UNIVERSITÉ DE ROUEN	ANNÉE 2004-2005
FACULTÉ DES SCIENCES	SYSTÈME D'EXPLOITATION
LICENCE INFORMATIQUE 3ème ANNÉE	EXAMEN
12 JANVIER 2005	3H00

Documents manuscrits autorisés + polycopié de cours.

Exercice 1 On souhaite réaliser la synchronisation de deux processus au moyen de sémaphores, les deux processus correspondant à l'itération infinie du schéma suivant :

```
        Phase A                        Phase B
P1  <--------------> RV  <---------------------> RV2
P2  <--------------> RV                          RV2
```

L'execution des deux processus doit donc être telle que :

1. Le processus P_1 ne peut entrer dans sa phase B qu'après que le processus P_2 a, d'une part terminé sa phase A et, d'autre part pris connaissance que lui même a terminé sa phase A (RV)
2. Le processus P_2 ne reprend sa phase A que quand le processus P_1 a terminé sa phase B et va rentrer lui même en phase A (RV2)
3. Les durées des phases A des deux processus et de la phase B du processus P_1 sont variables.

1) Écrire l'algorithme des processus P_1 et P_2. On pourra utiliser les opération $P_n()$, $V_n()$, $Z()$ et $Initsem()$

2) Ce modèle est maintenant enrichi de la manière suivante. Nous disposons de n processus indépendants réalisant une phase A puis une phase B. Nous souhaitons que tous les processus se donnent rendez-vous à la fin de leur phase A puis à la fin de leur phase B dans les mêmes conditions que précédemment. Pour résoudre cette question vous aurez certainement besoin de distinguer l'un des processus. Donner l'algorithme d'un processus général, puis l'algorithme du processus particulier.

Exercice 2 On considère la commande correspondant au source suivant. Dans cette commande, les retours d'appels-systèmes ne sont pas vérifiées afin de faciliter la lecture du code :

```
#include<stdlib.h>
#include<unistd.h>
#include<fcntl.h>
main(){
int d1,d2;
unlink("toto");unlink("tutu");                          /*1 */
d1 = open("toto",O_RDWR|O_CREAT,0666);                  /*2 */
link("toto","tutu");unlink("toto");                     /*3 */
write(d1,"abcdefgh",6);                                 /*4 */
if (fork() ==0) {                                       /*5 */
  lseek(d1,0L,SEEK_SET);                                /*6 */
  write(d1,"ABC",3);                                    /*7 */
  d2 = open("toto",O_RDWR|O_CREAT|O_EXCL,0666);         /*8 */
  printf("fils : d2=%d\n",d2);                          /*9 */
  write(d2,"DEF",3);                                    /*10*/
  sleep(1); }                                           /*11*/
```

```
else {                                                  /*12*/
  write(d1,"123",3);                                    /*13*/
  d2 = open("toto",O_RDWR|O_CREAT|O_EXCL,0666);         /*14*/
  printf("pere : d2=%d\n",d2);                          /*15*/
  write(d2,"456",3);                                    /*16*/
  sleep(1); }                                           /*17*
}                                                       /*18*/
```

On supposera qu'aucune activité externe ne crée, ni ne modifie de fichier possédant l'un des liens `toto` ou `tutu` dans le répertoire ou l'on travaillera.

1) Quels sont les différents résultats qu'il est possible d'obtenir et quel contenu pour les fichiers de nom `toto` et `tutu` (on supposera qu'une ouverture réussie renvoie le plus petit descripteur libre) ?

2) Décrire les tables du système en matière d'entrées-sorties pendant l'execution du sleep des deux processus.

