JUST DO IT
HÖHERE ABTEILUNG FÜR INFORMATIONSTECHNOLOGIE

tgm
Die Schule der Technik

Laborprotokoll

DezSys03: Remoting Patterns

Systemtechnik Labor
5BHITT 2015/16, Gruppe X

Stefan Erceg

Version 1.0

Note:

Betreuer: Prof. Borko

Begonnen am 16. Oktober 2015

Beendet am 16. Oktober 2015

Inhaltsverzeichnis

1 Einführung

Diese Übung zeigt die einfache Implementierung von verteilten Anwendungen.

1.1 Ziele

Das Ziel dieser Übung ist die Vertiefung in die Thematik der Implementierung verteilter Anwendungen. Es soll durch das Einarbeiten in schon bestehenden Sourcecode auch die Wiederverwendung von Software veranschaulicht werden - mit all den positiven und negativen Aspekten.

Wichtige Elemente der verteilten Programmierung soll anhand der Identifizierung der vorgestellten Remoting Patterns geschult werden.

1.2 Voraussetzungen

- Grundlagen der verteilten Programmierung
- Verwendung von SCM-Tools
- Erweiterte Kenntnisse von SW-Entwicklungsprozessen
- Verständnis der Basic Remoting Patterns

1.3 Aufgabenstellung

Analysieren Sie die mitgelieferte Implementation (siehe Resources) der verteilten LeelaApplikation. Identifizieren Sie dabei alle verwendeten Elemente der "Basic Remoting Patterns" im Code und erstellen Sie UML-Klassendiagramme für die Pakete evs2009, comm und mapping.

Schließen Sie die unfertigen Tests ab, und dokumentieren Sie etwaige Schwierigkeiten. Schreiben Sie zumindest zwei neue Testfälle für entsprechend wichtige Teilaspekte der Implementierung und begründen Sie Ihre Entscheidung. Beurteilen Sie die Einsatzmöglichkeit dieser Implementierung und führen Sie etwaige Verbesserungsvorschläge an.

Was ist zu tun?

- Analyse
- Erweitern der Testfälle
- Kritik und Verbesserungsvorschläge

Protokollieren Sie alle notwendigen Arbeitsschritte zur Inbetriebnahme der Software!

Bewertung: 8 Punkte

Software-Analyse und Ablaufbeschreibung ... 1Pkt
Identifikation und Beschreibung der Basic Remoting Patterns ... 1Pkt
UML-Diagramme ... 1Pkt
Testfälle ... 2Pkt
konstruktive Verbesserungsvorschläge / Kritikpunkte ... 3Pkt

2 Ergebnisse

2.1 Software-Analyse & Ablaufbeschreibung

„Die Applikation wird durch die Auswahl eines Peers (Name) gestartet. Da kein NamingService verwendet wird, sondern eine statische Liste an teilnehmenden Peers (peers.csv), welche durch einen PeerReader ausgelesen werden kann, muss dieser Name auch vorher definiert werden. Die Anforderung der ACID-Implementierung wird durch die Instanzen des TransactionManager und des SessionPeer gelöst. Ein entfernter Peer meint, seine Anfragen sofort an den Peer zu leiten, jedoch fängt diese der SessionPeer ab. Dieser kommuniziert dann mit dem TransactionManager, der Befehle erst nach erfolgreichem Endstatus an die Resources weiterleitet. Die Befehle werden in einer Liste zwischengespeichert.

