4 Übungsaufgabe #4: Callback

4.1 Allgemeines

Das bisher implementierte Fernaufrufsystem unterstützt bei der Parameterübergabe ausschließlich *Call-by-Value*. In dieser Aufgabe soll nun zusätzlich eine *Call-by-Reference*-Semantik für Objektreferenzen implementiert werden. Dies lässt sich mittels Fernaufrufen realisieren, sofern auf den Zustand eines Objekts nur über Methoden (und nicht direkt auf Member-Variablen) zugegriffen wird.

Entscheidendes Kriterium dafür, ob ein als Parameter transferiertes Objekt *by-Value* oder *by-Reference* übergeben wird, soll das Marker-Interface *VSRemote* (vgl. *java.rmi.Remote*) darstellen:

```
public interface VSRemote {}
```

Implementiert ein Objekt (direkt oder über eine Oberklasse) dieses Interface, so ist es gegebenenfalls *by-Reference* zu übertragen, d. h. ein Methodenaufruf auf Server-Seite führt seinerseits zu einem Fernaufruf an dem beim Client liegenden Objekt. Jeder sonstige Parameter soll wie bisher *by-Value* weitergegeben werden.

4.2 Callback (für alle)

Durch die Einführung von *Call-by-Reference* wird ein Server in die Lage versetzt Rückrufe („Callbacks") an beim Client liegenden Objekten zu tätigen. Dies führt dazu, dass die bisherige Trennung in „Client" und „Server" nicht mehr sinnvoll ist, da beide Einheiten sowohl Client- als auch Server-Funktionalität bereitstellen müssen. Aus diesem Grund soll zunächst eine neuer Objekttyp (*VSRemoteEntity*) bereitgestellt werden, der sich auf beiden Seiten einsetzen lässt. *VSRemoteEntity* könnte demnach zum Beispiel über folgende Methoden verfügen:

```
public class VSRemoteEntity {
    public void init(int serverPort);
    public void exportObject(Object object);
    public Object lookup(String host, int port, Class interfaceClass);
}
```

VSRemoteEntity bietet somit die grundsätzlichen Voraussetzungen für eine Erweiterung der Funktionalität um Rückrufe. Als abschließender Schritt muss nun *Call-by-Reference* ermöglicht werden. Dies beinhaltet im Wesentlichen die Umsetzung zweier Anforderungen:

1. Auf der Seite des Aufrufers muss jeder *by-Reference* zu übertragende Parameter (→ *VSRemote*) in eine Remote-Referenz umgewandelt werden.
2. Auf der Seite des Aufgerufenen muss mittels dieser Remote-Referenz ein Proxy für das betreffende Objekt erzeugt und der lokalen Funktion zur Verfügung gestellt werden.

Aufgaben:

→ Zusammenfassung von *VSClient* und *VSServer* zu *VSRemoteEntity*

→ Integration der Callback-Funktionalität

Hinweise:

- Bei der Implementierung von *VSRemoteEntity* soll auf die bisherigen Klassen *VSClient* und *VSServer* zurückgegriffen werden. (Es ist nicht Sinn der Sache den Code per Copy&Paste zusammenzusetzen!)
- Das Interface *VSRemote* soll ab sofort grundsätzlich Schnittstellen kennzeichnen, die über Fernaufrufe in Anspruch genommen werden können (vgl. *java.rmi.Remote*). Es dient also nicht ausschließlich der Identifizierung der Parameterübergabeart.

4.3 Namensdienst (optional für 5,0 ECTS)

Nachdem in Aufgabe 3 nur eine vereinfachte Funktionalität zum Auffinden von Remote-Objekten umgesetzt worden ist, soll nun ein leistungsfähigerer Namensdienst bereitgestellt werden. Dieser muss über folgende Schnittstelle (semantisch analog zu *java.rmi.Naming*) verfügen:

```
public interface VSNameService {
    public void bind(String name, VSRemote object);
    public String[] list(String name);
    public VSRemote lookup(String name);
    public void rebind(String name, VSRemote object);
    public void unbind(String name);
}
```

Da ein Namensdienst eine Komponente darstellt auf die (unter anderem) per Fernaufruf zugegriffen wird, ist eine Realisierung als Remote-Objekt sinnvoll. Dabei ist zu beachten, dass der Dienst nicht nur von außerhalb (per Fernaufruf), sondern zusätzlich auch lokal aufgerufen werden kann. Entscheidend dafür, welcher Fall benötigt wird, sind die in *name* enthaltenen Host- und Port-Informationen.

Aufgaben:

→ Implementierung eines Namensdienstes als Remote-Objekt

Hinweise:

- Da der Namensdienst auf jeder *VSRemoteEntity* als Remote-Objekt betrieben wird, ist keine zusätzliche Registry mit eigenem Port erforderlich: Die (in *name* anzugebende) Adresse des Namensdienstes soll der Adresse der korrespondierenden *VSRemoteEntity* entsprechen.
- Die Registrierung eines Remote-Objekts (*bind()*) erfolgt nicht notwendigerweise beim Namensdienst des selben Hosts, auf dem es zuvor exportiert wurde.

4.4 Testen der Implementierung (für alle)

Der aktuelle Stand der Implementierung des Fernaufrufsystems soll abschließend durch die Implementierung eigener Testfälle überprüft werden. Dabei ist das in der Tafelübung vorgestellte *JUnit* einzusetzen. Die Gestaltung der konkreten Testfälle ist frei wählbar - grundsätzlich sind zwei Szenarien denkbar:

1. Portierung (und Erweiterung) der Messageboard-Implementierung aus Aufgabe 1
2. Implementierung eines eigenen Testfalls

Unabhängig davon, welche der beiden Möglichkeiten gewählt wird, ist darauf zu achten, dass die implementierten Testfälle ausreichend viele Fehlerszenarien abdecken. Insbesondere ist nicht nur zu überprüfen, ob Methoden bei korrektem Aufruf die richtigen Resultate liefern, sondern ebenfalls, ob falsche oder unsinnige Parameterwerte den Erwartungen gemäß behandelt werden. Dies beinhaltet unter anderem, dass eine von einer Methode geworfene Exception dem Aufrufer in der korrekten Form signalisiert wird.

Aufgaben:

→ Implementierung umfangreicher Testfälle

Hinweise:

- Vereinfachend dürfen bei den Tests alle Entitäten auf dem selben (physikalischen) Rechner gestartet werden.
- Im Fernaufruf selbst begründete Fehlerszenarien (z. B. unerreichbarer Host, Verbindungsabbruch) müssen bei den Tests nicht berücksichtigt werden.
- Im Fokus der Tests steht das implementierte Fernaufrufsystem, **nicht** die Remote-Objekte! Letztere müssen daher so konzipiert bzw. erweitert werden, dass sich mit ihrer Hilfe diverse Szenarien für das Fernaufrufsystem durchspielen lassen.

4.5 Abgabe: am 23.6.2010 in der Übung

Die für diese Teilaufgabe erstellten Testfälle sind in *vsue.tests* abzulegen und alle weiteren Dateien in einem Subpackage *vsue.callback* zusammenzufassen.