parte dei casi, i nomi dei file possono essere scelti dagli utenti. Ci sono però delle regole da rispettare nel dare i nomi ai file e queste regole dipendono dal sistema operativo con cui si lavora.

B.1.6 Cos'è una «directory»

Una directory è una specie di «cartella personale» in cui ogni utente conserva i propri file e utilizza i programmi che lo interessano.

Entrando in un calcolatore con sistema operativo unix o linux facendo uso del proprio username e della propria password, ci si trova in una directory identificata dal proprio username.

ESEMPIO: l'utente **bio1** del calcolatore di un centro di calcolo, al proprio ingresso nel calcolatore si troverà automaticamente nella directory **bio1**; l'utente Rossi entrerà invece nella directory rossi e così via.

La directory denominata con il nome dell'*account* è la directory principale dell'utente (e può essere indicata come ~nome_directory). Ogni utente può generare altre directory (dette *sub-directory*), di livello gerarchicamente inferiore, per suddividere in modo razionale e organico i propri programmi e i propri dati. Si possono creare *alberi* comprendenti diversi livelli di sottodirectory.

ESEMPIO: la subdirectory **sequenze** dell'utente bio1, si chiamerà ~bio1/sequenze; nella directory ~bio1/sequenze potranno essere presenti a loro volta altre directory (~bio1/sequenze/nucleic e ~bio1/sequenze/amino) e così via.

Le directory e le subdirectory sono particolari tipi di file, che contengono i nomi e gli indirizzi dei file in esse contenuti, ma che non è possibile editare.

ESEMPIO: la directory ~bio1/sequenze sarà presente nella directory ~bio1 come file sequenze; le subdirectory di «sequenze» non saranno visibili nella directory ~bio1 mentre saranno invece presenti come file **nucleic** e **amino** nella directory ~bio1/sequenze.

B.1.7 Cos'è un programma per calcolatore

Un programma per calcolatore è un *oggetto* informatico in cui si introducono dati (l'INPUT) che il programma elabora e restituisce modificati nell'OUTPUT.

I programmi vengono scritti utilizzando particolari linguaggi (i linguaggi di programmazione), quali il Fortran, il C, il Pascal, il COBOL e altri. Consistono in una successione di «istruzioni» che consentono l'acquisizione dell'input, la sua elaborazione e la generazione dell'output.

ESEMPIO: il programma SOMMA potrebbe contenere le seguenti istruzioni:
- leggi numero 1
- leggi numero 2
- numero 3 = numero 1 + numero 2
- stampa numero 3
- stop

In questo esempio, l'input è rappresentato da *numero 1* e *numero 2*, l'output da *numero 3*.

I programmi vengono scritti in un determinato linguaggio di programmazione (di solito detto **di alto livello**, vedere Glossario) e altri programmi (i **compilatori**, moduli del sistema operativo) li traducono in linguaggio macchina e li trasformano in programmi eseguibili che generalmente funzionano solo con il sistema operativo per cui sono stati compilati. Quindi, mentre il linguaggio ad alto livello è trasportabile da un sistema all'altro, i programmi eseguibili sono in grado di funzionare solo su certe macchine o su certi sistemi operativi.

ESEMPIO: il programma scritto in fortran è contenuto nel file *somma.f*. Il compilatore del Fortran legge il file *somma.f* e genera il file *somma,* che potrà essere **eseguito** dal calcolatore. Notare che il file somma.f è un normale file di testo, mentre il file somma è un complicato file binario, completamente illegibile per l'uomo, ma di facile esecuzione per il calcolatore.

Alternativamente alcuni linguaggi di alto livello (detti di scripting) invece che essere compilati possono essere **interpretati**, come per esempio il linguaggio **Perl** illustrato nell'Appendice C. In tal caso, invece di un modulo compilatore, si dovrà utilizzare un modulo **interprete**, che è in grado di leggere direttamente il file di testo contenente il programma e di convertirlo immediatamente in linguaggio macchina, eseguendone le istruzioni.