Die Kommunikation schließt nun folgende Klassen und Interfaces ein:

➢***AbsoluteObjectReference*** *hat notwendige Informationen eines Peers, wie Protokoll und Bestimmungsort. Das AOR wird vom Requestor verwendet.*

➢***Lookup*** *liefert das AOR eines Peers zurück, das durch dessen Name indentifizierbar ist.*

➢***Requestor*** *bietet ein dynamisches Interface zum Aufruf von Methoden über den RequestHandler an.*

➢***RequestorHandler*** *arbeitet als Schnittstelle zwischen dem lokalen Peer und den Anfragen von entfernten Peers. Dabei nutzt der RequestHandler den Invoker für die einzelnen Server Instanzen.*

➢***Invoker*** *bietet die Methode handleRequest(byte[]), welche eingehende Anfragen abarbeitet.*

➢***ProtocolPluginServer*** *wird als Interface in den einzelnen Plugins implementiert und bearbeitet die eingehenden Aufrufe. Die einzelnen Protokolle werden beim Aufruf des Invokers instanziert und konfiguriert.*

➢***ProtocolPluginClient*** *ist als Interface in den Protokollen als Schnittstelle nach außen vorgesehen. Durch das AOR wird der richtige Requestor ausgewählt und verwendet um eine Anfrage an einen entfernten Peer zu senden.“ [1]*

2.2 UML-Diagramme

2.2.1 Package „evs2009“

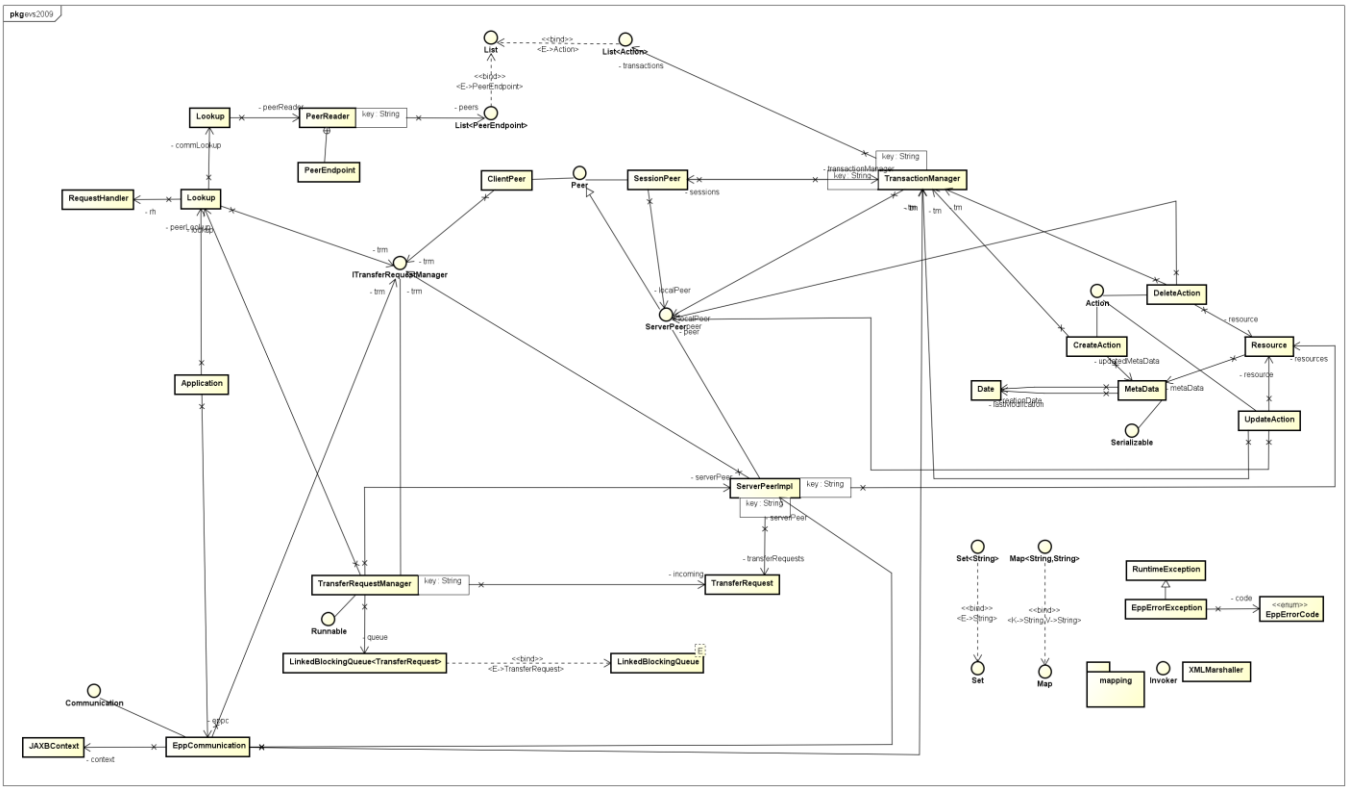

Abbildung 1: Package „evs2009“ [Abb1]