Exercice 3 On désire implanter un système Client/Serveur d'échange de fichiers entre processus. Le schéma général est le suivant :

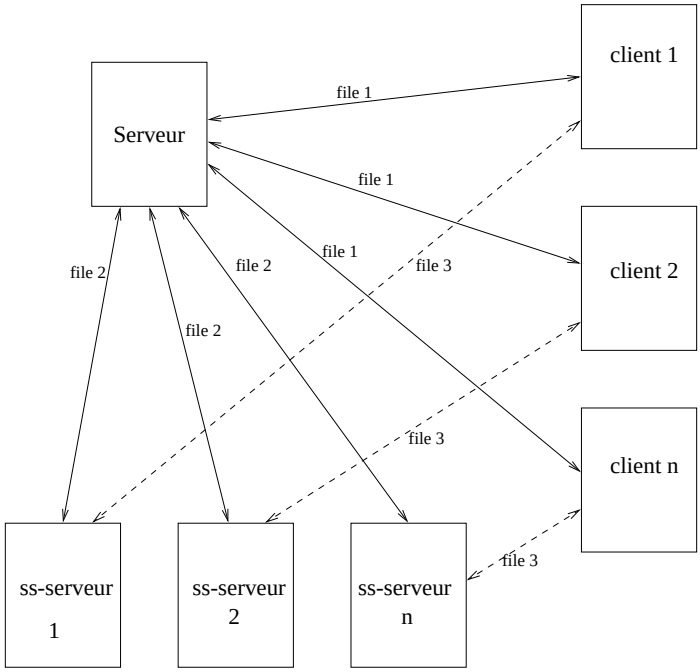

FIG. 1 – Echange de fichiers inter-processus via des files de messages

On utilisera uniquement trois files de messages. Une entre les clients et le serveur principal, une entre le serveur et les sous-serveurs et une dernière entre clients et sous-serveurs.

Le principe général est celui-ci :

- Un client envoie au serveur principal trois informations : son identifiant, s'il veut émettre ou recevoir, le fichier à émettre ou recevoir.
- Le serveur principal lit ces informations, se duplique (création d'un sous-serveur) et envoie les informations précédentes dans une file à destination du sous-serveur.

- Le sous-serveur gère la communication avec le client. Ils s'échangent des fichiers en lisant ceux-ci bloc par bloc. On supposera que la taille du bloc est prédéfinie.

1) La file 2 est-elle nécessaire ? Justifiez votre réponse.

2) Les n sous-serveurs et n clients vont utiliser la même file de message. Expliquez le mécanisme qui va permettre ce multiplexage. Comment peut-on savoir qu'un message est destiné au client k.

3) Le client ne connait pas à l'avance la taille du fichier qu'il va recevoir or il reçoit celui-ci bloc par bloc. Comment faire pour que le client (ou le sous-serveur) n'entre pas dans une boucle infinie sur la reception de paquets ? Proposez un modèle de données adapté.

4) On désire maintenant implanter un système de compte-rendu géré par le serveur principal. Celui-ci récupèrera des informations comme la taille du fichier par l'intermédiaire des sous-serveurs. Expliquez comment effectuer ces rapports en utilisant une des files de messages déjà existante. Quelle file doit-on utiliser ? Pourquoi ?

5) On souhaite maintenant changer de modèle et utiliser des tubes nommés. Ces tubes seront au nombre de deux, un pour la demande des clients vers le serveur, l'autre pour le traitement client/sous-serveur. Mettez en évidence le problème posé par un seul tube entre clients et sous-serveurs et une communication multiplexée.

6) Il faut à nouveau décrire une structure de données adaptée. La structure qui consiste a ne mettre que le bloc à envoyer n'est pas suffisante. En effet, il risque de vous manquer certains paquets. Décrire un protocole vous permettant de récupérer l'intégralité de votre fichier dans ce modèle avec uniquement deux tubes.

7) Pour simplifier le fonctionnement, on peut multiplier le nombre de tubes. Proposer une solution simple permettant un fonctionnement correct.

8) Comme à la question 3, le client (ou le sous-serveur) ne connait pas à l'avance la taille du fichier. Proposer une solution à ce problème (différente de la réponse à la question 3).