Tipicamente i linguaggi compilati sono eseguiti più velocemente dei linguaggi interpretati, dato che sono già tradotti in linguaggio macchina; inoltre, una volta compilati, essi hanno il vantaggio di non richiedere altri programmi se non il sistema operativo. Al contrario, i linguaggi interpretati richiedono un interprete installato sul computer dove devono essere eseguiti e, per le ragioni dette sopra, hanno tempi di esecuzioni generalmente più lunghi. Tuttavia, anche i linguaggi interpretati presentano diversi vantaggi; per esempio la semplicità di esecuzione, specialmente quando sono necessarie continue modifiche al programma.

B.2 Introduzione ai sistemi operativi *linux* e *unix*

B.2.1 Unix e linux

Linux è uno dei sistemi operativi più diffusi. Appartiene alla famiglia dei sistemi operativi *unix* e si distingue per il fatto di essere stato sviluppato (e di essere tuttora mantenuto) da un gruppo internazionale di informatici che hanno voluto metterlo gratuitamente a disposizione di tutti. Per quanto riguarda gli utenti del calcolatore, si ricorda che il sistema operativo è l'interfaccia con la macchina, il linguaggio con cui si comunica alla macchina le operazioni e i comandi che si vogliono siano eseguiti.

Il sistema operativo unix/linux è piuttosto semplice da usare anche se i comandi sono talvolta poco facilmente memorizzabili. Tipicamente si dà un comando digitandolo sulla tastiera e premendo il tasto <invio>: il sistema operativo risponde eseguendo il comando oppure mostrando un messaggio di errore se non riesce a interpretare il comando ricevuto. Anche per i sistemi operativi unix, e in particolare per Linux, è sempre più frequente l'uso di interfacce grafiche che rendono l'uso del computer più facile e immediato.

Un utente speciale nei calcolatori multiutente è il cosiddetto **system manager** (username: **root**) che è di solito un tecnico specializzato in elettronica e/o informatica, che si occupa della gestione e della manutenzione del calcolatore. Il system manager, tra le altre cose, si occupa di aggiornare il sistema operativo e di compiere operazioni complesse, quali per esempio: aggiungere o cancellare utenti, acquisire e mantenere il software applicativo, aggiornare le banche dati e presiedere alla risoluzione di qualsiasi problema tecnico gli venga segnalato dagli utenti che gli sono affidati.

Generalmente ai calcolatori multiutente si accede in modo remoto, utilizzando un programma come *telnet* dal proprio personal computer. Per potersi connettere, telnet ha bisogno del numero IP del calcolatore remoto o del suo nome dominio (vedere Glossario). Inoltre è ovviamente necessario disporre di uno username e della corrispondente password.

Da qualche anno si sta diffondendo l'uso di linux anche su personal computer. A questo proposito sono disponibili molti pacchetti integrati di facile installazione, che oltre al sistema operativo contengono numerosi programmi applicativi.

Una volta connessi al computer si riceve la «schermata di benvenuto», con la richiesta di identificazione, tramite:

- username <invio>
- password <invio>

Lo username è un identificativo dell'utente (di solito il suo cognome o una sigla seguita da un numero); la password è la chiave elettronica che, unita allo username, consente l'accesso al calcolatore. Quando si digita la propria password, i caratteri non vengono mostrati sullo schermo, a ulteriore protezione dell'accesso *controllato* al calcolatore.

A questo punto, si è NEL calcolatore, nella propria directory principale e non si è soli, c'è anche il sistema operativo!

Entrare nel calcolatore si dice anche: fare login.

Per uscire è necessario dare il comando exit.

