Politecnico di Milano
FACOLTÀ DI INGEGNERIA DELL'INFORMAZIONE

Corso di Laboratorio Software

Laboratory of Operating Systems and Software Design

Period: 2005-2006
prof. Giovanni AGOSTA
prof. William FORNACIARI

Written part of the exam (09.03.06)

SURNAME (readable) .....NAME (readable).....

Matricola.....Signature.....

Indirizzo di studio (if relevant): .....

Mandatory: write all the above data so that they are **READABLE**

Q1	Q2	Q3	Q4	TOT

NOTE PER LO SVOLGIMENTO/Notes

Si raccomanda di essere **sintetici** (al fine di evitare inutili perdite di tempo in trattazioni generiche e poco significative ai fini della valutazione finale) e **ordinati** allo scopo di migliorare l'interpretazione da parte dei correttori. I temi proposti debbono essere risolti utilizzando unicamente lo spazio lasciato al termine del testo di ogni quesito, il retro delle pagine o, eventualmente, utilizzando lo spazio finale. Verranno corretti **SOLO** i fogli componenti il tema d'esame.

One of the goals of the student is to present the discussion in a **concise** and **readable** way, to simplify the evaluation phase, and using only the stapled sheets: only such sheets will be considered.

È vietato consultare testi o appunti di qualunque genere così come interagire con i vicini. Chiunque sia trovato in possesso di documentazione relativa al corso, anche se non strettamente attinente al tema d'esame, vedrà annullata la prova.

It is not allowed to use any textbooks or note, as well as to interact with the other students. The owning of such type of material (even if not strictly relevant with the questions of the current exam) or, in general, the offending the above rule, will invalidate the written exam.

Non è consentito uscire durante la prima mezz'ora, il compito deve essere comunque riconsegnato, anche in caso di ritiro.

La presenza allo scritto (anche non consegnando) comporta la rinuncia a eventuali voti precedenti.

It is not allowed to exit during first 30 minutes of the exam and, in any case, the stapled sheets cannot be removed from the room. Note that a copy of the exam, with some solutions, will be made available on the web.
The simple presence to the written exam implies to give up to any of the previous evaluations.

Question Q1

Suggested time for the exercise: 15 minutes. Il tempo consigliato per svolgere l'esercizio è di 15 min.

Nel contesto dell'algoritmo del banchiere con un singolo tipo di risors, analizzare i seguenti stati e dire se sono safe o unsafe.
Se sono safe, dimostrare il modo in cui tutti i processi possono completare l'esecuzione.
Altrimenti, mostrare come potrebbe avvenire il deadlock.

1) Stato A/State A:

Processes	Max(P_i)	Loan(P_i)	Claim(P_i)
P_1	4	1	3
P_2	6	4	2
P_3	8	5	3
P_4	2	0	2
		a=1	

Dove Max(P_i) e' il massimo numero di risorse necessarie al completamento di P_i, Loan(P_i) e' il numero di risorse attualmente allocate a P_i, e Claim(P_i) e' Max(P_i)-Loan(P_i).
Il numero delle risorse disponibili e' "a".

Where Max(P_i) is the maximum number of resources needed for the completion of P_i, Loan(P_i) is the number of resources currently held by P_i, and Claim(P_i) = Max(P_i) – Loan(P_i).
The number of currently available resources is "a".

2) Stato B/State B:

Processes	Max(P_i)	Loan(P_i)	Claim(P_i)
P_1	8	4	4
P_2	8	3	5
P_3	8	5	3
		a=2	

Question Q2

Suggested time for the exercise: 15 minutes. Il tempo consigliato per svolgere l'esercizio è di 15 min.

Discutere i meccanismi di soluzione hardware per il problema della mutua esclusione, ed i vantaggi e svantaggi rispetto ai meccanismi software.

Discuss the hardware solutions to the mutual exclusion problem, as well as the advantages and disadvantages of these methods with respect to the software solutions.

Quesito D3

Suggested time for the exercise: 15 minutes. Il tempo consigliato per svolgere l'esercizio è di 15 min.

