Programmazione di Sistema Unix

(20 Dicembre 2019)

Si progetti un'applicazione multiprocesso in C che aiuti uno studente universitario a consultare le recensioni che gli studenti hanno assegnato agli insegnamenti del proprio corso di laurea, al fine di selezionare gli insegnamenti opzionali da inserire nel proprio piano di studi. L'applicazione deve presentare la seguente interfaccia:

Recensioni_insegnamento archivio_CdL

dove ***archivio_CdL*** è il nome assoluto del file che archivia tutte le recensioni degli insegnamenti di un CdL. Ciascuna riga di tale file contiene tutte le informazioni relative a una specifica recensione, con (in quest'ordine) la data di pubblicazione in formato YYYYMMDD (quindi, ad esempio, la data di oggi verrebbe rappresentata con la stringa "20191220"), il nome dell'insegnamento, il voto (da 0 a 10) di recensione, il commento incluso nella recensione, ecc.

L'applicazione deve essere composta da un processo iniziale **P0** che, prima di tutto, si deve interfacciare con lo studente, da cui riceve (via terminale) il nome dell'insegnamento di interesse (es., "Sistemi Operativi", "Elettronica Digitale", ecc.). Per ogni richiesta ricevuta dallo studente, il processo **P0** crea due processi figli, **P1** e **P2**. **P1** deve selezionare tutte le recensioni relative all'insegnamento di interesse dello studente e inviarle quindi a **P2**, che deve organizzarle in ordine di data decrescente e inviare quindi i risultati a **P0** che provvede infine alla visualizzazione.

P0 continua a rispondere alle richieste dello studente fino all'inserimento della stringa "fine" o alla pressione dei tasti Ctrl-C. In entrambi i casi, il processo **P0** deve stampare il numero di richieste di servizio gestite prima di terminare.

```c
// recensioni_insegnamento archivio_Cdl
#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
#include <unistd.h>
#include <signal.h>
#include <string.h>
#include <sys/wait.h>

int richieste_servite = 0;

void sigint_handler(int signo) {
   printf("Numero di richieste servite:%d\n", richieste_servite);
   exit(0);
}

int main(int argc, char **argv) {

   int fd, p1, p2, p1p2[2], p2p0[2], status;
   char cdl[80];
   char results[1024];

   // controllo argomenti
   if (argc != 2) {
       perror("errore numero parametri \n");
       exit(1);
   }
   // controllo path assoluta
   if (argv[1][0] != '/') {
       perror("errore path non assoluto \n");
       /* se si vuole passare argomenti, possibile uso di
       *  fprintf(stderr, "%s non ha una path assoluta\n", argv[1]); */
       exit(2);
   }

   // controllo esistenza file
   fd = open(argv[1], O_RDONLY);
   if (fd < 0) {
       perror("Errore apertura file \n");
       exit(3);
   }
   close(fd);

   signal(SIGINT, sigint_handler);  // installo gestore per il segnale
   printf("\n Inserire insegnamento di interesse:\n");
   memset(cdl, 0, sizeof(cdl));
   scanf("%s", cdl);
```

```
while (strcmp(cdl, "fine")!=0) { // ciclo per la gestione delle richieste

    if (pipe(p1p2) < 0) {   // pipe per la comunicazione P1 --> P2
        perror("Errore creazione pipe p1p2\n");
        exit(4);
    }

    p1 = fork();
    if (p1 < 0) {
        perror("Errore fork p1\n");
        exit(5);
    }
    if (p1 == 0) {       //Codice processo figlio P1
        close(p1p2[0]);  // chiudo canale non necessario

        close(1);        // redirezione stdout
        dup(p1p2[1]);
        close(p1p2[1]);
        printf("P1 cerca %s\n", cdl);
        execlp("grep", "grep", cdl, argv[1], (char *)0);
        perror("Errore in grep P1\n");
        exit(6);
    }
    if (pipe(p2p0) < 0) {
        perror("Errore creazione pipe p2p0\n");
        exit(7);
    }
    p2 = fork();
    if (p2 < 0) {
        perror("Errore fork p2\n");
        exit(8);
    }
    if (p2 == 0) {  // Codice processo figlio P2
        close(p1p2[1]);
        close(p2p0[0]);

        close(0);    // redirezione stdin
        dup(p1p2[0]);
        close(p1p2[0]);

        close(1);    // redirezione stdout
        dup(p2p0[1]);
        close(p2p0[1]);
        execlp("sort", "sort", "-r", "-n",(char *)0);
        perror("Errore in sort P2\n");
        exit(9);
    }
```

```
        // Siamo di nuovo nel padre P0
        close(p1p2[0]);
        close(p1p2[1]);
        close(p2p0[1]);
        memset(results, 0, sizeof(results));
        while (read(p2p0[0], results, sizeof(results)) > 0) {
            write(1, results, strlen(results) + 1);
            memset(results, 0, sizeof(results));
        }

        close(p2p0[0]);
        wait(&status);
        wait(&status);

        richieste_servite++;

        printf("\n Inserire insegnamento di interesse:\n");
        memset(cdl, 0, sizeof(cdl));
        scanf("%s", cdl);
    }

    printf("Totale richieste servite: %d\n", richieste_servite);
    return 0;
}
```

