Intergiciels – 2IR

1h45, documents autorisés

17 avril 2015

Les exercices sont indépendants.

1 Sockets (7 pt)

On souhaite réaliser un service d'alarme auquel un processus peut s'abonner pour être prévenu quand une certaine date ou un certain événement survient. Nous avons donc un serveur d'alarme et un ou plusieurs clients non nécessairement toujours présents. Le principe est le suivant :

— Le client établit une communication en mode connecté avec le serveur ;
— Il envoie au serveur les détails de son alarme (la date à laquelle il veut être prévenu par exemple) et les informations nécessaires pour que le serveur le recontacte plus tard ;
— Il attend la validation ou le refus du serveur. Celui-ci peut refuser la requête selon ses critères arbitraires (date dans le passé, trop grand nombre de requêtes, mauvaise humeur...) ;
— La connexion est terminée.

Plus tard, quand la date ou l'événement survient, le serveur envoie un datagramme au client pour l'avertir. Le serveur n'attend pas de réponse et ne se préoccupe pas de savoir si le message a effectivement été reçu.

Pour l'implantation du serveur, on supposera que sont fournies deux opérations `ajouter_alarme` et `attendre_alarme_suivante`. La première opération ajoute dans une liste d'attente une requête d'alarme à détecter, la deuxième *bloque* jusqu'à ce qu'une date ou événement survienne et renvoie alors la requête correspondante. Ces opérations sont purement locales et ne mettent en jeu aucune communication.

Questions :

1. Quelles sont les informations que doit connaître le client pour contacter le serveur ?
2. Quelles sont les informations que le client doit fournir au serveur dans sa requête pour que le serveur puisse lui envoyer l'alarme ?
3. Donner l'algorithme du client pour envoyer sa requête (séquence des appels systèmes tels que `connect`, `accept`, `read`..., en indiquant informellement les valeurs des paramètres essentiels ; on ne demande pas du code C).
4. Comment le client peut-il savoir que le serveur lui a envoyé un message d'alarme ?
5. Pour réaliser le serveur, combien d'activités concurrentes (de threads) faut-il au minimum ?
6. Donner l'algorithme de chacun de ces threads (toujours en indiquant les appels systèmes liés à la communication)

2 RMI (8 pt)

2.1 Questions de cours

On souhaite définir un objet accessible à distance avec une méthode ayant la signature suivante :

```
void foo(A a, B b);
```

1. Écrire l'interface `I` de l'objet accessible à distance, en supposant données les classes ou interfaces `A` et `B`.
2. On souhaite que le paramètre `a` soit transmis par copie. Quelle(s) condition(s) doit (doivent) être satisfaite(s) par `A` ?
3. On souhaite que le paramètre `b` soit transmis par référence, c'est-à-dire corresponde à un objet local au site de l'appelant. Quelle(s) condition(s) doit (doivent) être satisfaite(s) par `B` ?

2.2 Problème

On considère un hôtel qui offre un service de réservation accessible à distance. Le service `Hôtel` offre trois opérations :

— `réserver` : à partir d'un nom de client (chaîne), d'une date (chaîne) et d'un nombre de chambres (entier), cette opération renvoie un entier qui correspond au numéro de réservation si celle-ci peut être effectuée ou à -1 sinon

— `annuler` : à partir d'un numéro de réservation (entier), cette opération annule la réservation correspondante. Cette opération ne retourne rien. Si le numéro de réservation n'est pas valide, cette opération ne fait rien.

— `lister` : à partir d'un numéro de réservation (entier), cette opération retourne une chaîne de caractères qui fournit les caractéristiques de la réservation correspondante (nom du client, date, nombre de chambres). Si le numéro de réservation n'est pas valide, cette opération ne fait rien.

4. Donner l'interface RMI `Hôtel` avec les trois opérations précédentes.
5. Comment le client obtient-il un point d'accès au service `Hôtel` ?
6. Donner un code client qui effectue une réservation pour 3 chambres, puis une réservation pour 2 chambres, puis annule la première réservation.
7. Modifier la signature de `réserver` pour n'avoir qu'un seul paramètre `Réservation` regroupant le nom, la date et le nombre de chambres. Donner le code de la classe `Réservation`.
8. On ajoute une nouvelle fonctionnalité : l'hôtel peut annuler de lui-même une réservation mais doit en informer le client. Pour cela, mettre en place un mécanisme de callback : indiquer les modifications aux classes existantes et la ou les classes supplémentaires nécessaires, et indiquer quelle(s) méthode(s) de `Hôtel` il faut modifier ou ajouter.

3 Intergiciel à messages – JMS (5 pt)

On souhaite réaliser un service d'alarme auquel un processus peut s'abonner pour être prévenu quand une certaine date ou un certain événement survient. Nous avons donc un serveur d'alarme et un ou plusieurs clients.

Un client envoie au serveur une requête indiquant à quelle date ou à quel événement il s'intéresse. Quand cette date ou cet événement survient, le serveur envoie un message uniquement destiné au client intéressé.

1. Indiquer quelles sont les Destinations nécessaires, en précisant s'il s'agit de Topic ou de Queue.
2. Comment le client peut-il indiquer au serveur la destination qui le concerne ?

Pour assurer la résistance aux défaillances du service d'alarme, on installe plusieurs serveurs sur différentes machines. On souhaite que le client n'ait pas à savoir combien il y a de serveurs.

3. Qu'est-ce que cela change pour la (ou les) Destination(s) servant à communiquer avec le(s) serveur(s) ?
4. Le client ne souhaite être informé qu'une et une seule fois pour une requête d'alarme donnée. Comment réaliser cette propriété ?