

Sistemi Operativi

5 dicembre 2006

Compito B

Si risponda ai seguenti quesiti, giustificando le risposte.

- (a) Si descriva il meccanismo attraverso cui i programmi richiamano i servizi del sistema operativo. Si faccia qualche esempio.
(b) Qual è la differenza tra un interrupt e una trap?

Risposta:

- (a) I programmi richiamano i servizi del sistema operativo per mezzo delle chiamate di sistema (system call): sono solitamente disponibili come speciali istruzioni assembler o come delle funzioni nei linguaggi che supportano direttamente la programmazione di sistema (ad esempio, il C). Esistono vari tipi di chiamate di sistema relative al controllo di processi, alla gestione dei file e dei dispositivi, all'ottenimento di informazioni di sistema, alle comunicazioni. L'invocazione di una chiamata di sistema serve ad ottenere un servizio dal sistema operativo; ciò viene fatto passando dal cosiddetto *user mode* al *kernel mode* per mezzo dell'istruzione speciale TRAP. Infatti, il codice relativo ai servizi del sistema operativo è eseguibile soltanto in kernel mode per ragioni di sicurezza. Una volta terminato il compito relativo alla particolare chiamata di sistema invocata, il controllo ritorna al processo chiamante passando dal kernel mode allo user mode.
(b) Un interrupt è un segnale inviato dal controller di un dispositivo alla CPU per segnalare un evento (ad esempio un errore od il completamento di un'operazione), mentre una trap è un'interruzione software generata da un processo per mezzo di un'apposita istruzione.
- Si dia la definizione di processo e si descriva come i processi sono rappresentati nei sistemi operativi.

Risposta: Un processo è un programma in esecuzione nel sistema; quindi non consiste del solo codice, ma anche di tutte le risorse correlate: il program counter (PC), i registri, lo stack, lo stato di esecuzione, lo spazio di indirizzamento, le variabili globali, i file aperti, i timer in scadenza, i segnali e le routine di gestione dei segnali, le informazioni di accounting ecc. Nei sistemi operativi si tiene traccia dei processi in esecuzione in un'apposita struttura dati del kernel, ovvero, la tabella dei processi in cui ogni entry è costituita da un PCB (Process Control Block) in cui sono memorizzate tutte le informazioni sul processo relativo.

- Si consideri un sistema con scheduling della CPU a priorità con tre code, A, B, C, di priorità crescente, con prelazione tra code. La coda A è FCFS; le code B e C sono round robin con quanto di 15 e 10 msec, rispettivamente. Se un processo nella coda B o C consuma il suo quanto di tempo, viene spostato in fondo alla coda A o B rispettivamente.

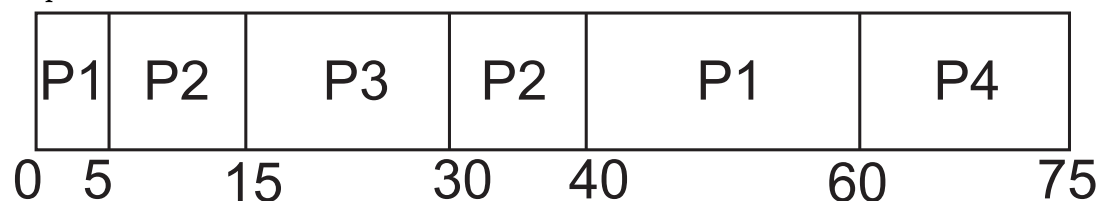
Nelle code A, B, C entrano i seguenti processi:

	coda	arrivo	burst
P_1	A	0	25ms
P_2	C	5	20ms
P_3	B	15	15ms
P_4	A	20	15ms

Si determini:

- il diagramma di GANTT relativo all'esecuzione dei quattro processi;
- il tempo di attesa medio;
- il tempo di reazione medio.

Risposta:



(a)

Sistemi Operativi

5 dicembre 2006

Compito B

(b) tempo di attesa medio = $\frac{(40-5)+(30-15)+0+(60-20)}{4} = \frac{35+15+40}{4} = 22,5 \text{ ms}$;

(c) tempo di reazione medio = $\frac{0+0+0+(60-20)}{4} = \frac{40}{4} = 10 \text{ ms}$.

4. Si consideri la seguente situazione, dove P_0, P_1, P_2 sono tre processi in esecuzione, C è la matrice delle risorse correntemente allocate, Max è la matrice del numero massimo di risorse assegnabili ad ogni processo e A è il vettore delle risorse disponibili:

	<u>C</u>				<u>Max</u>		
	<u>A</u>	<u>B</u>	<u>C</u>		<u>A</u>	<u>B</u>	<u>C</u>
P_0	0	3	1		0	5	1
P_1	2	2	2		2	4	3
P_2	1	3	0		2	4	1

<u>Available (A)</u>		
<u>A</u>	<u>B</u>	<u>C</u>
1	2	x

- (a) Calcolare la matrice R delle richieste.
 (b) Determinare il minimo valore di x tale che il sistema si trovi in uno stato sicuro.

