

## Laboratorio di Sistemi Operativi 2011-2012: esercitazione finale.

### Estensione per Indirizzo SR Vecchio Ordinamento

(AA 2009/2010 e precedenti)

#### Un sistema per la gestione delle prenotazioni

Le prenotazioni ospedaliere avvengono tramite centri di prenotazioni unificate, che raccolgono tutte le richieste degli utenti e organizzano l'agenda delle visite dei vari reparti.

L'esercitazione consiste nella progettazione e implementazione di un sistema di gestione delle prenotazioni per il centro di prenotazioni di un ospedale.

Il centro di prenotazioni deve gestire prenotazioni per diverse prestazioni (per esempio *oculistica*, *ortopedia*, etc.). Ogni richiesta ha una *priorità* -specificata dall'utente dalla linea di comando-, che comporterà un maggiore/minore tempo di attesa (e costo associato).

#### Architettura generale del sistema

Il sistema risultante implementa il paradigma client-server: i client rappresentano gli utenti, il server il centro di prenotazioni.

La soluzione deve essere basata sull'utilizzo di code multiple:

- una coda (chiamata *MSG\_Q\_\_main\_bus*) utilizzata dai client per contattare il server e dal server per inviare le prenotazioni;
- una coda per ciascun servizio che i client sono potenzialmente interessati a prenotare. In particolare è necessario fornire prenotazioni almeno per i seguenti reparti: *oculistica*, *ortopedia*, *radiologia* (rispettivamente *MSG\_Q\_\_oculistica*, *MSG\_Q\_\_radiologia* e *MSG\_Q\_\_ortopedia*).

Il server riceve le richieste avanzate dai client; forka e affida la gestione di ogni richiesta a un figlio. La gestione della richiesta comporta le seguenti operazioni:

- attribuzione di un numero di sequenza a ciascuna richiesta pervenuta (per mezzo dell'uso della mutua esclusione, due richieste che arrivano concorrentemente devono essere messe in sequenza correttamente);
- calcolo del numero di turno, basato sia sul numero di sequenza, sia sulla priorità eventualmente espressa dall'utente; il numero di turno deve essere calcolato con una formula che combini ordine di arrivo e priorità, come per esempio:

$$\text{numero\_turno} = \text{numero\_di\_sequenza} - \text{priorità}$$

in questo caso sarà necessario gestire il caso dei primissimi numeri di sequenza, che combinati con le priorità potrebbero dare luogo a numeri di turno minori di zero 0.

- calcolo del costo della prestazione: il costo delle prestazioni fornite dai reparti (chiamiamolo *costo\_visita*) è memorizzato nel file di configurazione per ciascuno dei reparti; poiché tale costo dipende anche dalla priorità, una possibile modalità di calcolo della prestazione è

$$\text{costo\_prestazione} = \text{costo\_visita} + (\text{priorità} * \text{costo\_priorità})$$

L'incremento di costo associato alla priorità, *costo\_priorità*, è memorizzato nel file di configurazione.

- Il server figlio invia su socket al client la fattura per la prestazione richiesta; a tale scopo, il client deve aprire un socket server con nome il proprio *PID* e estensione *.sock* (per esempio *80580.sock*). Il server si connette a tale socket e invia la fattura.  
La fattura deve contenere: destinatario (il *PID* del client...), data e ora dell'evasione della richiesta, le informazioni relative alla prestazione richiesta e alla priorità, al costo calcolato.

## Terminazione

Per terminare l'esecuzione del processo server principale implementare un gestore di segnali che intercetti il segnale *SIGQUIT*; tale gestore provvederà alla deallocazione della coda *MSG\_Q\_\_main\_bus* e di tutti gli oggetti *IPC* utilizzati nell'esecuzione, oltre che alla terminazione propriamente detta.

## Client

Il client invia la propria richiesta sulla coda del server, e attende che gli venga inviata la risposta; dopo la ricezione, visualizza la notifica contenente il numero di prenotazione ed il costo associato.

Quindi, riceve la fattura via socket (si veda sopra) e la stampa a video.

Nel caso venga eseguito con argomenti, il client prende tre interi, il primo dei quali indica il numero di figli da crearsi, il secondo il numero di servizio (si vedano le costanti *OPHTHALMOLOGY*, etc. nel file header del progetto), e il terzo il livello di priorità. Per esempio, data una invocazione di tipo

*./client 4 ORTHOPEDICS NOHURRY*

si *intende* che i quattro figli saranno tutti interessati a prestazioni ortopediche con priorità *NOHURRY* (si veda nuovamente il file *header\_proj.h*).

*Diversamente*, il client viene eseguito senza argomenti, e per impostazione predefinita effettua una sola prenotazione di una prestazione in *radiologia*, con priorità *ILLWAIT* (si veda nuovamente il file header del progetto).

## Simulatore dell'erogatore delle prestazioni

*Progettare* e implementare un ulteriore programma, (si chiamerà *attuatore.c*), che simuli l'effettiva erogazione delle prestazioni sanitarie richieste.

L'attuatore riceve da linea di comando il nome del reparto su cui deve operare e crea **due thread**, *THR\_QUEUE* e *THR\_INPUT*.

Il thread *THR\_QUEUE* preleva i messaggi (secondo l'ordine indotto dal *numero\_di\_turno*) dalla coda del reparto man mano che arrivano e li aggiunge a una lista in memoria contenente gli appuntamenti.

Il thread *THR\_INPUT* riceve da tastiera dei comandi per:

- mostrare la lista;
- ricercare e visualizzare la prenotazione per un dato processo;
- ordinare in modo crescente sulla base del numero di processo e visualizzare la lista ordinata.

### **Grado di finitura del software, compilazione e dettagli vari**

Le funzioni fondamentali devono essere documentate con chiarezza (quelle *non* commentate dovrebbero essere auto-esplicative): occorre specificare che input prendono, qual è la loro funzione, se ci sono delle precondizioni, etc. .

Il codice deve essere suddiviso in file sulla base di una articolazione funzionale. È necessario costruire un *Makefile* per la compilazione dei file costituenti il progetto; i file devono essere compilati con le seguenti opzioni:

*-Wall -pedantic*

L'elaborato finale deve contenere anche una *documentazione* in cui si illustra come eseguire i programmi (argomenti, dipendenze, etc.) che collettivamente costituiscono il *sistema per la gestione delle prenotazioni*.