2.2.2 Package „comm“

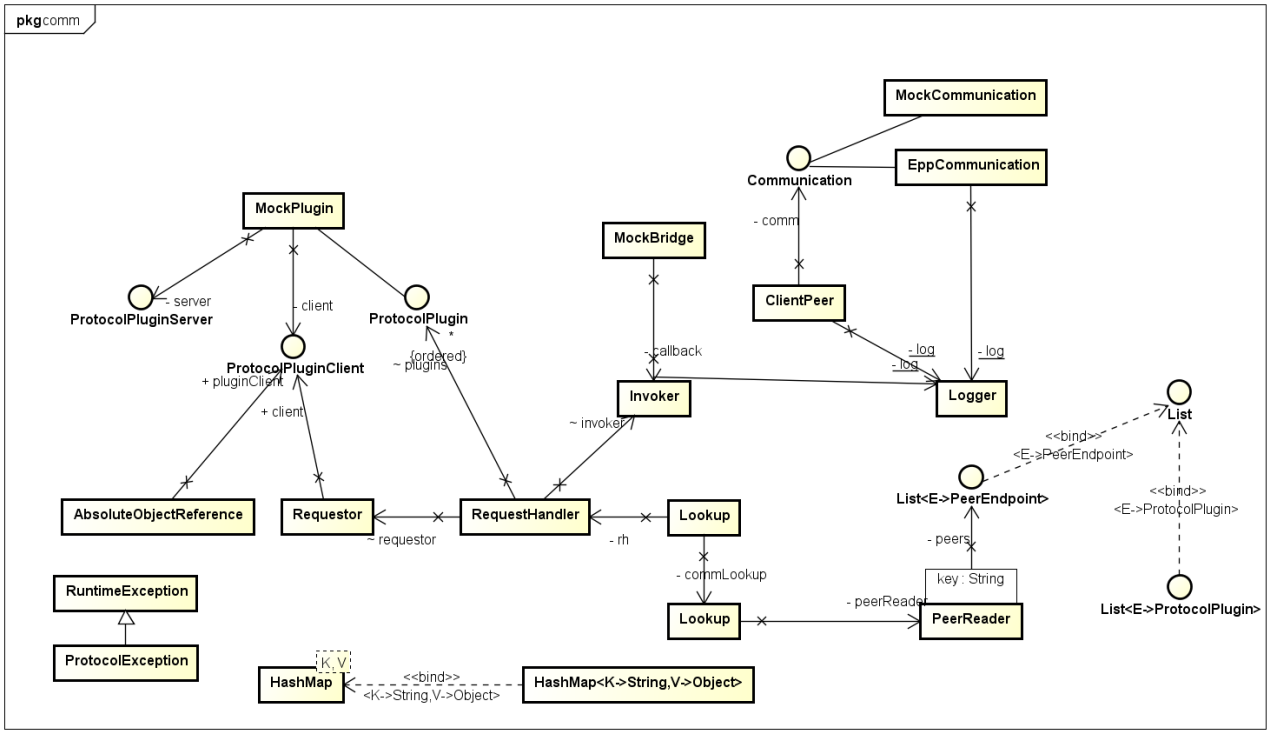

Abbildung 2: Package „comm“ [Abb2]