Programmazione Shell

Si scriva uno script shell per aiutare il responsabile di un'azienda a controllare la quantità dei prodotti presenti nei vari magazzini al fine di redigere l'inventario di fine anno. In particolare, lo script deve presentare la seguente interfaccia:

controlla_prodotto dir nome_prodotto

dove ***dir*** è un nome assoluto di directory e ***nome_prodotto*** è una stringa rappresentante un nome di un prodotto venduto dall'azienda (per esempio "lampada da muro", "lampadario X", "lampadario Y", etc.) e stoccato nei suoi magazzini.

Si supponga che le informazioni sulla disponibilità delle varie merci siano conservate nella directory ***dir*** all'interno del file system della macchina su cui dovrà eseguire lo script ***controlla_prodotto***. Le informazioni sono salvate in file di testo (quindi con estensione .txt) che si trovano all'interno della directory ***dir*** o in una delle sue sottocartelle. Ciascun file contiene tutte le informazioni relative a uno specifico magazzino e ha il nome della città dove si trova il magazzino. Ciascuna riga di tali file conterrà tutte le informazioni relative a una singola tipologia di prodotto presente nel magazzino, con (in quest'ordine) nome prodotto, nome produttore, paese di provenienza, quantità stoccata, peso, collocazione, data di produzione ecc.

Lo script ***controlla_prodotto*** deve quindi esplorare ricorsivamente la directory ***dir*** e tutte le relative sottodirectory per analizzare il contenuto dei magazzini, selezionare le informazioni relative al prodotto di interesse, estrarre la sola informazione sulla quantità disponibile e sommarla a quella presente negli altri magazzini. Il dato complessivo e il nome del prodotto devono essere aggiunti in modalità append a un file di risultati, con il nome di "inventario_2019.txt", posizionato all'interno della home directory dell'utente.

Prima di terminare, il file comandi deve anche stampare a video il nome del magazzino (ovverosia del file corrispondente) in cui è presente la maggior quantità del prodotto di interesse tra tutti i magazzini analizzati.

File: controlla_prodotto.sh

```
#!/usr/bin/env sh
# controlla_prodotto.sh dir nome_prodotto

# controllo parametri
if test $# -lt 2
then
   echo "$0 <dir> <nome_prodotto>"
   exit 1
fi

# controllo path assoluta
case $1 in
/*);;
*) echo "$1 deve essere un nome assoluto di directory"; exit 2;;
esac

if test ! -d "$1"
then
   echo "$1 deve essere una directory"
   exit 3
fi

# aggiornamento PATH
PATH=`pwd`:$PATH
export PATH

echo 0 > /tmp/.counter
> /tmp/.max_mag_name
echo 0 > /tmp/.max_mag_val

sh controlla_prodotto_ric.sh "$1" "$2"

# scrivo il risultato di inventario nel file e a video
echo "$2 `cat /tmp/.counter`" | tee -a $HOME/inventario_2019.txt

echo "Magazzino con maggior quantita di $2: `cat /tmp/.max_mag_name`"

rm -f /tmp/.counter
rm -f /tmp/.max_mag_name
rm -f /tmp/.max_mag_val
```

File: controlla_prodotto_ric.sh

```
#!/usr/bin/env sh
 cd "$1"

 for i in *.txt
 do
   if test -f "$i" -a -r "$i"; then
      info_prod=`grep "$2" "$i"`

      # se il prodotto e' presente vado a sommare
      # la quantita' nel magazzino i con quella
      # precedentemente stoccata
      if test -n "$info_prod"
      then
         qta=`echo "$info_prod" | cut -f 4 -d ,`

         # aumento il mio contatore memorizzato su file
         qta_inventariata=`cat /tmp/.counter`

         # aggiorno la quantita'
         echo `expr $qta + $qta_inventariata` > /tmp/.counter

         # controllo massimo
         if test $qta -gt `cat /tmp/.max_mag_val`
         then
            echo $qta > /tmp/.max_mag_val
            echo "`pwd`/$i" > /tmp/.max_mag_name
         fi
      fi
   fi
 done

 for d in *
 do
   if test -d "$d" -a -x "$d"
   then
      sh controlla_prodotto_ric.sh "$d" "$2"
   fi
 done
```