

Université d'Oran Es-Sénia  
Faculté des Sciences  
Département d'Informatique

**CONCOURS D'ENTREE EN POST GRADUATION  
SYSTEMES INFORMATIQUES REPARTIS**

**MATIERE : SYSTEMES & RESEAUX**

- 
- **DUREE : 01H30 (Une Heure Trente Minutes)**
  - **DOCUMENTS NON AUTORISES**
- 

**IMPORTANT :**

*Une bonne lecture et compréhension d'un exercice ou d'un problème sont deux éléments clés pour le résoudre. Prenez donc le temps de lire attentivement l'ensemble des énoncés avant de commencer à répondre. La clarté, la précision et la concision des réponses, ainsi que leur présentation matérielle, seront des éléments importants d'appréciation et de notation.*

**Exercice : (6 points)**

- 1- Qu'est ce qu'un système multiprogrammé ? Un système de traitement par lots ? Un système en temps partagé ?
- 2- Citer les outils de synchronisation de haut niveau utilisés pour synchroniser les processus dans un système multi-tâches?
- 3- Généralement, les processus dans un système multi-tâches coopèrent pour réaliser une tâche particulière. Citer chacun des modèles de coopération entre processus.
- 4- Quel est l'objectif principal d'un système d'exploitation distribué ?

**Problème : (14 points)**

Dans ce problème, on souhaiterait définir un service d'accès à des données dupliquées afin d'assurer la tolérance aux fautes. Les données sont dupliquées en  $N$  exemplaires sur  $N$  serveurs différents. Un client peut faire deux opérations sur ces données :

- Des lectures qui seront notées par **Lire(x,v)**, où  $x$  représente une donnée et  $v$  la valeur lue à partir de cette donnée.
- Des écritures qui seront notées par **Ecrire(x,v)**, où  $x$  représente une donnée et  $v$  la valeur à écrire dans cette donnée.

Tout client, qui voudrait utiliser ce serveur, dispose d'une primitive d'accès multiple distant, notée **Exec(Opération, liste(S), x, v, liste(res))**, dont le fonctionnement est défini comme suit :

- Le paramètre **Opération** indique le type d'opération à réaliser : **Lecture** ou **Ecriture**.
- Le paramètre **liste(S)** contient la liste des serveurs sur lesquels le client veut réaliser la lecture ou l'écriture ; cette liste peut comprendre un ou plusieurs serveurs.
- Le paramètre  $x$  désigne la donnée à lire ou à écrire.
- Le paramètre  $v$  définit la valeur écrite dans le cas d'une opération d'écriture, et il prendra la valeur **nil** dans le cas d'une lecture.
- Le paramètre **liste(res)** est une liste des résultats qui a la même taille que **liste(S)**.

Un résultat peut avoir les valeurs suivantes :

- Pour une lecture : la valeur lue.
- Pour une écriture : la valeur écrite.
- Dans le cas où un serveur est défaillant (n'est pas disponible) on aura comme résultat la valeur **erreur** ; on considérera qu'un serveur est défaillant s'il ne répond pas au bout d'un temps défini, mesuré par un délai de garde (time out).

### Questions :

1. On souhaiterait que le service décrit ci-dessus puisse préserver l'invariant suivant, qui est supposé vrai à l'initialisation : « **Tous les serveurs en état de marche (c'est-à-dire non défaillants) contiennent des versions identiques des données** » ; ceci veut dire que le résultat d'une opération de lecture est le même quel que soit le serveur sur lequel a été lue la donnée. Expliquer en détail, comment se déroulent une opération de lecture et une opération d'écriture pour préserver cet invariant. Quelles propriétés doit avoir la primitive **Exec**, si on veut que plusieurs clients puissent exécuter des opérations concurrentes sur les données ?
2. On voudrait maintenant modifier le protocole de lecture et d'écriture comme suit : « **Une lecture n'est possible que si  $R$  serveurs (au moins) sont disponibles ; une écriture n'est possible que si  $W$  serveurs (au moins) sont disponibles** ». Chaque serveur maintient en outre un numéro de version pour chaque donnée et ce numéro est incrémenté de 1 à chaque écriture. Partant des hypothèses de ce nouveau protocole, répondez aux points suivants :
  - a. Lors d'une lecture, les valeurs lues sur différents serveurs peuvent-elles avoir des numéros de versions différents ? Justifier votre réponse à l'aide d'un exemple commenté.
  - b. L'invariant de la question 1 est modifié comme suit : « **Une opération de lecture délivre toujours une valeur à jour de la dernière opération d'écriture** ». Montrer que la condition  $R+W > N$  garantit que cet invariant est bien vérifié.
  - c. Quelles sont les modifications à apporter au protocole de la question 1 ?
  - d. Quel est d'après vous l'intérêt de ce nouveau protocole ?