Examen de « Programmation Réseau »
Licence Informatique – Université Paris Diderot

Jean-Baptiste Yunès

17 Mai 2013

De nombreuses normes décrivent des mécanismes divers et variés de « calcul » à distance. Leur fonctionnement repose généralement sur trois entités : un annuaire, un serveur et un client.
L'annuaire recense l'ensemble des services de calcul qui ont bien voulu s'y déclarer.
Le client qui souhaite effectuer un appel à une fonction distante, interroge l'annuaire pour en obtenir les informations nécessaires à la réalisation de l'appel.
Le serveur qui désire offrir un service de calcul distant, se déclare auprès de l'annuaire en lui transmettant les informations utiles pour qu'un client puisse se connecter à lui, puis passe dans un mode passif dans lequel il attend des connexions entrantes afin de réaliser le service et d'en renvoyer la réponse. Un serveur peut attendre plusieurs connexions entrantes (une par fonction « exposée »).
Bien entendu pour garantir l'interopérabilité, il est nécessaire de fixer le format d'échange des données.
Pour répondre aux questions de l'examen vous devez choisir un langage parmi C ou Java.

Ordinateur, téléphones, etc sont interdits. La documentation papier est autorisée (sauf les feuilles du voisin).

1 Format d'échange de données

Une chaîne de caractère est codée sous la forme d'un triplet :
– le caractère `S`,
– longueur l (représentée sur un octet) dont la valeur est au plus 80
– et les l caractères de la chaîne (chacun 8 bits).
Le type correspondant s'appelera `netString`.

Un entier (toujours 32 bits) est représenté comme un couple :
– le caractère `I`,
– suivi par la représentation big-endian de sa valeur.
Le type correspondant s'appellera `netInt`.

Un message, c'est-à-dire une enveloppe d'objets de l'un des types précédents, transitant sur le réseau sera un couple :
– longueur l (représentée sur deux octets en big-endian),
– suite de l octets.
Le type correspondant s'appellera `netMsg`.

1. décrire les types dans le langage que vous avez choisi
2. écrire les fonctions `stringToNetString` et `netStringToString`.
3. écrire les fonctions `intToNetInt` et `netIntToInt`.
4. écrire les fonctions `addIntToNetMsg` et `addStringToNetMsg` permettant d'ajouter à un message existant et en fin de celui-ci un entier ou une chaîne de caractères passé(e) en paramètre. Attention le type

passé en paramètre est un type du langage, il devra donc devra être correctement transformé pour être ajouté.

À l'aide de ces fonctions, on va écrire des fonctions permettant de manipuler des messages.
D'abord des messages d'enregistrement de service. Ces messages sont constitués dans l'ordre, d'un entier de valeur 1 (pour représenté la volonté de s'enregistrer dans l'annuaire), d'une chaîne de caractère constituant le nom de la fonction distante qui pourra être appelée, d'un entier qui sera le nombre de paramètres que la fonction distante attend, d'un entier qui sera l'adresse de la machine sur laquelle le service de calcul est disponible, d'un

entier qui sera le numéro de port auquel il faudra s'adresser pour obtenir le calcul.

5. écrire une fonction `infoServiceToNetMsg(netMsg,name,params,address,port)` permet-

tant de fabriquer un message d'enregistrement et une fonction `netMsgToInfoService(netMsg,`

dantes. On précisera le type des arguments, etc

Ensuite des messages de consultation. Ces messages sont constitués dans l'ordre, d'un entier de valeur 2 (pour représenté la volonté de consulter l'annuaire), d'une chaîne de caractère constituant le nom de la fonction distante

3 Serveur

Un serveur qui désire offrir un service de calcul doit établir un service de connexion sur lequel les clients pourront se connecter afin de réaliser des appels distants, puis d'enregistrer ce service auprès de l'annuaire. La réponse à un appel est un message constitué d'un entier qui s'il vaut 0 signifie que l'appel a échoué et 1 si l'appel a réussi. Si l'appel réussi, un autre entier représentant la valeur de la fonction est intégré au message.

14. écrire une fonction `createService` permettant d'initier la connexion d'un service de calcul
15. écrire une fonction `registerService` permettant d'enregistrer le service correspondant auprès de l'annuaire
16. écrire une fonction `wait` permettant au serveur de se mettre en attente d'appels entrants.
17. que faudrait-il faire de particulier dans ces différentes fonctions si on voulait permettre à ce que sur un même numéro de port, plusieurs fonctions puissent être appelées ? (en considérant que les fonctions peuvent être longues à être exécutées).