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 2](#_Toc450138079)

[1.1. Description 2](#_Toc450138080)

[1.2. Collection 2](#_Toc450138081)

[1.1. Réception d’un objet 2](#_Toc450138082)

[1.2. Emission d’un objet 2](#_Toc450138083)

[1.3. Service d’information 2](#_Toc450138084)

[1.4. Gestion des abonnements ? 2](#_Toc450138085)

[2. Client 2](#_Toc450138086)

[2.1. Description 2](#_Toc450138087)

[2.2. Connexion 3](#_Toc450138088)

[2.3. Déposer une Donnée ou un Service 3](#_Toc450138089)

[2.4. Récupérer une Donnée ou un Service 3](#_Toc450138090)

[2.5. Récupération des informations sur la collection 3](#_Toc450138091)

[2.6. Abonnement ? 3](#_Toc450138092)

[3. Serveur de classes 3](#_Toc450138093)

[4. Utilisation 3](#_Toc450138094)

# Serveur de collection universel

## 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.

## 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.

## 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.

## 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 cast cette objet en DistantObject de façon à le rendre Serializable, et le retourne. Ensuite, l’historique des clés demandées est mis à jour.

## Service d’information

Définir le service

Gateway

Lister les actions possibles

## Gestion des abonnements ?

# Client

## Description

Bla

## Connexion

bla

## Déposer une Donnée ou un Service

bla

## Récupérer une Donnée ou un Service

Bla

## Récupération des informations sur la collection

Bla

## Abonnement ?

bla

# Serveur de classes

bla

# Utilisation

bla