Projet Systèmes et réseaux - Licence informatique S6

Université de Bretagne Occidentale, Département Informatique

Date limite de remise du projet : Vendredi 27 mars 2009

Travail à faire en binôme

Documents à rendre :

- un rapport (imprimé) décrivant votre travail
- les codes sources (imprimés)
- une version électronique des codes sources à adresser à `rodin@univ-brest.fr`

On souhaite réaliser un système Lecteurs/Ecrivains avec des processus lecteurs, des processus écrivains et trois serveurs (un pour les sémaphores (S1), un pour le nombre de lecteurs (S2) et un pour le livre (S3)) pouvant s'exécuter sur des machines différentes.
Les lecteurs et les écrivains connaissent (à partir de la ligne de commande) les machines supportant les serveurs et les ports associés au service qu'elles rendent.
Vous pouvez faire évoluer les prototypes des fonctions proposées dans les squelettes de programmes ci-dessous.
Vous pouvez vous servir, comme point de départ, de l'exemple de client-serveur TCP fourni ici:
`http://www.lisyc.univ-brest.fr/pages_perso/rodin/FTP/Enseignements/L3/Systeme`

1 Point de vue des lecteurs

Les lecteurs ont un fonctionnement classique : ils demandent d'abord au serveur S2 le nombre de lecteurs actuels. Si celui-ci est nul, ils positionnent un sémaphore d'accès puis demandent alors le livre au serveur S3. Si celui-ci est positif, ils peuvent avoir accès directement au livre. Après réception du livre (transfert TCP), ils libèrent le livre s'ils sont le dernier lecteurs.
La mise en place de sémaphores protégeant le livre et le nombre de lecteurs est nécessaire. On a donc un squelette de programme de la forme suivante :

```
Initialisations
P_Distant_S1(Sem_Nb_Lecteurs)
nblect = Get_Nb_Lecteurs_S2()
si (nblect == 0) alors
       P_Distant_S1(Sem_Livre)
fsi
Add1_Nblecteurs_S2()
V_Distant_S1(Sem_Nb_Lecteurs)

Get_Livre_S3()

P_Distant_S1(Sem_Nb_Lecteurs)
Sub1_Nblecteurs_S2()
nblect = Get_Nb_Lecteurs_S2()
si (nblect == 0) alors
       V_Distant_S1(Sem_Livre)
fsi
V_Distant_S1(Sem_Nb_Lecteurs)
```

Les fonctions `P_Distant_S1, V_Distant_S1, Get_Nb_Lecteurs_S2,Add_1_Nblecteurs_S2, Sub1_Nblecteurs_S2, Get_Livre_S3` ouvrent, à chaque appel, une nouvelle connexion vers les serveurs correspondants, se déroulent, puis ferment la connexion (pas d'ouverture globale de connexion en début de programme).

2 Point de vue des écrivains

Les écrivains ajoutent du texte dans le livre. Le fonctionnement des écrivains est donc le suivant : rédaction d'un texte, demande d'accès au livre, écriture dans le livre, libération du livre pour un autre écrivain ou un/des lecteurs. On a donc un squelette de la forme suivante :

```
Initialisations
phrase = Leture_Phrase_A_Ajouter()
P_Distant_S1(Sem_Livre)
Add_Livre_S3(Phrase)
V_Distant_S1(Sem_Livre)
```

Remarque : les squelettes proposés peuvent entrainer la famine des écrivains. On ne cherchera pas à résoudre ce problème.

3 Point de vue du livre (serveur S3)

Le serveur S3 peut recevoir deux types de requètes, une demande de livre de la part d'un lecteur ou une demande d'ajout dans le livre de la part d'un écrivain.
Par construction des lecteurs et des écrivains, des lectures simultanées sont possibles, mais il est impossible qu'une demande d'écriture et qu'une demande de lecture puisse être concurrente.
Le serveur S3 a donc le fonctionnement suivant : initialisations, attente sur une socket d'écoute d'une connexion, traitement de la requète par un processus (pas un thread) de service, retour en attente.
Le processus de service entraîne soit l'envoi du fichier, soit l'ajout en fin de la phrase transmise.

4 Point de vue du compteur de lecteurs (serveur S2)

Le serveur S2 peut recevoir 3 types de requètes : valeur du compteur, incrément ou décrément de la valeur. Par construction des lecteurs, il n'est pas possible que deux requètes arrivent simultanément. On a donc un fonctionnement très simple : initialisation, attente sur une socket d'écoute, acceptation de la connexion et attente de la requète, traitement de la requète par le serveur, réponse à la requète, fermeture de la connexion et retour en attente.

5 Point de vue du serveur de sémaphores (serveur S1)

Le serveur S1 gère 2 sémaphores et est réceptif à deux requètes (P et V) en provenance des différents lecteurs et écrivains. Ces requètes peuvent se dérouler simultanément et seront gérées par des processus (pas des threads) indépendants. On utilisera deux sémaphores se trouvant dans un tableau de sémaphores géré via un IPC System V.
Le fonctionnement du serveur S1 est donc le suivant : initialisations, attente sur une socket d'écoute d'une connexion, traitement de la requète par un processus (pas un thread) de service, retour en attente.
Le processus de service doir donc lire la requête (quel sémaphore, quel ordre) et essayer de l'effectuer. Lorsqu'il y parvient, il répond alors au demandeur (et par conséquent le débloque dans le cas d'un P) et se termine. Signalons que quel que soit l'ordre (P ou V), le processus de service devra répondre au demandeur.