B.2.2 Alcuni comandi fondamentali di unix e linux

■ **ls** (lista) è un comando che serve per avere la lista dei propri file. La lista viene data in ordine alfabetico. È possibile richiedere la lista dei file accompagnata da altre informazioni che si considerino utili, per esempio: la dimensione dei file, la data in cui sono stati creati o modificati per l'ultima volta, le protezioni. Il comando ls è anche utile per avere una lista dei file che abbiano in comune una porzione del loro nome (utilizzo dell'asterisco).

Nota L'asterisco (*) è molto utile e molto usato per indicare una qualunque stringa di caratteri di qualunque lunghezza.

ESEMPIO: ls *.seq dà la lista di tutti i file presenti nella directory il cui nome finisca per «.seq».

Per esempio con il comando **ls citta*.dat** si otterrà la lista dei file il cui nome comincia per «citta» e termina per «.dat», tra i file potranno essere compresi quelli come: citta.dat e cittadini.dat).

ESEMPIO: la maggior parte dei comandi può essere utilizzata con estensioni che aggiungono funzionalità ai comandi stessi. Per chiedere una lista dei propri file con informazioni riguardanti sia la loro dimensione sia la data della loro creazione o ultima modifica, si può utilizzare il comando **ls -l**.

■ **mkdir** Per creare la subdirectory sequenze nella directory bio1, è necessario dare il comando: *mkdir sequenze*

■ **cd** per muoversi da una directory all'altra, si può dare il comando: cd *nomedir* cd è l'acronimo di *change directory*.

ESEMPIO: il comando cd ~bio1/sequenze/nucleic sposterà l'utente dalla directory in cui si trova alla directory bio1/sequenze/nucleic. Nello unix, tutte le directory sono «figlie» della directory principale «/» (root). Al di sotto della dir principale di root, ci sono altre directory, che di solito identificano i dischi rigidi del sistema (/usr, /usr1, /usr2 e così via). In uno dei dischi rigidi, c'è di solito la directory people (/usr1/people), in cui vengono create dal system manager le directory degli utenti della macchina. Per passare alla directory di ordine immediatamente superiore si deve digitare «cd .. <invio>», mentre per tornare alla directory principale si può digitare il comando: cd <invio>

■ **pwd** è il comando che informa sulla directory in cui si trova l'utente (*print working directory*).

ESEMPIO: il comando pwd dato subito dopo il login restituisce il nome alla directory principale dell'utente.
axdid.mat.uniroma2.it> pwd <invio>
/users/citteric
dove *axdid.mat.uniroma2.it*> è di solito il «prompt» del sistema, ovvero la stringa di caratteri che il calcolatore mostra per indicare che è pronto a ricevere dei comandi.

■ **more** il comando *more nomefile* consente di visualizzare a terminale il contenuto di un file. Se il file è più lungo di una schermata del terminale, viene visualizzato a schermate successive, che si ottengono in sequenza premendo la barra degli spazi. Il more può essere interrotto digitando la lettera *q* (quit).

■ **tail** il comando *tail nomefile* consente di visualizzare a terminale solo la parte finale (la «coda») di un file.

■ **cp** per duplicare un file cambiandogli nome, il comando è: *cp file1 file2* che copia il contenuto del file **file1** nel file **file2**, che crea contestualmente. Prima del comando esisteva solo il **file1**, dopo il comando esistono **file1** e **file2** e il loro contenuto è identico.

■ **rm** è possibile cancellare i propri *file* col comando: *rm file1* (*remove*). Per proteggere i propri file da eventuali distrazioni, si può impostare il comando in modo che chieda la conferma a ogni richiesta di cancellazione (vedere oltre, il Paragrafo sui nomi dei file e le note sulle protezioni).

■ **rmdir** è il comando per cancellare una directory. Per poter cancellare una directory è però comunque necessario che siano già stati cancellati tutti i file e directory file in essa contenuti.