Descrivere l'impiego delle primitive della famiglia exec (exec, excve, execp...), discutendo le differenze fra le varie funzioni e proponendo un esempio di uso di execve.

Describe the usage of the primitives of the exec family (exec, execve, execp...), with a discussion of the differences among the various functions of this family, and give an example of use of the execve primitive.

Question Q4

Suggested time for the exercise: 45 minutes. Il tempo consigliato per svolgere l'esercizio è di 45 min.

Supponiamo di voler realizzare un sistema di allocazione dei processi in un sistema distribuito, che esegua l'allocazione e l'avvio di un nuovo processo in questo modo (P_x sono i processori che costituiscono il sistema, p_x i processi attivi sui vari processori, A_x i processi di sistema che si occupano dell'allocazione e B_x i processi che si occupano di fornire le informazioni sul carico, uno per ciascun processore):

1) Un processo p_i su un processore P_j intende lanciare un nuovo processo; esegue allora una variante della fork (fork_remote), che attiva il meccanismo di allocazione.
2) Il processo di servizio A_j su P_j riceve la richiesta di esecuzione della fork, e invia richieste di informazioni a tutti i processi B_k (su ciascun P_k);
3) Ciascun processo B_k risponde inviando i dati presi da /proc/loadavg (o /proc/uptime, a scelta);
4) A_j determina quale processore P_k deve eseguire il processo, ed invia i dati necessari (attraverso una funzione migration).

Data una funzione fork_remote() che copia i dati necessari per attivare il processo in remoto in un'area di memoria condivisa leggibile da A, una funzione migration() che esegue la migrazione (ricevendo come parametri la posizione dei dati nella memoria condivisa e l'indirizzo del processore target) una funzione decision() che calcoli quale sia il processore piu' adatto ad eseguire il nuovo processo date le informazioni trasmesse da ciascun B_k:

1. Sviluppare il codice necessario per la gestione della memoria condivisa (comprensivo di gestione della mutua esclusione, se necessaria).
2. Sviluppare il codice relativo alla comunicazione fra i processi di servizio A_k e B_k.
3. Sviluppare il codice relativo alla raccolta delle informazioni da parte di B_k.

Discutere inoltre i seguenti problemi:

1. Quali obiettivi si dovrebbe porre la funzione decision() nella determinazione del processo piu' adatto? Quali politiche dovrebbe adottare per raggiungere questi obiettivi?
2. Cosa succederebbe se due processi p_i e p_j decidessero allo stesso tempo di lanciare ciascuno un processo figlio? Se questo creasse problemi di carico, quali sarebbero le possibili soluzioni?

You have to develop, within a distributed system, the process allocator subsystem that receives requests for new processes and executes them on the available processors. Let P_x be the processors in the system, p_x the user processes, A_x the allocator system processes, and B_x the system processes that serve information requests from remote A_x processes, consider the following operation:

1. A process p_i on processor P_j decides to launch a new child process; it executes a fork_remote function, which activates the allocation mechanism;
2. Process A_j on P_j, detects the request for a new process, and sends enquiries to the B_k process on each P_k in the system to compute the overall system load information;
3. Each B_k reads load information from /proc/loadavg (or /proc/uptime, choose one), and sends the data back to A_j;
4. A_j chooses the most suitable processor P_k, and allocate the new process there.

Given the following functions:

- fork_remote(), saves all the data needed to create the new process in a shared memory area readable by A_j;
- migration(), receives a target processor address and the address of the required data on the shared memory, and executes the allocation and data transfer;
- decision(), receives the information from each B_k (as an array of processor addresses and load data items), and returns the address of the selected processor.

Develop the following:

1. The code required to manage the shared memory (including, if required, the management of mutual exclusion);
2. The code required for the communication between A_k and B_k;
3. The code required for B_k to collect the information from /proc/loadavg or /proc/uptime.

Discuss the following issues:

1. What goals should the decision() function have? What policies should it adopt to reach the goals?
2. What happens if two processes p_i and p_j, on different processors, try to fork a new child at the same time? If there is a load imbalance as a result, what can be done to solve the problem?

Nome	Cognome

Nome	Cognome

Nome	Cognome