Praktikum Verteilte Systeme

Aufgabe 7 - Chat-System mit Java RMI

Ziel dieses Praktikums ist es, eine Chat-Anwendung auf Basis von Java RMI zu entwickeln. Dabei wird das Konzept der Proxies an einem praktischen Beispiel erprobt. Es wird. u.a. verwendet, um einen Callback-Mechanismus für die Benachrichtung mehrerer Clients beim Eintreffen von neuen Nachrichten zu realisieren. Für diese Aufgabe gibt es kein in OSCA bereitgestelltes Projekt.
Die Aufgabe muss bis

Montag: 13.05.2019 Dienstag: 14.05.2019 Donnerstag: 16.05.2019

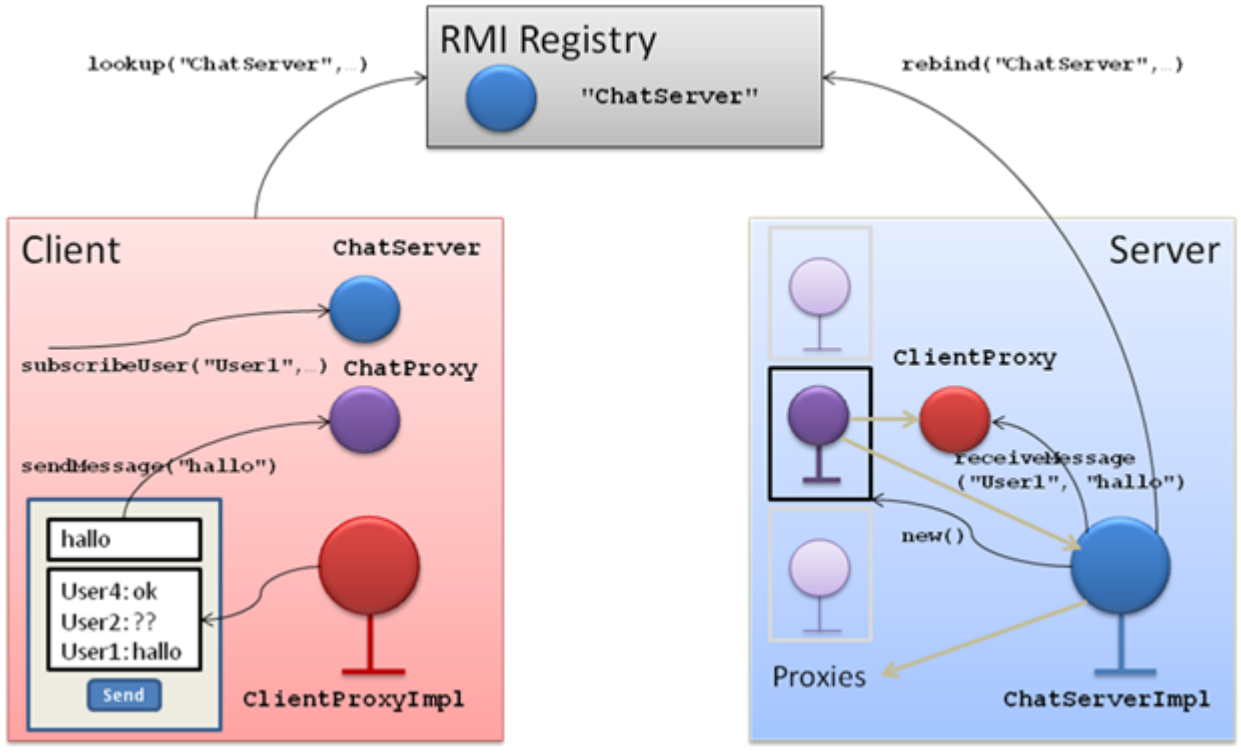
zu 80% fertiggestellt werden.

