Unix - Shell dei comandi
Sostituzioni

Prima dell'esecuzione di un comando, esso viene scandito (assieme ad i suoi parametri) e vengono operate delle sostituzioni

1. Sostituzioni delle variabili/parametri: ad ogni variabile viene sostituito il suo valore

Ad es.:

x=12

echo $x

Produce 12

Unix - Shell dei comandi
Sostituzioni

Prima dell'esecuzione di un comando, esso viene scandito (assieme ad i suoi parametri) e vengono operate delle sostituzioni

2. Sostituzioni dei comandi: oogni comando compreso tra `` viene eseguito e sostituito col risultato

Ad es.:

echo `pwd`

Stampa il direttorio corrente

Unix - Shell dei comandi
Sostituzioni

Prima dell'esecuzione di un comando, esso viene scandito (assieme ad i suoi parametri) e vengono operate delle sostituzioni

3. Sostituzioni dei nomi dei file: i caratteri *, ?, [ ] vengono sostituiti coi nomi dei file secondo il pattern matching sul file system

Ad es.:

ls *

Elenca i file del direttorio corrente

Unix - Shell dei comandi
Sostituzioni

E’ possibile decidere quali sostituzioni applicare e quando…

- ' <comando> ‘ : sospende tutte le sostituzioni
- " <comando> “ : solo sostituzioni di tipo 1 e 2

Comandi relativi all'espansione:

' (quote) nessuna espansione (non 1, 2, 3)

" (doublequote) solo sostituzioni 1 e 2 (non la 3)

```
y=3
echo ' * e $y ' # produce * e $y

echo " * e $y " # produce * e 3
Il processore esegue una sola espansione
y=3
x='$y'
echo $x # stampa $y
```

Esempi

ls [a-z]*[0-9]*[A-Z]

ls *[!0-9]*

ls [a-z,A-Z,0-9][a-zA-Z0-9]

ESECUZIONE di COMANDI in SHELL
In UNIX (quasi) ogni comando viene eseguito da un nuovo shell
lo **shell attivo** mette in esecuzione un **secondo shell**
Il **secondo shell**
- esegue le sostituzioni
- ricerca il comando
- esegue il comando

Lo **shell padre** attende il completamento dell'esecuzione del sottoshell (comportamento **sincrono**)

AMBIENTE DI SHELL:
insieme di variabili di shell
Per esempio, una variabile registra il **direttorio corrente**
Inoltre, ogni utente specifica come fare la ricerca dei comandi nei vari direttori del file system
la variabile **PATH** indica i direttori in cui ricercare ogni comando da eseguire
la variabile **HOME** indica il direttorio di accesso iniziale
PATH=$PATH:/usr/mio/direttorio

AMBIENTE di un FILE COMANDI

Si noti l' ambiente dei file comandi composto di:

- **variabili predefinite**
- **direttorio corrente** - insieme di variabili usate e variate dall''utente

L'ambiente è accessibile agli **shell figli**, che possono accrescere il tutto e aggiungere nuove variabili

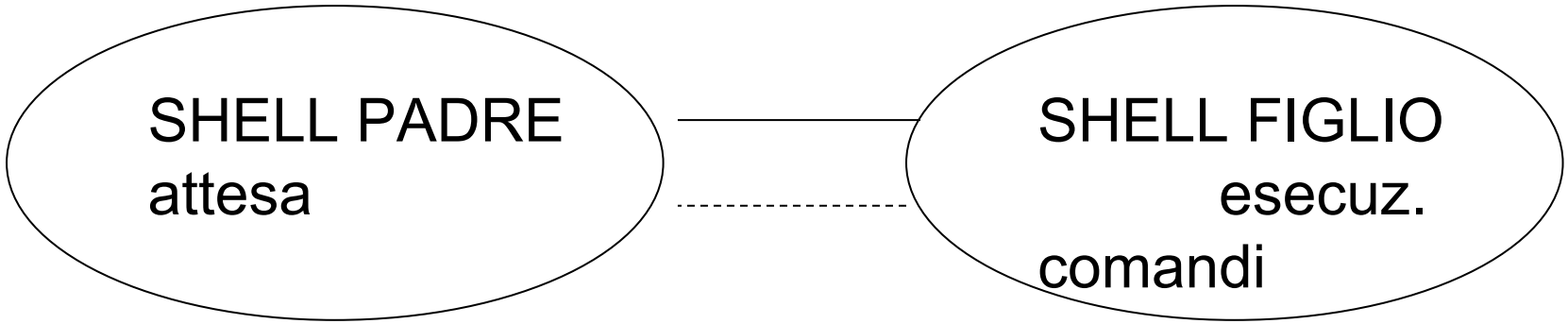
solo l'ambiente predefinito iniziale viene **copiato** ai figli **con i valori iniziali.**

