

ASR2-Système : les processus

Semestre 2, année 2009-2010

Département d'informatique
IUT Bordeaux 1

Avril 2010

◀ ▶ ⏪ ⏩ 🔍

Définitions

Dans un système multi-tâches
un **processus** est une entité abstraite qui représente un **programme en cours d'exécution**

◀ ▶ ⏪ ⏩ 🔍

Définitions (suite)

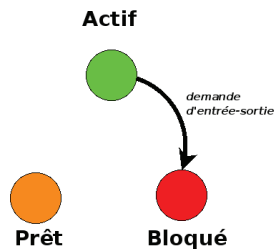
Définitions

- **États d'un processus** : actif, bloqué, prêt
- **contexte** d'un processus : valeur du **compteur ordinal**, **registre d'état**, **pointeur de pile** etc.
- **Process Control Block** : structure de données qui décrit un contexte
- la **table des processus** contient les PCB des processus présents.

◀ ▶ ⏪ ⏩ 🔍

Changements d'états

- Le processus actif se bloque quand il demande une opération d'entrée sortie.



◀ ▶ ⏪ ⏩ 🔍

Les processus

- 1 Concepts de base et définitions
 - Définitions
 - Définitions
 - Les états des processus
 - Changements d'état
 - Multitâche préemptif / coopératif
 - Définitions (suite)
- 2 Les interruptions
 - Exemple : chronomètre
 - Interruptions et multitâche
- 3 Politiques d'ordonnancement
 - Ordonnanceur
 - Critères d'évaluation
 - Tourniquet
 - Priorités
 - Files multiples

◀ ▶ ⏪ ⏩ 🔍

Un processus est défini par

- un **ensemble d'instructions** à exécuter (code du programme) ;
- un **espace mémoire** pour les données de travail (pile, tas) ;
- d'**autres ressources**, comme des descripteurs de fichiers ouverts, des ports réseau, etc.
- des **droits d'accès**

◀ ▶ ⏪ ⏩ 🔍

États des processus

Un processus peut être dans l'état

actif si il est effectivement en cours d'exécution

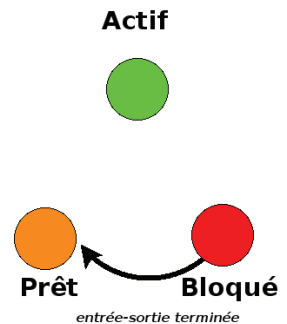
bloqué en attente d'un évènement (fin d'E/S, délai,...)

prêt si ni bloqué, ni actif.



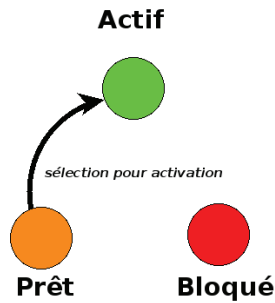
◀ ▶ ⏪ ⏩ 🔍

- un processus bloqué devient prêt quand l'opération d'E/S est terminée

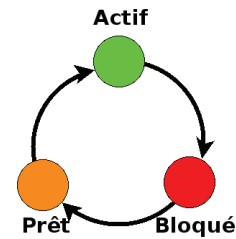


◀ ▶ ⏪ ⏩ 🔍

- un processus prêt peut être choisi pour devenir actif.
- il peut y avoir plusieurs processus actifs : le choix incombe à l'**ordonnanceur** (scheduler)
- diverses **politiques d'ordonnancement** sont envisageables



Transitions d'état

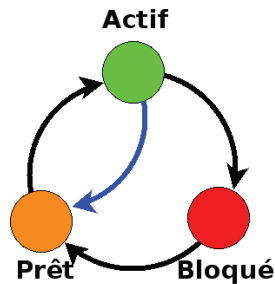


Multitâche préemptif / coopératif

Problème : cas d'un processus actif qui ne fait jamais d'E/S?

Solution : le **processus actif** peut également

- rendre volontairement la main (système **coopératif**)
- être interrompu au bout d'un **quantum de temps** (système **préemptif**)



Définitions (suite)

- une **commutation de contexte** se produit quand l'exécution d'un processus s'interrompt, ou reprend.
- **ordonnanceur** : composant du système, choisit un processus prêt pour l'activer.
- **quantum de temps** (dans système multitâche préemptif).
- **interruption** : signal matériel qui modifie la séquence normale d'exécution des instructions.

Les interruptions

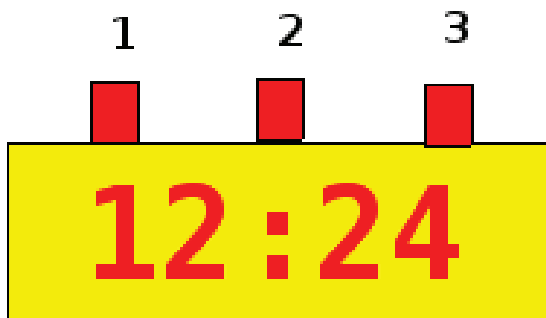
Une interruption

- est un signal qui modifie la séquence normale d'exécution des instructions par le processeur
- déclenche
 - la **sauvegarde de l'état** du programme (quelques registres)
 - l'exécution de la **routine de traitement** de l'interruption.

Système réactif

- Les interruptions sont à la base du fonctionnement des systèmes d'exploitation
- **système réactif** qui répond aux événements causés par son environnement (périphériques)

Exemple, chronomètre



Exemple, chronomètre

- **Programmation naïve** (sans interruption) :

```

secondes = 0
répéter indéfiniment :
    attendre 1 seconde
    secondes ++
    afficher secondes
fin - répéter
    
```

- suppose l'existence d'une fonction "attendre"
- Comment remettre le chronomètre à zéro ?

