Examen Conception d'Applications Réparties

Maîtrise Informatique – Maîtrise GMI

Université des Sciences et Technologies de Lille

2002-2003

3 heures – Tous documents autorisés

1. Espace de données partagées

Le but de cet exercice est de définir un serveur CORBA de données partagées (`ServDP`) dans lequel des clients viennent déposer et retirer des données. Un tel service peut servir par exemple dans un environnement distribué, à ce que des stations de travail déposent des travaux à effectuer, et qu'une ou plusieurs autres stations, en fonction de leurs disponibilités, récupèrent ces travaux et les exécutent éventuellement de façon concurrente. Ce service s'inspire du système Linda qui implémente un tel espace de données sous forme d'une mémoire partagée répartie. Il s'agit ici d'en réaliser une implantation sous forme d'objets CORBA.

Interface IDL

1 `ServDP` stocke des couples clé – valeur, où clé est une chaîne de caractères et valeur une séquence d'octets. Le stockage de chaque valeur est ainsi identifié par une clé unique. Son interface fournit les méthodes suivantes :
- `deposer` : prend 2 paramètres qui permettent au client de spécifier la clé et la valeur à stocker
- `retirer` : le 1er paramètre est la clé à retirer. La valeur en retour associée à la clé est fournie au client par un 2ème paramètre.
- `lire` : idem `retirer` sauf que le couple clé – valeur reste stocké.

Elle comprend de plus l'accès à un attribut de type entier contenant le nombre de couples clé – valeur stockés.
Définir l'interface CORBA IDL correspondant à cette spécification. Si l'hypothèse sur l'unicité des clés stockées dans l'espace est relâchée (i.e. on considère que plusieurs clés identiques peuvent être stockées simultanément), quelles en sont les conséquences sur l'interface IDL ?

2 En se replaçant dans l'hypothèse d'une clé unique, définir des exceptions pouvant survenir pour chacune des méthodes de l'interface. Ces exceptions remontent autant d'informations que possible, par exemple un code et un message.

3 Proposer une autre signature pour les méthodes `retirer` et `lire` qui, à l'aide de la valeur de retour, indique si la clé est présente ou non. Discuter les avantages et les inconvénients des deux approches. On suppose maintenant que certaines valeurs de clé peuvent être

rejetées car contenant des caractères non valides. Comment prendre en compte cette situation ?

4 En ne considérant pas les exceptions, quelles méthodes de l'interface définie en 1 pourraient être déclarées `oneway` ? Rappeler les caractéristiques des appels de méthodes `oneway`.

5 On considère maintenant que la valeur peut être soit une valeur nulle, soit une séquence d'octets. Définir le profil des méthodes prenant en compte ce changement.

6 Jusqu'à présent, les clients qui ont besoin de lire ou de retirer des données n'ont d'autres alternatives, lorsque la donnée est absente, que d'interroger périodiquement le serveur. Cette interrogation réalisée à distance, est potentiellement coûteuse. On souhaite mettre en place un mode plus interactif dans lequel, en cas d'absence de données, les clients sont prévenus dès que cette donnée devient disponible. Mettre en place les éléments IDL qui permettent de faire cela.

7 On se place maintenant dans le cas où une station maître dépose dans l'espace de données une tâche à effectuer, et qu'elle veut être renseignée sur l'état d'activité de la station esclave ayant pris en charge la tâche : essentiellement, on veut savoir si la station esclave est plantée ou si elle est toujours en marche. Un tel mécanisme s'apparente à de la détection de défaillances et permet dans les systèmes tolérant les fautes de pouvoir réagir (typiquement re-affectation de la tâche à une nouvelle station esclave) en cas de défaillance. Proposer deux mécanismes permettant de réaliser cette détection et définir les interfaces IDL les spécifiant.

2. Transfert de fichiers fiable

On souhaite mettre en place un logiciel de transfert de fichiers d'une machine vers une autre machine. Les fichiers à transférer ont une taille ne dépassant pas 5 Moctets. Les concepteurs décident, dans un premier temps, de réaliser le logiciel à l'aide de la technologie JavaRMI. La spécification du logiciel fait apparaître un ensemble de fonctionnalités décrites ci-dessous.