In caso di aggiunta di **nuove variabili o di modifica di variabili predefinite**, i nuovi valori sono **copiati solo se** le **variabili** sono **esplicitamente esportate**

L'ambiente padre è preservato in **ogni caso**

OGNI COMANDO è ESEGUITO da un NUOVO e DISTINTO SHELL

che ottiene una copia (parziale) dell'ambiente del padre ma non può modificarla



export PATH, varnuova, varmia

Non esiste modo di **riportare variabili** (o altro) dal **figlio al padre** (UNIX suggerisce la **condivisione di file**)

ESECUZIONE IN PARALLELO &

Lo shell padre aspetta il completamento del figlio
esecuzione in **foreground (sincrona)**
È possibile non aspettare il figlio, ma proseguire
esecuzione in **background (asincrona)**
Lo shell invocante è immediatamente attivo
<comando> [<argomenti>] **&**
Process ID: <number>
I**dentificatore del processo** in background
Quando termina lo shell padre viene avvisato
I processi in background si eliminano con
kill <PID> ; # o anche; **kill -9** <PID>
I processi in background mandano l'output sulla console
si possono mescolare i messaggi dei diversi processi
Devono prendere l'input da file (**ridirezione**)
altrimenti ci sarebbe confusione sull'input
(INTERFERENZA)

Unix-Shell di Comandi- Es1.

Si realizzi un file di script che permetta di copiare un filesorgente in un file destinazione. Se questo esiste già, si chiede conferma della sovrascrittura

Invocazione:

copia filesorg filedest

Soluzione

```
case $# in
     0 | 1) echo Usage $0 filesorg filedest
              exit 1;;
     2) if test ! -f $1
          then echo $1 non esiste; exit 2
          fi;;
esac
if test -f $2
then echo il file $2 esiste: vuoi ricoprirlo?
        read answer
        case $answer in
        n* | N* ) exit;;
        esac
fi
if test -d $2
       then echo $2 "è un direttorio"; exit 3
else if cp $1 $2
         then echo ho copiato il file $1 nel file $2
        else echo errore sulla copia
       fi
fi
```

Unix – Shell di Comandi – Es 2.

Si scriva un file di script che permetta di muovere su rischiesta tutti I file passati come argomento in un direttorio specificato. Se l'operazione di spostamento ha successo si deve stampare a video "si spostato il file in dir" altrimenti "ci sono problemi per dir"

Invocazione:

moveat dir file1 file2 …fileN

Soluzione

```
case $# in
  0|1) echo almeno due argomenti; exit 1;;
esac
if test -d $1
then
  for i in $* # per tutti gli argomenti
  do
      if test $i != $1
      then
          echo $i in $1 ?
          read risposta
          if test $risposta = "si" -o $risposta = "s"
          then
            if mv $i $1
# il risultato della mv pilota la alternativa
            then echo si $1/$i
            else echo problemi per $1
            fi
         fi
      fi
done
fi
```

Unix – Shell di Comandi – Es 3.

Si scriva un file di script che permetta di inserire in un file (il cui nome è dato come parametro), senza distruggerlo, stringhe lette da input se soddisfano le seguenti condizioni:

- Stringhe con secondo carattere compreso tra (b-g) e che terminano con a
- Stringhe con secondo carattere pari a s e penultimo carattere compreso tra (0-9)
- Stringhe con terzo carattere pari a oe ultimo carattere compreso tra (2-9)

Soluzioni

```
while echo "vuoi finire(si/no)?"; read fine
test $fine != si
do
    echo inserisci stringa
    read a
    echo $a
    case $a in
    ?[b-g]*a) echo OK1; echo $a >> $1;;
    ?s*[0-9]?) echo OK2; echo $a >> $1;;
    ??o*[2-9]) echo OK3; echo $a >> $1;;
    *) echo NO case;;
    esac
done
```