■ **mv** Se si vuole modificare il nome di un file, mantenendo intatto il suo contenuto, si può usare il comando: *mv file1 file2* (*move)* che modifica solo il nome del file. Pri-

ma del comando esisteva solo il **file1**, dopo il comando solo il **file2**, ma il contenuto è invariato.

■ **who** è un comando che dà informazioni sul numero e sull'username degli utenti che stanno lavorando sul calcolatore interattivamente. È anche possibile vedere con quale programma ognuno sta impegnando il calcolatore.

■ **ps** è un comando che dà informazioni sui programmi con cui l'utente sta impegnando il calcolatore. Con l'estensione **-ef**, fornisce informazioni su tutti i processi attivi (per esempio: i moduli del sistema operativo, le procedure «in *batch*», vedere oltre).

■ **pico** è un programma di editing molto semplice che consente di creare, modificare e salvare (in altre parole **editare**) file. La sintassi di pico è riportata nella parte bassa dello schermo, all'apertura di un file.

pico nomefile crea il file nomefile e dà la possibilità di inserirvi dei dati. Una volta che i dati siano stati inseriti, bisogna premere i tasti ctrl+x (come per scrivere una M maiuscola si devono premere contemporaneamente i tasti **shift** e quello della **M**); questo farà comparire nella parte sinistra in basso dello schermo la dicitura:

Save modified buffer (ANSWERING «No» WILL DESTROY CHANGES)?

cui si può rispondere con **Y** (se si vuole salvare sul disco quanto è stato inserito nel file) o **N** (se si vogliono perdere le modifiche fatte).

Per quanto riguarda le funzioni di pico, è disponibile una schermata di aiuto SOLO quando si sta già editando un file: la schermata di aiuto si ottiene premendo i tasti ctrl+g.

■ **chmod** è un comando che serve a modificare il modo di accesso dei file. Ogni file può essere *leggibile (r = read), scrivibile (w = write)* o *eseguibile (x = execute)* e ognuna di queste *modalità d'accesso* può essere consentita all'utente proprietario del file *(u = user)*, al gruppo (g = *group*) o ad altri utenti (o = *other*) con diritto d'accesso alla directory in cui il file si trova. Le informazioni sui modi d'accesso ai singoli file si ottengono con il comando **ls -l** e si modificano con il comando **chmod (chi) (modalità) nomefile**.

ESEMPIO: (in grassetto sono riportati i comandi al sistema operativo e in caratteri normali le risposte del sistema)

```
$ ls -l
total 16
-rw-r--r--    1 manuela  users           9 Nov 21 09:18 ilmioscript.pl
-rw-r--r--    1 manuela  users          13 Nov 21 09:17 seq1.seq
-rw-r--r--    1 manuela  users          15 Nov 21 09:17 seq2.seq
-rw-r--r--    1 manuela  users          10 Nov 21 09:17 seq3.seq
$ chmod +x ilmioscript.pl
$ ls -l
total 16
-rwxr-xr-x    1 manuela  users           9 Nov 21 09:18 ilmioscript.pl
-rw-r--r--    1 manuela  users          13 Nov 21 09:17 seq1.seq
-rw-r--r--    1 manuela  users          15 Nov 21 09:17 seq2.seq
-rw-r--r--    1 manuela  users          10 Nov 21 09:17 seq3.seq
```

il file ilmioscript.pl è diventato *eseguibile* per tutti gli utenti.

■ **man** è importantissimo comunque ricordare che esiste sempre a disposizione dell'utente il cosiddetto *help on-line*, ovvero un manuale dei comandi unix a disposizione mediante il comando *man* (manuale). È possibile richiedere le pagine del manuale riguardanti un solo comando alla volta.

ESEMPIO: man ls darà le pagine del manuale che spiegano l'utilizzo del comando ls, con esempi e definizione delle estensioni.