Beschreibung des Chat-Systems

Das System besteht aus einem Server, an den beliebig viele Clients angeschlossen werden können. Das System soll nach bekanntem Prinzip arbeiten: Nachrichten werden auf einem Client eingegeben und über den Server an alle angeschlossenen Clients weiter geleitet.

Es gibt zwei prinzipielle Möglichkeiten, ein solches System zu realisieren:

1. Die Nachrichten werden vom Server vorgehalten. Verbundene Clients überprüfen in zyklischen Abständen das Vorliegen neuer Nachrichten und rufen diese ab (Polling).
2. Clients registrieren im Server jeweils ein sog. Callback-Objekt. Dieses beinhaltet Methoden zum Empfang und zur Anzeige von Nachrichten. Das folgende Bild veranschaulicht die Struktur:



Sie können von folgenden Klassen ausgehen.

- Die Klasse **`ChatServer`** stellt nach außen eine Methode zur Verfügung, mit der sich Clients beim Server unter Angabe eines Namens und einer Referenz auf das Client-Callback-Objekt (**`ClientProxy`**) registrieren und de-registrieren können:

```
public interface ChatServer extends Remote {
    public ChatProxy subscribeUser (String username,
      ClientProxy handle) throws RemoteException;
    public boolean unsubscribeUser (String username)
        throws RemoteException;
}
```

- **`ClientProxy`** repräsentiert den Client im Server und unterstützt eine Methode zum Empfang von Nachrichten:

```
public interface ClientProxy extends Remote {
      public void receiveMessage (String username,
            String message) throws RemoteException;
}
```

- **`ChatProxy`** repräsentiert den Kommunikationskanal im Client und unterstützt eine Methode zum Versenden von Nachrichten. Die Implementierungsobjekte werden im Server realisiert:

```
public interface ChatProxy extends Remote {
    public void sendMessage (String message)
      throws RemoteException;
}
```

Die entsprechenden Implementierungsklassen realisieren die Schnittstellen-Methoden sowie weitere Methoden. Die Implementierung der Schnittstelle **`ChatServer`** erzeugt in der **`main()`**Routine ein neues **`ChatServerImpl`** Objekt und registriert das Interface in der RMI Registry unter einem frei wählbaren Namen. Daneben benötigen Sie noch einen Client (GUI oder interaktive Shell). Dieser liest eine Referenz auf das **`ChatServer`** Objekt aus der Registry aus und ermöglicht es einem User, sich unter Angabe eines Namens per **`subscribeUser()`** Methode beim Chat zu registrieren.
Dabei wird **`handle`** übergeben, damit der Server beim Eintreffen von neuen Nachrichten **`receiveMessage()`** bei jedem **`ClientProxy`** aller registrierten Clients aufruft.

7.1 Erstellen eines Sequenzdiagramms und Implementierung der Registrierung (5 Punkte)

Zeichnen Sie ein Sequenzdiagramm des Ablaufs beim Registrieren eines Clients Ordnen Sie die Instanzen der Klassen folgendermaßen (von links nach rechts) an:

aClient : ChatClient	aChatProxy : ChatProxy	theRegistry : Registry	***Netz-werk***	theServer : chatServer	theList : NewsProxyList	aChatProxy : ChatProxy

Überlegen Sie, welche Informationen über einen neu registrierten Client Sie im `ChatProxy` auf der Serverseite speichern müssen. Fügen Sie diese als Einträge der `NewsProxyList` zu.

7.2 Implementieren des Nachrichtenversands (4 Punkte)

Implementieren Sie das Chat-System. Testen Sie Ihr System, in dem Sie einen Chat aufsetzen

7.3 Überlegungen zur Bewertung der Verteilungsansätze (1 Punkte)

Ordnen Sie (unabhängig von der Fehlerbehandlung) die Verteilungsansätze in den Aufgaben 5, 6 und 7 ein in Bezug auf ihren Koppelungsgrad (Skala von eng bis lose gekoppelt)