Exemple, chronomètre

Utilisation d'une interruption BTN1 :

```
quand BTN1 se produit , lancer Remise-à-zero  
  
secondes = 0  
  
répéter indéfiniment :  
  attendre 1 seconde  
  secondes ++  
  affichage = secondes  
fin-répéter  
  
routine Remise-à-zero :  
  centiemes = 0  
  secondes = 0  
  retour
```

Exemple de système réactif : chronomètre

Utilisation d'une horloge (centième de seconde)

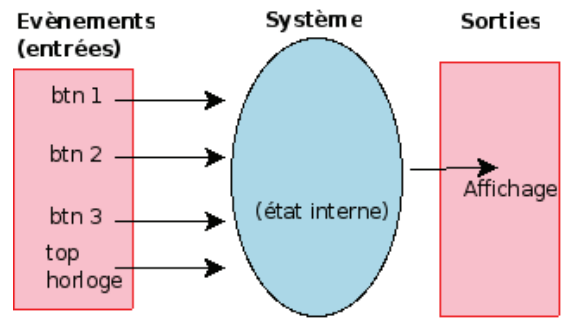
```
quand BTN1 se produit , lancer Remise-à-zero  
  
centiemes = 0  
secondes = 0  
  
répéter :  
  afficher (secondes)  
fin-répéter
```

Exemple, chronomètre

Utilisation d'une horloge (centième de seconde)

```
routine Remise-à-zero :  
  secondes = 0  
  
routine Top-Horloge :  
  centiemes ++  
  if (centiemes == 100) {  
    centiemes = 0  
    secondes ++  
  }  
  retour
```

chronomètre : un système réactif



Exemple, chronomètre (exercice)

Exercice :

- Ajouter un bouton STOP/RESTART
- ce bouton déclenche l'interruption BTN2
- conseil : utiliser une variable pour le "mode" (tourne, arrêté)

Dans un système multitâche

Dans un système multitâche, les interruptions sont causées par

- les **périphériques** (fin d'exécution de requête)
- des **signaux d'horloge**
- des **évènements extérieurs**
- dérivations en cas d'**erreur** (accès illégal à la mémoire, division par zéro ...)
- **interruptions logicielles** provoquées par instruction spéciale

Exemple : déroulement d'une interruption disque

- Déroulement
- 1 mettre le processus actif à l'état prêt
 - 2 déterminer la cause de l'interruption (p. ex : fin de lecture)
 - 3 trouver le processus demandeur (qui est bloqué)
 - 4 lui transférer les données reçues
 - 5 le remettre à l'état prêt
 - 6 activer un des processus prêts

Exemple : interruption horloge

Exemple : quantum de temps épuisé (ordonnancement avec réquisition, preemptive scheduling)

- Déroulement
- 1 remettre le processus actif à l'état prêt
 - 2 activer un des processus prêts

L'ordonnanceur

Définition

- partie du noyau d'un système d'exploitation multitâches
- consulte la liste des processus prêts, en choisit un pour l'activer
- applique une **politique d'ordonnement**

◀ ▶ ⏪ ⏩ ⏴ ⏵ ⏶ ⏷ ⏸ ⏹ ⏺ ⏻ ⏼ ⏽ ⏾ ⏿ 🔍 ↻

Choix de politique

Mais...

- ces **objectifs** sont **contradictoires**
- le comportement des processus ne peut pas être prévu

Il n'y a **pas de politique optimale**.

On emploie donc des **heuristiques**

◀ ▶ ⏪ ⏩ ⏴ ⏵ ⏶ ⏷ ⏸ ⏹ ⏺ ⏻ ⏼ ⏽ ⏾ ⏿ 🔍 ↻

Ordonnement avec priorités

On affecte une priorité (numérique) à chaque processus.

Déroulement

- choix du processus le plus prioritaire
- dans la même classe de priorité : tourniquet

Propriétés

- risque de **coalition**

Pour remédier aux problème d'équité, on peut faire varier les priorités (par exemple, baisse en fin de quantum).

◀ ▶ ⏪ ⏩ ⏴ ⏵ ⏶ ⏷ ⏸ ⏹ ⏺ ⏻ ⏼ ⏽ ⏾ ⏿ 🔍 ↻

Quelques critères

Comment reconnaître une **bonne politique** d'ordonnement ?

- équité
- efficacité
- minimiser le temps de réponse
- minimiser le temps d'exécution
- maximiser le rendement

Mais ...

◀ ▶ ⏪ ⏩ ⏴ ⏵ ⏶ ⏷ ⏸ ⏹ ⏺ ⏻ ⏼ ⏽ ⏾ ⏿ 🔍 ↻

Tourniquet

Ordonnement circulaire

synonymes : tourniquet, round-robin, FIFO, ...

Principe

- liste circulaire des processus prêts
- à la fin de son quantum de temps, un processus actif est placé en fin de liste

Propriétés

- **équité** garantie //
- choix du quantum : compromis entre *overhead* et // temps de réponse

◀ ▶ ⏪ ⏩ ⏴ ⏵ ⏶ ⏷ ⏸ ⏹ ⏺ ⏻ ⏼ ⏽ ⏾ ⏿ 🔍 ↻

Files multiples

File multiples

On définit des **classes** de processus

- à chaque classe correspond une liste (circulaire) de processus
- chaque classe est sélectionnée régulièrement

Propriétés

- respect des priorités
- évite les coalitions

◀ ▶ ⏪ ⏩ ⏴ ⏵ ⏶ ⏷ ⏸ ⏹ ⏺ ⏻ ⏼ ⏽ ⏾ ⏿ 🔍 ↻