I comandi del sistema operativo si possono utilizzare in due modi diversi:
1. **interattivo**: quando si lavora in modo interattivo, i comandi si digitano a terminale e l'esecuzione di ogni comando deve terminare prima di lanciare il successivo;
2. in **background**: lavorare in background significa lanciare una procedura (un programma o una lista di comandi) che viene *sottomessa* al sistema come un processo separato. Dopo aver sottomesso una procedura in background, si può continuare a usare il proprio terminale o personal interattivamente.
   Solitamente le procedure lanciate in *background (o batch)* godono di una priorità inferiore a quella assegnata ai processi interattivi: è pertanto buona norma, ogniqualvolta le circostanze lo consentano, di lanciare in batch programmi che richiedono lunghi tempi di esecuzione. Questo consente al sistema una migliore gestione delle risorse e minori tempi di attesa per gli altri utenti.

B.2.3 I file in linux e unix

In qualsiasi sistema operativo ogni file ha un nome che lo identifica. L'utente è sempre libero di scegliere un nome mnemonico che lo aiuti a ricordare quale file contiene quale informazione. Sul nome mnemonico esistono poche limitazioni: non può cominciare con un carattere che non sia una lettera o un numero, non può contenere asterischi o spazi (più precisamente può contenere tali caratteri, ma la gestione di questi file è poi resa complicata dalla natura stessa dei comandi).

Modificare un file e poi salvarlo sull'hard disk, in unix e in linux significa perdere la versione precedente dello stesso file.

ESEMPIO: Un utente crea il file **lettera** con il comando pico. Nella lettera scrive «cara mamma», poi salva il file sul disco con ctrl+x e Y. Nella directory si troverà il file **lettera**.

Se ripete il comando pico, aggiunge un paio di righe e di nuovo salva con ctrl+x e Y, nella directory sarà poi ancora presente il file **lettera** nella sua seconda versione. La prima versione della lettera non può più essere recuperata, a meno che, al momento di salvare la seconda versione, non si decida di assegnarle un nuovo nome (per esempio *lettera2*).

Quando ci si trova in una determinata directory, il nome di un file è unico. In realtà il nome **completo** di un file è costituito dal nome con cui è stato definito fino a ora, preceduto da tutto l'albero della directory che lo contiene: /nomedisco/people/nomeutente/nomefile se si è nella directory principale dell'utente, dove nomedisco identifica il disco del sistema su cui il file risiede; people è la directory che ospita gli utenti del sistema e nomeutente è lo username dell'utente.

Il nome completo del file, unito al nome del computer su cui si trova, lo identifica in modo univoco nel mondo.

Ogni volta che si parte del nome a sinistra del nomemnemonico non sono esplicitate, il sistema assume per default che il nome del calcolatore sia quello **locale**, che il nome del disco sia quello su cui si sta lavorando e che pure il nome della directory sia quello della directory in cui si trova l'utente.

Il concetto di **default** sta a indicare che il calcolatore in determinate circostanze è in grado di *sottintendere* determinate informazioni, **a meno che** non gliene vengano date di **diverse** in forma esplicita.

B.2.4 Segnali

Oltre ai comandi, esistono anche dei *segnali* in grado di eseguire particolari funzioni da tastiera.

Si possono suddividere in un piccolo numero di categorie: i segnali che interrompono i comandi (tipo I); che richiamano comandi (R); che controllano la posizione del cursore (C); che controllano lo scorrere dei dati sullo schermo del terminale (D).

Nella breve lista che segue, ctrl + N si deve interpretare come il contemporaneo premere dei tasti *ctrl* e della lettera *N* (come per scrivere una M maiuscola si devono premere contemporaneamente i tasti *shift* e quello della *M*).

Tipo	Segnale	Funzione
tipo I	ctrl+C	cancella l'esecuzione dei comandi e programmi
tipo R	freccia in su	richiama i comandi precedentemente eseguiti
	freccia in giù	mostra la riga successiva nei comandi precedentemente eseguiti
tipo C	backspace	cancella il carattere che precede la posizione del cursore