- Authentification de l'utilisateur par son nom et son mot de passe,
- Manipulation de fichiers

Après une authentification réussie, les utilisateurs ont accès aux fonctionnalités suivantes de manipulation de fichiers :
- Envoi d'un fichier sur la machine distante
- Lister le contenu d'un répertoire
- Changer de répertoire
- Quitter le système de transfert de fichiers

8 Donner les interfaces de `ManipulationDeFichiers` et d'`Authentification` des classes Java qui vont permettre la réalisation des fonctionnalités de manipulation de fichiers décrites ci-dessus.

9 Plusieurs cas d'erreur peuvent se produire lors de la manipulation de ces fichiers. Quelles sont les exceptions à gérer ?

Fiabilité du transfert de fichiers

10 Cette application de dépôt de fichiers est partagée par de nombreux utilisateurs. Le risque est que l'écriture d'un fichier écrase un ficher existant portant le même nom dans le répertoire où s'est positionné l'utilisateur. Dans ce cas, on souhaite que le fichier soit tout de même enregistré, sans perte de l'ancien. Proposez une solution permettant d'éviter l'écrasement du premier fichier.

11 Le réseau qui permet l'accès à cette application est peu fiable. Les coupures sont fréquentes. Aussi pour permettre le transfert de très gros fichiers, il est nécessaire de mettre en place un mécanisme de reprise efficace après une coupure réseau. L'envoi de fichier est donc découpé en blocs de données de 128 Koctets. Chaque fois qu'un bloc est envoyé, si il est correctement reçu et enregistré sur le disque, il est acquitté. Cet acquittement est noté côté serveur et envoyé côté client. Après une coupure réseau, il y a reconnexion et authentification à nouveau de l'utilisateur. L'envoi du fichier peut alors reprendre à partir du dernier acquittement noté côté serveur. Définissez une nouvelle interface permettant la gestion de l'envoi de fichier entre un client et un serveur suivant cette description.

12 Donnez le code Java côté client et côté serveur suivant l'interface définie en 11 et permettant l'envoi de fichier.

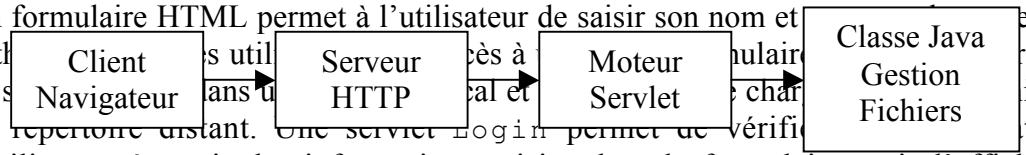
13 Suite aux questions 10, 11 et 12, donnez la nouvelle interface `ManipulationDeFichiers`.

Accès Internet

14 Les concepteurs de l'application décident de faciliter l'accès à cette application par un accès via un navigateur Web. Ils proposent de mettre en œuvre un environnement à base de servlets. Le serveur de fichiers est indépendant du serveur de Servlet. Il reste accessible via JavaRMI. Donner l'architecture logicielle à mettre en place.

15 Y-a-t-il besoin de gérer une session ? un cookie ? pourquoi ?

16 Un formulaire HTML permet à l'utilisateur de saisir son nom et [illegible]. Après auth[illegible]s util[illegible]ès à [illegible]mulair[illegible] permet de s[illegible]ans u[illegible]al et [illegible] char[illegible]ier dans un répertoire distant. Une servlet `Login` permet de vérifi[illegible]tion de l'utilisateur à partir des informations saisies dans le formulaire, puis l'affichage du formulaire suivant. On vous demande d'écrire la servlet suivante appelée servlet `ChargementFichier` qui permet, à partir du nom de répertoire distant, d'un nom de fichier et d'un objet de type `file`, de gérer l'échange nécessaire l'envoi du fichier de façon fiabilisée vers le serveur de fichiers.



17 Les concepteurs se posent maintenant la question du passage à un environnement Web Services. Que vont-ils devoir changer ?

Une question supplémentaire : avez-vous déjà utilisé une telle application ? si oui, dans quel contexte ?