ANDRE Julien

ROLLIN Antoine

04/05/2016

SI4

RMI-Spork

Applications Réparties – Mini-Projet

L'objectif de ce projet est de mettre en œuvre une application répartie représentant un portail générique de partage de données, à l'aide des différents outils Java vus en cours. Un client pourra alors utiliser ce portail pour déposer ou récupérer des données ou des services.

Pour ce faire nous aurons besoin de :

- Un serveur utilisant la technologie RMI et capable de stocker des données et des services
- Une application cliente qui permet d'interagir avec le serveur
- Un serveur de classes capable de mettre à disposition des fichiers java compilés

Table des matières

1. Serveur de collection universel

1.1. Description

Le serveur de collection est l'entité qui permet de collecter et distribuer les données et services. Elle utilise la technologie JNDI pour mettre des annuaires d'objet Java à disposition des clients. Le serveur représente le cœur de notre application.

1.2. Collection

La base de données est représentée par une HashMap pour pouvoir stocker des objets et les référencer par des clés uniques.

La collection enregistre également les actions client à l'aide de Queues représentant un historique. Pour des raisons de mémoire, ces unités de stockage disposent d'une certaine capacité. Si ces Queues atteignent la limite de stockage, alors l'élément le plus vieux laisse sa place au nouvel élément (principe de la FIFO).

Dans notre projet, nous utilisons un objet nommé Gateway pour interagir avec la base de données. A chaque émission ou réception d'objet, les statistiques sont mises à jour et sont accessibles par diverses méthodes de la classe.

1.1. Réception d'un objet

Lorsqu'un objet est réceptionné par le serveur de collection (méthode put), il est stocké dans la base de données et référencé par une clé (un objet ne peut être référencé par une clé qui existe déjà). Ensuite, l'historique des clés reçus est mis à jour.

1.2. Emission d'un objet

Lorsqu'un objet est demandé par un client (méthode get), le serveur de collection va vérifier l'existence de l'objet souhaité. S'il existe, il caste cet objet en DistantObject de façon à le rendre Serializable, et le retourne. Ensuite, l'historique des clés demandées est mis à jour.

1.3. Service d'information

Le serveur met à la disposition des clients un service d'information permettant d'obtenir des statistiques sur la collection. Ce service d'information utilise l'objet Gateway notamment pour :

- Récupérer la liste des objets récupérables
- Récupérer l'ensemble des N dernières clés enregistrées
- Récupérer l'ensemble des N dernières clés utilisées
- Récupérer l'ensemble des clés les plus utilisées

1.4. Gestion des abonnements ?

2. Client

2.1. Description

Le client est l'entité qui va envoyer ou recevoir des données ou services depuis le portail générique.

2.2. Connexion

La connexion d'un client est représentée par la récupération de l'objet distant CollectionServer à l'aide du JNDI. Ainsi, il peut effectuer toutes actions qu'autorise le serveur, à savoir l'envoi d'une donnée ou d'un service et l'utilisation du service d'information.

2.3. Déposer une Donnée ou un Service

Après s'être « connecté » au serveur, le client peut déposer (méthode put) une donnée ou un service. Les objets déposables sur le serveur doivent implémenter l'interface DataInterface pour les données ou l'interface ServiceInterface pour les services. Ces interfaces permettent aux clients qui récupéreront l'objet de pouvoir l'exploiter.

Pour déposer un objet, le client doit spécifier un clé (chaine de caractères) qui sera utilisé pour référencer l'objet dans la collection du serveur.

2.4. Récupérer une Donnée ou un Service

Après s'être connecté au serveur, le client peut également récupérer (méthode get) une donnée ou un service à l'aide de la clé qui le référence. L'objet distant étant récupéré, le client va d'abord tenter de le caster en service à l'aide de l'interface ServiceInterface. Si le cast échoue, il le caste alors en donnée à l'aide de l'interface DataInterface. L'objet est alors utilisable à travers les méthodes définies par les interfaces.

2.5. Récupération des informations sur la collection

Lorsque le client se connecte au serveur, il récupère la collection distante et donc le service d'information qu'elle embarque. Ce service est alors utilisable par le client et propose un panel de fonctions offrant la récupération d'informations concernant la collection (voir plus haut).

2.6. Abonnement ?

bla

3. Serveur de classes

Le problème avec l'architecture actuelle est que le client n'a pas accès aux interfaces qu'implémente le serveur de collection et son service d'information. Il ne peut donc les utiliser qu'en ayant à sa

disposition les fichiers .class nécessaires. C'est pourquoi un serveur de classes est utilisé pour mettre ces fichiers à la disposition du client.

4. Utilisation

bla