Hochschule RheinMain
DCSM - Informatik
Prof. Dr. Reinhold Kröger

Verteilte Systeme
WS 2015/16

Übungsblatt 5
Praktische Übungen

Ziel dieser Übung ist es, einen Java-basierten Gateway für das in Übungsblatt 3 entwickelte RPC-Protokoll umzusetzen. Die Funktionalität des Filesystems aus Übungsblatt 1 soll dazu über ein Java-Interface bereitgestellt werden. Das Gateway nimmt damit eine Client-Rolle gegenüber dem C-Server aus Aufgabe 3 ein.

Informationen zur Socket-Programmierung unter Java finden Sie unter den folgenden Referenzen:

- Oracle: "Lesson: All About Sockets (The Java™ Tutorials > Custom Networking)", https://docs.oracle.com/javase/tutorial/networking/sockets/
- Galileo Computing: "Galileo Computing :: Java ist auch eine Insel - 21.6 Mit dem Socket zum Server“, http://openbook.rheinwerk-verlag.de/javainsel9/javainsel_21_006.htm

Hinweis:
Sie finden in Ihrem SVN-Repository im Ordner 5 vier Eclipse-Projekte, die Sie für die Umsetzung verwenden sollen, sowie einen fertigen Server für x86 und AMD64 Linux-Platformen. Das zur Verfügung gestellte Skelett besteht aus folgenden Projekten:

1. „VFileSystemServerAPI“ enthält das Java-Interface für das VSFilesystem. Java-Anwendungen sollen darüber die Filesystem-Funktionalität verwenden können. **Hier sind keine Änderungen vorzunehmen!** (d.h. ändern verboten)
2. „VFileSystemServerCLI“ enthält ein fertigen Client der auf dem Java-Interface aufbaut. Dieser dient dazu Ihr Gateway zu testen. Hier **müssen keine** Änderungen vorgenommen werden (d.h. Sie können etwas änderen, es sollte aber nicht notwendig sein).
3. „VFileSystemServerGateway“ enthält den Gateway. Dieser implementiert das Java-Interface und kommuniziert mit dem C-Server. **Die drei enthaltenen Klassen sind mit Ihrer Lösung zu erweitern.** Die FilesystemImpl-Klasse greift auf die Klassen des VFileSystemServerMessages-Projekts zu.
4. „VFileSystemServerMessages“ enthält die Umsetzung des Fileserver-RPC-Protokolls. Hierzu gibt es für jeden Payload-Typen des Protokolls eine Klasse, die die enthaltenden Informationen vorhält. Diese Klassen implementieren ein Marshaller-Interface und kümmern sich damit selber um ihre Serialisierung, bzw. Deserialisierung. **Diese Funkionalität sollen Sie in den gegebenen Klassen umsetzen.**

Sie müssen für die Umsetzung keine weiteren Dateien/Klassen anlegen. Sollten Sie dennoch die Notwendigkeit sehen, neue Dateien/Klassen anzulegen, dann denken Sie daran diese mit `svn add <Dateiname>` dem SVN hinzuzufügen.

Aufgabe 1 (Projekt „VFileSystemServerGateway "):

In dem Projekt „VFileSystemServerGateway finden Sie drei Klassen, die Sie vervollständigen müssen.

- Die Klasse „Main“ ist ausführbar und muss um den Code zum Verbindungsaufbau zum Server erweitert werden. Anschließend erzeugt sie mit dem Socket eine neue Instanz der Klasse „Connection“, unter deren Nutzung wiederum eine Instanz von "FileSystemImpl" erzeugt und dem Kommandozeilen-Interface übergeben wird. Achten Sie darauf, dass die notwendigen Informationen über den Hostname und Port des Servers über die Programmargumente übergeben werden.
- Die Klasse "Connection" verwaltet die Socket-Verbindung. Darin muss die Methode „remoteOperation“ implementiert werden, die eine "FileServerMessage“ entgegen nimmt, diese mit deren marshall-Methode serialisiert, über den Socket an den Server schickt, die Antwort empfängt, diese nun mit der unmarshall-Methode deserialisiert und das Ergebnis an den Aufrufenden zurück gibt. Denken Sie hier auch wieder daran, dass der Server bei der Antwort zuerst 4 Byte mit der Längeninformation verschickt.
- Die Klasse „FilesystemImpl“ implementiert das Java-Interface „Filesystem“ und ist an die Schnittstelle des C-Filesystems aus Aufgabe 1 angelehnt. In den jeweiligen Methoden müssen Sie eine „FileServerMessage“ mit der jeweiligen Payload-Klasse erzeugen und richtig befüllen. Diese übergeben Sie der remoteOperation-Methode der Connection-Klasse. Das Ergebnis packen Sie aus und übergeben es als jeweiligen Rückgabe, bzw. Out-Parameter. Achten Sie darauf, dass auch Fehler-Messages vom C-Server zurückkommen können. In diesem Fall werfen Sie IOExceptions mit der jeweils mitgegebenen Fehlermeldung.

In dem Projekt „VFileSystemServerMessages“ müssen Sie die marshall, bzw. unmarshall-Methoden des Marshall-Interfaces implementieren. Da der Gateway gegenüber dem C-Server die Client-Rolle übernimmt, müssen Sie für die Request-Typen nur die marshall-Methode implementieren und für die Response-Typen entsprechend die unmarshall-Methoden. Ein Sonderfall ist die Klasse „FileServerMessage“, die die Referenz auf die jeweiligen Payload-Klassen enthält. Diese muss das vollständige Marshall-Interface umsetzen, soll aber dort nur den Header bauen und die Serialisierung des Payloads an die jeweilig referenzierte Payload-Klasse deligieren.