Übungsaufgabe #5: CORBA-Client und Server

15.12.2003

In dieser Aufgabe soll das Bibliothekssystem mit Client und Server unter Verwendung von CORBA implementiert werden. Hierzu steht unter `/proj/i4mw/pub/aufgabe5` eine IDL bereit, die zu verwenden ist.
Generell gilt im Folgenden: Alle Client-Klassen sollen im Paket `library.client` abgelegt werden, alle Server-Klassen in `library.server`.

Teil 1: Client

a) Als Client soll der aus den ersten Aufgaben bekannte `LibraryFrontend` dienen; dieser kommuniziert nun mit Hilfe von CORBA mit dem Bibliotheksserver. Es soll wie bisher die Möglichkeit bestehen, ein neues Medium der Datenbank hinzuzufügen, Medien auszuleihen und zurückzugeben und sich alle Medien der Datenbank anzeigen zu lassen. Im Unterschied zu den vorherigen Aufgaben sollen Medien immer durch Angabe des Titels (nicht ID!) angesprochen werden; die Methoden `register()` und `get()` erhalten also als Parameter nun den Titel. Zur Vereinfachung sollen außerdem keine unterschiedlichen Medientypen (CD, Book usw.) unterschieden werden; das Bibliothekssystem kennt nur `Item`s.
Ein einfacher Testserver (IOR in `/proj/i4mw/pub/aufgabe5/LibraryDB.ior`) steht bereit, mit dem der eigene Client getestet werden kann.

b) Im nächsten Schritt soll der Client so erweitert werden, dass er den CORBA-Namensdienst verwendet, um ein `LibraryDB`-Objekt zu bekommen. Der Testserver ist dort unter dem Namen `mw/LibraryDB` registriert. Der eigene Server aus Teil 2 soll unter <login>`/LibraryDB` dort registriert werden (wobei <login> der eigene Login-Name sein soll). Unter `/proj/i4mw/pub/aufgabe5/Nameservice.ior` wird die Adresse des Root-Namensdiensts bekanntgegeben.

Teil 2: Einfacher Server

c) Im zweiten Teil soll nun die Serverseite implementiert werden. Als erster Schritt soll ein ganz einfacher `LibraryDBServant` implementiert werden, der
   - bei `register` eine `AlreadyExistsException` erzeugt,
   - bei `list` eine leere Sequenz von `Items` zurückliefert und
   - bei `get` eine `NotFoundException` erzeugt.

   Die Methoden `lock()` und `unlock()` sind hier nicht nötig, da die Medien über Remote-Referenzen angesprochen werden und nicht wie bisher zum Client kopiert werden.
   Eine Klasse `library.server.Main` soll einen `LibraryDBServant` erzeugen, und ihn beim Namensdienst als <login>`/LibraryDB` anmelden. Der Servant kann nun mit dem eigenen Client getestet werden.

d) Im nächsten Schritt soll nun ein Servant für die Medien (`ItemServant`) implementiert werden. Die `ItemServant`-Objekte werden in der Methode `register()` erzeugt. Der `LibraryDBServant` kann nun entsprechend so erweitert werden, dass er die Medien in einem `Vector` speichert, und die in (c) beschriebenen Methoden sinnvoll implementiert.

Teil 3: Besserer Server

Da eine Bibliothek gewöhnlich eine große Anzahl an Medien enthält, ist es nicht sinnvoll, für jedes Medium einen aktiven Servant zu erzeugen. Der Server sollte daher nur einen Cache mit den zuletzt verwendeten Medien verwalten, die persistente Speicherung der Daten der Medien wird zweckmässigerweise in einer Datenbank vorgenommen. Hierzu wird eine einfache Pseudo-Datenbank unter `library.database.ItemDatabase` zur Verfügung gestellt.

e) Als erstes ist die Main-Methode anzupassen: Für die Implementierung des Caches muss ein eigener POA (`ItemPOA)` unterhalb des RootPOA erstellt werden (Policies: `USER_ID`, `USE_SERVANT_MANAGER` und `NON_RETAIN`). Für das dynamische Erzeugen von Medien-Servants muss beim `ItemPOA` ein Servant-Manager (`ItemServantLocator`) registriert werden.

f) Nun muss der `ItemServantLocator` implementiert werden:
- Als erstes braucht man eine eindeutige ID für Medien. Diese kann direkt aus dem Index in der Datenbank gewonnen werden kann (Konvertierung zwischen `int` und `byte[]`!)
- In der `preinvoke`-Methode soll der `ServantLocator` testen, ob es zu der angeforderten ID bereits einen `ItemServant` im Cache gibt. Wenn vorhanden, wird dieser verwendet, ansonsten wird ein neuer `ItemServant` mit den Daten aus der Datenbank erzeugt. Der Cache kann sehr einfach verwaltet werden (Hash über ID oder FIFO). Zum Konvertieren von CORBA-Referenzen in Objekt IDs kann die POA-Methode `reference_to_id` (von dem POA, der die CORBA-Referenz erzeugt hat, siehe (g)) verwendet werden.
- In der `postinvoke`-Methode muss, falls das Medium verändert wurde (Ausleihstatus, Ausleihzähler) dieses in die Datenbank zurückgeschrieben werden.

g) Zuletzt sind die Methoden `get` und `list` im `LibraryDBServant` anzupassen. Diese soll direkt über die Datenbank die IDs der passenden Medien ermitteln. Aus diesen IDs kann über den `ItemPOA` (mittels `create_reference_with_id`) eine CORBA-Objektreferenz erzeugt werden, die als Rückgabe der Methoden verwendet werden kann. Hierzu müssen an dieser Stelle keine Daten aus der Datenbank gelesen und auch keine Servants erzeugt werden (das soll später dynamisch durch den `ItemServantLocator` geschehen)!

<u>Bearbeitung:</u> bis zum 15.01.2004/18:00 Uhr
Alle Dateien sollen im Verzeichnis `/proj/i4mw/`*`loginname`*`/aufgabe5/` abgelegt und mit dem `abgabe`-Programm abgegeben werden.
Die Bearbeitung in 2er Gruppen möglich.

**Wir wünschen Ihnen ein schönes Weihnachtsfest
und einen guten Rutsch in das Jahr 2004**