Lezione 3: Redirezione dell'I/O e completamento dei comandi

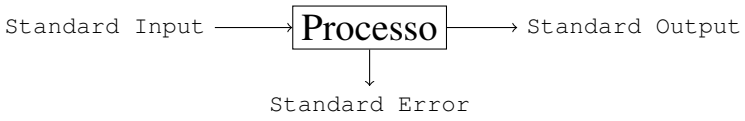
Laboratorio di Elementi di Architettura e Sistemi Operativi

12 Marzo 2013

Parte 1: Redirezione dell'Input/Output

I canali di Input/Output

- Ogni processo di un sistema operativo Unix ha tre canali associati:



- Normalmente, lo `Standard Input` è associato alla tastiera, mentre lo `Standard Output` e lo `Standard Error` sono associati allo schermo.
- Ogni canale può essere rediretto:
  - su file
  - su altro canale
  - verso un altro processo

Redirezione su file

- Standard input da file `comando < file`
- Standard output in file (se il file esiste già viene sovrascritto) `comando > file`
- Standard output aggiunto in coda al file `comando >> file`
- Standard error in file (se il file esiste già viene sovrascritto) `comando 2> file`
- Standard output e standard error su due file diversi `comando > fileout 2> fileerr`
- Standard output e standard error sullo stesso file `comando > file 2>&1` `comando &> file`

Esempi

- redirezione dell'output:

```
$ echo ciao a tutti > file
$ more file
ciao a tutti
```

- redirezione dell'output con append:

```
$ echo ciao a tutti >> file
$ more file
ciao a tutti
ciao a tutti
```

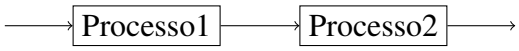
- Il comando `wc` fornisce numero di linee, parole, caratteri:

```
$ wc < progetto.txt
21 42 77
```

- Redirezione dei messaggi d'errore:

```
$ man 2 passwd 2> errori.txt
$ less errori.txt
No entry for passwd in section 2 of the manual.
```

La "pipe": redirezione tra processi



- Standard output del primo comando nello standard input del secondo:
  `comando1 | comando2`
- Permette di combinare più comandi semplici per eseguire compiti complessi

I comandi filtro

- I *filtri* sono una particolare classe di comandi che possiedono i seguenti requisiti:
  - leggono l'input dallo standard input,
  - effettuano delle operazioni sull'input ricevuto,
  - inviano il risultato delle operazioni allo standard output.
- I filtri sono degli ottimi strumenti per costruire pipeline che svolgano compiti complessi.
- Alcuni esempi di filtri che abbiamo già incontrato:
  `cat, head, tail, more, less, find, grep, wc`

Esempi

- La sequenza di comandi

```
$ ls /usr/bin > listafile.txt
$ wc -w listafile.txt
459
```

  ha lo stesso effetto della pipeline:

```
$ ls /usr/bin | wc -w
459
```

  I comandi ls e wc sono eseguiti in parallelo: l'output di `ls` è letto da `wc` mano a mano che viene prodotto.
- Visualizzare l'output di `ls` ricorsivo pagina per pagina `$ ls -R | more`
- Elencare i file con permessi `rw-r--r--`: `$ ls -l | grep "rw-r--r--"`

Parte 2: Storia dei comandi e completamento automatico

La storia dei comandi

La *storia dei comandi* è uno strumento fornito dalla shell che consente di evitare all'utente di digitare più volte gli stessi comandi:

- la storia memorizza gli ultimi 500 comandi inseriti dall'utente;
- la storia viene salvata nel file `.bash_history` al momento del logout (e riletta al momento del login);
- il comando `history` consente di visualizzare la lista dei comandi:

```
$ history | tail -4
512 ls -al
513 cd /etc
514 more passwd
515 history | tail -4
```

- ogni riga prodotta dal comando `history` è detta evento ed è preceduta dal *numero dell'evento*.
- Conoscendo il numero dell'evento che vogliamo ripetere, possiamo eseguirlo usando il metacarattere `!`:

```
$ !515
history | tail -4
513 cd /etc
514 more passwd
515 history | tail -4
516 history | tail -4
```

- Se l'evento è l'ultimo della lista è sufficiente usare `!!`:

```
$ !!
history | tail -4
514 more passwd
515 history | tail -4
516 history | tail -4
517 history | tail -4
```

- È anche possibile eseguire delle *ricerche testuali* per individuare l'evento a cui siamo interessati:

```
$ !ls
ls -al
total 491
drwxr-xr-x  16 root root      0 Oct 15 21:35 .
drwxr-xr-x  16 root root      0 Oct 15 21:35 ..
-rw-r--r--   1 root  root 87515 Jul 10 04:28 Muttrc
drwxr-xr-x   2 root root      0 Oct 15 21:27 WindowMaker
...
```

  In questo modo la shell cerca a partire dall'ultimo evento, procedendo a ritroso, un comando che inizi con `ls`.
- Racchiudendo con `?` la stringa da ricercare, la shell controllerà che quest'ultima appaia in un punto qualsiasi del comando:

```
$ !?ls?
```

Editing dei comandi

La shell mette a disposizione dell'utente dei semplici comandi di editing per facilitare la ripetizione degli eventi:

- utilizzando i tasti cursore:
  - con la freccia verso l'alto ↑ si scorre la storia dei comandi a ritroso (un passo alla volta) facendo apparire al prompt il comando corrispondente all'evento;
  - analogamente con la freccia verso il basso ↓ si scorre la storia nella direzione degli eventi più recenti.
  - le frecce sinistra ← e destra → consentono di spostare il cursore sulla linea di comando verso il punto che si vuole editare;
- le combinazioni di tasti `Ctrl-A` e `Ctrl-E` spostano il cursore, rispettivamente all'inizio ed alla fine della linea di comando;
- il tasto `Backspace` consente di cancellare il carattere alla sinistra del cursore.

Completamento dei comandi

Una caratteristica molto utile della shell è la sua abilità di tentare di completare ciò che stiamo digitando al prompt dei comandi:

```
$ pass<Tab>
```

- La pressione del tasto `<Tab>` fa in modo che la shell cerchi un comando che inizi con `pass`.
- Siccome l'unica scelta possibile è il comando `passwd`, questo sarà riportato automaticamente nel prompt.
- Se i caratteri digitati non permettono di identificare il comando, avviene quanto segue:
  - viene prodotto un suono di avvertimento al momento della pressione del tasto `<Tab>`;
  - alla seconda pressione del tasto `<Tab>` la shell visualizza una lista delle possibili alternative;
  - digitando ulteriori caratteri, alla successiva pressione del tasto `<Tab>`, la lista diminuirà fino ad individuare un unico comando.
- Oltre a poter completare i comandi, la shell bash può anche completare i nomi dei file:

```
$ tail -2 /etc/p<Tab><Tab>
passwd printcap profile
$tail -2 /etc/pa<Tab><Invio>
bianchi:fjKppCZxEvouc:500:500::/home/bianchi:/bin/bash
rossi:Yt1a4ffkGr02:501:500::/home/rossi:/bin/bash
```

- In questo caso alla prima doppia pressione del tasto `<Tab>`, la shell presenta tre possibili alternative.
- Digitando una `a` e premendo il tasto `<Tab>`, la shell può determinare in modo univoco il completamento del nome del file.