Risposta:

- (a) Matrice delle richieste $R = Max - C$:

	<u>R</u>		
	<u>A</u>	<u>B</u>	<u>C</u>
	0	2	0
	0	2	1
	1	1	1

- (b) Il minimo valore di x tale da rendere lo stato sicuro è 0. Infatti $R_0 \leq A = (1, 2, 0)$ e quindi si può eseguire P_0 fino alla sua terminazione. A questo punto A diventa $A + C_0 = (1, 2, 0) + (0, 3, 1) = (1, 5, 1)$ e quindi $R_1 \leq A$. Dopo la terminazione di P_1 , A diventa $A + C_1 = (1, 5, 1) + (2, 2, 2) = (3, 7, 3)$. Quindi è possibile mandare in esecuzione anche P_2 (dato che $R_2 \leq A$) ed ottenere il valore finale di A , ovvero, $A + C_2 = (3, 7, 3) + (1, 3, 0) = (4, 10, 3)$. Ne segue che la sequenza sicura è $\langle P_0, P_1, P_2 \rangle$.
5. Si illustri la differenza fra segmentazione e paginazione. Dire quale dei due schemi di gestione della memoria soffre di frammentazione interna e quale di frammentazione esterna.

Risposta: Sia la paginazione che la segmentazione sono delle tecniche di gestione non contigua della memoria. La paginazione prevede una suddivisione della memoria fisica in frame di dimensione uguale (una potenza di 2, fra 512 e 8192 byte). Anche lo spazio di indirizzamento logico viene organizzato in pagine della stessa dimensione dei frame. Il sistema operativo tiene traccia dei frame liberi; infatti per eseguire un programma di n pagine, servono n frame liberi in cui caricare il programma stesso. Per ogni processo si imposta una page table per tradurre gli indirizzi logici nei corrispondenti indirizzi fisici.

La segmentazione invece è uno schema di gestione della memoria che supporta la visione utente, ovvero, considera un programma come una collezione di segmenti. Un segmento è un'unità logica di memoria; ad esempio: programma principale, procedure, funzioni, variabili locali, variabili globali, stack, tabella dei simboli, memoria condivisa possono essere considerati segmenti separati, di dimensioni differenti.

Per ogni processo la segment table consente di tener traccia dei vari segmenti allocati e di tradurre gli indirizzi logici della forma $\langle \text{segmento}, \text{offset} \rangle$ nei corrispondenti indirizzi fisici.

Un sistema di gestione della memoria con paginazione può soffrire del problema di frammentazione interna; infatti è difficile che le immagini dei processi in memoria abbiano dimensioni che siano multipli esatti della dimensione delle pagine. Quindi l'ultima pagina sarà soltanto parzialmente utilizzata.

Un sistema con segmentazione invece può soffrire del problema di frammentazione esterna; infatti i segmenti hanno dimensioni variabili e, caricando e rimuovendo processi dalla memoria, può capitare che si formino delle zone di memoria libera tra un segmento e l'altro così piccole da non consentire l'allocazione di un nuovo segmento.

Sistemi Operativi

5 dicembre 2006

Compito B

6. Si illustri la differenza fra le nozioni di servizio di rete e servizio distribuito.

Risposta: I servizi di rete offrono ai processi le funzionalità necessarie per stabilire e gestire le comunicazioni tra i nodi di un sistema distribuito (es.: l'interfaccia fornita dalle socket). In sostanza gli utenti devono essere consapevoli della struttura del sistema e devono indirizzare esplicitamente le singole macchine. I servizi distribuiti invece sono modelli comuni (paradigmi di comunicazione) trasparenti che offrono ai processi una visione uniforme, unitaria del sistema distribuito stesso (es: file system remoto). I servizi distribuiti vanno quindi a formare il cosiddetto *middleware*, ovvero, uno strato software fra il sistema operativo e le applicazioni che uniforma la visione dell'intero sistema.

7. Si consideri un disco gestito con politica SSTF. Inizialmente la testina è posizionata sul cilindro 20 e lo spostamento ad una traccia adiacente richiede 2 ms. Al driver di tale disco arrivano richieste per i cilindri 80, 30, 31, 70, 33, rispettivamente agli istanti 0 ms, 30ms, 40 ms, 50 ms, 70 ms. Si trascuri il tempo di latenza. In quale ordine vengono servite le richieste?

Risposta: Le richieste vengono servite nell'ordine 30, 31, 33, 70, 80 come illustrato dal seguente diagramma:

