Zürcher Hochschule
für Angewandte Wissenschaften

School of
Engineering

Lernaufgabe

LE 2 – Implementierungskonzepte von Client/Server-Systemen und verteilte Objekte

Aufgabe 2: Verteilte Objekte

Lernziel

Sie sind in der Lage, die Programmierung von verteilten Objekten (mit Java RMI) aus Sicht eines Anwenders zu verstehen und anzuwenden.

Aufgabe

Probieren Sie die auf Moodle abgelegten Programme ChatClient und ChatServer (LE 2, Aufgabe 2) zuerst lokal aus (auf einem Notebook) und dann in einer Gruppe von 3-4 Personen. Beantworten Sie danach folgende Fragen:

- Wie ist das Design dieses Chat-Programms und was sind die Verantwortlichkeiten der verschiedenen Klassen?
- Wie viele Remote-Schnittstellen wurden entworfen und was ist der Zweck der Schnittstellen?
- Wie wird sichergestellt, dass sich zwei Chatter zur exakt selben Zeit nicht mit dem gleichen Namen einloggen können (Aspekt der Parallelität)?

Vorgehen

1. Entpacken Sie das ZIP-File (Aufgabe 2) in einem Arbeitsverzeichnis und importieren Sie das Projekt in Eclipse (Import->Existing Project into Workspace) oder einer anderen IDE.
2. Starten Sie die RMI-Registry in einem eigenen Command- bzw. Shell-Prompt im Verzeichnis des entpackten ZIP-Files mit dem Skript `startregistry` (Windows) bzw. `./startregistry.sh` (MacOS, Linux).
3. Öffnen Sie für den Chat-Client und -Server je eine separates Command- bzw. Shell-Prompt und wechseln Sie ins Verzeichnis des entpackten ZIP-Files.
4. Der Chat-Server wird aufgerufen mit: `java –cp bin ChatServer` oder einfach mit dem Skript `server`. Der Chat-Server wird wider beendet mit `Ctrl + C`.
5. Ein Client wird aufgerufen mit: `java –cp bin ChatClient localhost oder <remote ip>` oder einfach mit dem Skript `client localhost` oder `<remote ip>`.
6. Geben Sie einen Benutzernamen (Nickname) im Eingabefeld des Clients ein und loggen Sie sich im Chat ein (Login drücken). Danach kann ein Post bzw. Text im Eingabefeld eingegeben werden (Enter bzw. Return drücken, um ihn zu verschicken).
7. Probieren Sie die Programme aus und schauen Sie sich den Source-Code an.

Hinweise, Tipps

Falls der ChatClient einen ChatServer auf einem anderen Rechner nicht erreichen kann (connection refused), kann der Grund dafür sein, dass das RMI-Programm nicht an eine extern aufrufbare IP-Adresse gebunden wurde. Wenn mehrere Netzwerkadapter zur Verfügung stehen, muss das RMI-Programm (Server oder Client) wie folgt an eine bestimmte IP-Adresse gebunden werden: `java –cp bin –Djava.rmi.server.hostname=<ip address> ChatServer oder ChatClient <remote ip>.`

Der Kommunikationsfluss zwischen Client und Server mit der Registry kann mit dem Aufruf `rmiregistry -J-Djava.rmi.server.logCalls=true` für das Verständnis protokolliert werden (ist im Skript drin).

Falls Sie die Programme ohne IDE (Eclipse o.a.) ändern und neu kompilieren möchten, können Sie dies mit dem Befehl `javac src\*.java –d bin`. Noch einfacher geht es, wenn Sie Ant installiert haben (http://ant.apache.org/). Dann können Sie die Programme kompilieren indem Sie `ant` auf der Kommando-Zeile eingeben☺

Ergebnis

Fassen Sie die Erkenntnisse zu dem Fragen zusammen und diskutieren Sie sie in ihrer Gruppe.

Zeit: 30‘