Réalisation d'un gestionnaire de tas réparti partagé

Nombre d'étudiants par projet : 2

Contact : Pierre Sens
Email : Pierre.Sens@lip6.fr

Sujet :

L'objectif est de concevoir un gestionnaire de tas réparti entre des processus situés sur des machines différentes. Le tas est stocké dans un segment mémoire dupliqué sur chaque processus. Chaque processus peut allouer une zone en lui associant un nom symbolique puis la lire et la modifier.

Les processus peuvent faire les opérations suivantes :

- Création d'une nouvelle donnée (t_malloc) : le processus alloue dans tous les tas une donnée à la même adresse et lui associe un nom.
- Demande un accès en lecture (t_access_read) : le processus demande l'accès en lecture à une donnée dont il connaît le nom. Si un accès en écriture est en cours sur un autre processus l'appel est bloqué.
- Demande un accès en écriture (t_access_write) : le processus demande l'accès en écriture à une donnée dont il connaît le nom. Si un accès est en cours sur un autre processus l'appel est bloqué.
- Manipulation de la donnée : une fois l'accès autorisé, la donnée peut être lue et modifiée directement en mémoire.
- Fin de manipulation de la donnée (t_release) ; le processus signale qu'il n'utilise plus la donnée. Toutes les modifications faites sur la donnée sont alors propagées à tous les processus.
- Destruction de la donnée (f_free) : le processus détruit la donnée dans tous les tas.

Le code suivant illustre un exemple d'utilisation sur un processus:

```
…
int*cpt ;
double *a;

init_data() ;  /* Initialisation du tas */

t_malloc(sizeof(int),"compteur"); /*Création de la donnée "compteur" */

cpt = t_access_write("compteur") ; /*Demander l'accès en écriture à la
donnée */
*cpt = 23*45;  /* Ecriture de la donnée  en local*/
t_release(cpt) ; /* Signaler la fin de la manipulation de la données : les
modifications sont propagées à tous les processus */

…
cpt = t_access_read(("compteur") ; /* Demander l'acces  en lecture */
printf(« %d \n », *cpt) ; /* Lecture de la donnée */
t_release(cpt) ; /* Signaler la fin de la manipulation de la données : les
t_free(cpt) ; /* Destruction */
```

Mise en oeuvre :

Un serveur de noms maintient la correspondance entre les adresses et le nom des données. Il maintient aussi pour chaque donnée sur les nombres d'accès en cours.

Le segment est découpé en pages, il est répliqué sur tous les processus client. Les clients doivent « mappé » le tas à la même adresse (utilisation de mmap).

Dans un deuxième, temps on s'intéressera à la fiabilisation du serveur de noms en créant des serveurs « mirroirs » maintenant une copie de la table de correspondance du serveur. En cas de défaillance, les processus client pourront basculer automatiquement sur un serveur disponible.

La mise en œuvre de ce projet reposera sur les sockets pour les communications entre clients et serveurs. Lors des expérimentations, il faudra simuler des fautes de serveurs.