DEA ISC

Construction des Applications Parallèles et Réparties

(mars 2000)

Les exercices portant sur les applications réparties et les applications parallèles sont indépendants. A titre indicatif le barème est le suivant : applications réparties (12 points) ; applications parallèles (8 points)

Des réponses précises et concises sont demandées

Applications Réparties

L'objet du problème est l'étude d'un *service de désignation* (appelé aussi *service de noms*) dont on rappelle que l'objectif est de mettre en œuvre la correspondance entre des noms symboliques manipulés par les programmes et des noms internes manipulés par le système. Les noms internes sont uniques et désignent sans ambiguïté une entité dans le système réparti. Par contre une entité donnée peut être désignée par plusieurs noms symboliques. Les noms permettent de désigner n'importe quelle entité manipulable par des programmes, par exemple des fichiers.

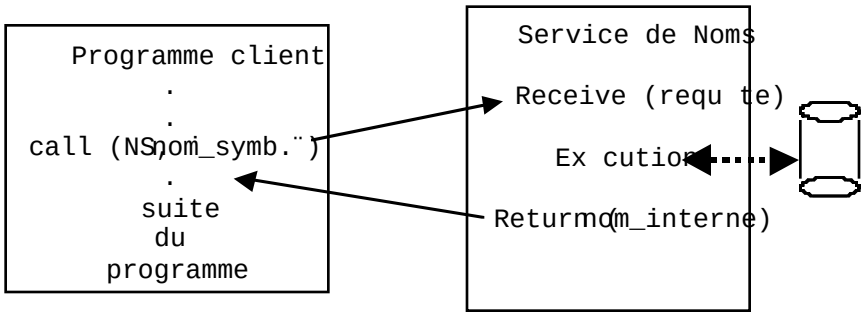
Question 1 – Structure des noms internes

Pour ce qui concerne la structure des noms internes (appelés aussi noms « système »), on distingue deux approches selon que le nom interne contient ou non l'identification du site de localisation de l'entité désignée.

Rappeler brièvement les avantages et les inconvénients de ces deux approches (15 lignes maximum).

Question 2 – Architecture d'un service de désignation

L'accès au service de désignation est réalisé par un modèle d'exécution client-serveur comme cela est illustré dans la figure ci-dessous.



une recherche dans un fichier qui contient la table de correspondance entre noms symboliques et noms internes. Le nom interne est renvoyé en résultat au programme client. Un code d'erreur est retourné si le nom symbolique n'est pas connu du service de désignation.

Sur un site, tout accès d'un programme client à une entité distribuée doit être précédé d'un appel au service de désignation pour connaître le nom interne de cette entité. Comment peut-on optimiser simplement le coût d'exécution des programmes clients en limitant les consultations du service de désignation (20 lignes maximum) ?

Question 3 – Service de désignation distribué

On considère maintenant que la table de correspondance n'est plus centralisée sur un seul serveur mais partitionnée entre plusieurs serveurs. Un nom symbolique donné est géré par un seul serveur de noms.

Question 3-a

En conservant le principe d'un modèle d'interaction de type client-serveur entre le programme client et le service de désignation, décrire un schéma d'exécution qui permette à un programme client de récupérer sur le « bon » serveur de désignation le nom interne associé à un nom symbolique donné. Dans un premier temps, on suppose qu'un site client ne connaît qu'un seul serveur de noms (appelé serveur de proximité).

Décrire le principe de l'algorithme exécuté par le programme client et le principe de l'algorithme exécuté par une instance du service de désignation sur un des serveurs (30 lignes maximum).

Question 3-b

On suppose maintenant qu'un site client connaît tout ou partie des serveurs de noms. Proposer, pour le site client, une architecture qui permette d'accélérer la traduction d'un nom symbolique. On décrira de façon plus détaillée l'algorithme du site client.

Question 4 – modèle d'exécution asynchrone

On souhaite maintenant comparer le schéma d'exécution précédent avec un schéma d'exécution asynchrone, fondé sur un bus à messages fiable (garantie de délivrance des messages) et un modèle de communication de type *Publish/Subscribe*.

Le bus à message est accessible à partir des primitives suivantes :

- *Subscribe (évt, fct)* : définit une association entre un nom d'événement (*évt*) et une action à exécuter (*fct*). Après exécution de la primitive *Subscribe*, l'occurrence de l'événement noté *évt* entraîne l'exécution de la fonction *fct*.
- *Publish (évt, one/all, params)* : un événement de nom *évt* avec les paramètres *params* est émis sur le bus. Si le paramètre *one* est positionné, une seule fonction associée à cet événement est exécutée (prise au hasard parmi les fonctions associées). Si le paramètre *all* est positionné, toutes les fonctions associées à cet événement sont exécutées.

Question 4.a

En utilisant ces primitives, donner une solution pour la configuration de la question 3 (service de désignation distribué).

Question 4.b

On s'intéresse maintenant à l'enregistrement d'un nom dans le service de désignation. Cette demande fait l'objet d'une requête spécifique depuis un programme client (le nom symbolique est passé en paramètre). Le service de noms vérifie que le nom symbolique n'est pas déjà enregistré. Il crée un nom interne et enregistre l'association < nom symbolique – nom interne > dans la table. Le résultat retourné au client est, soit le nom interne, soit un code d'erreur si le nom symbolique existait déjà.

A l'aide des primitives *Publish* et *Subscribe*, donner une solution de ce problème pour un service de noms distribués. On s'attachera à vérifier l'unicité d'un nom symbolique dans la table de correspondance.

Pour chacune des questions 4-a et 4-b , on précisera les types d'événements et les fonctions (réactions) qui leurs sont associées.