JUST DO IT
HÖHERE ABTEILUNG FÜR
INFORMATIONSTECHNOLOGIE

tgm
Die Schule der Technik

Laborprotokoll

RMI

Systemtechnik Labor
4BHIT 2015/16, GruppeX

Thomas Fellner

Version 0.1
Begonnen am 22. April 2016
Beendet am 25. April 2016

Note:
Betreuer: M. Borko

Inhaltsverzeichnis

1 Einführung

Verteilte Objekte haben bestimmte Grunderfordernisse, die mittels implementierten Middlewares leicht verwendet werden können. Das Verständnis hinter diesen Mechanismen ist aber notwendig, um funktionale Anforderungen entsprechend sicher und stabil implementieren zu können.

1.1 Ziele

Diese Übung gibt eine einfache Einführung in die Verwendung von verteilten Objekten mittels Java RMI. Es wird speziell Augenmerk auf die Referenzverwaltung sowie Serialisierung von Objekten gelegt. Es soll dabei eine einfache verteilte Applikation in Java implementiert werden.

1.2 Voraussetzungen

- Grundlagen Java und Software-Tests
- Grundlagen zu verteilten Systemen und Netzwerkverbindungen
- Grundlegendes Verständnis von nebenläufigen Prozessen

1.3 Aufgabenstellung

Folgen Sie dem offiziellen Java-RMI Tutorial [1], um eine einfache Implementierung des PI-Calculators zu realisieren. Beachten Sie dabei die notwendigen Schritte der Sicherheitseinstellungen (Security-Manager) sowie die Verwendung des RemoteInterfaces und der RemoteException.

Implementieren Sie ein Command-Pattern [2] mittels RMI [3] und übertragen Sie die Aufgaben/-Berechnungen an den Server. Sie können am Client entscheiden, welche Aufgaben der Server übernehmen soll. Die Erweiterung dieser Aufgabe wäre ein Callback-Interface auf der Client-Seite, die nach Beendigung der Aufgabe eine entsprechende Rückmeldung an den Client zurück senden soll. Somit hat der Client auch ein RemoteObject, welches aber nicht in der Registry eingetragen wird sondern beim Aufruf mittels Referenz an den Server übergeben wird.

2 Ergebnisse

2.1 Ant

2.2 Java Policy

RMI braucht bestimmte Genehmigungen. Um dem Programm diese zu geben, muss die Datein java.policy unter dem Verzeichnis der JDK `/usr/lib/jvm/{java-version}/jre/lib/security/java.policy` ins Homeverzeichnis (`/home/{name}/.java.policy`) kopiert werden, wobei bei diesem Block

```
1 grant codeBase "file:${{java.ext.dirs}}/*" {
        permission java.security.AllPermission;
3 };
```

Listing 1: java.policy

der Ausdruck `${{java.ext.dirs}}/*` dann noch zu `/home/name/-` geändert werden muss. Dies sagt aus, dass alle Datein (-) unter dem Homeverzeichnis diese permission haben. In disem Fall ist alles erlaubt, wegen des Ausdrucks `java.security.AllPermission`.

2.3 Callback

Client gibt skeleton zu Server. Muss nicht in die Registry gespeichert werden, da nur der Server antworten soll und nicht irgendein anderer Client

Literatur

[1] The java tutorials - trail rmi. `http://docs.oracle.com/javase/tutorial/rmi/`.

[2] Vince Huston. Command pattern. `http://vincehuston.org/dp/command.html`.

[3] Michael Borko. Beispiel konstruktor für command pattern mit java rmi. `https://github.com/mborko/code-examples/tree/master/java/rmiCommandPattern`.

Tabellenverzeichnis

Listings

Abbildungsverzeichnis