2.2.3 Package „mapping“

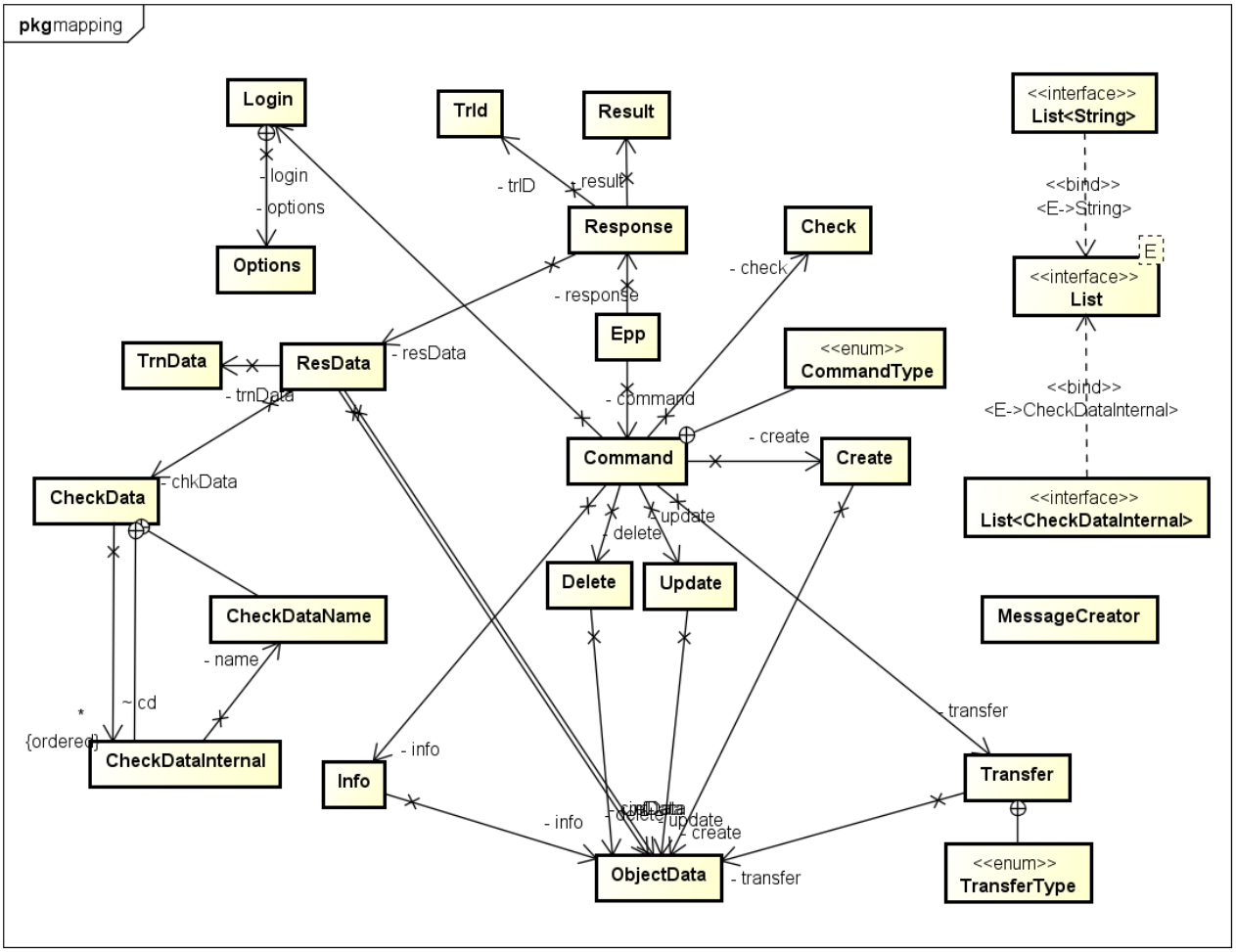

Abbildung 3: Package „mapping“ [Abb3]

2.3 neue Testfälle

Folgende von Janeczek & Mair neu geschriebenen Testfälle [2] wurden zum Projekt hinzugefügt:

- Der Test „correctDoubleLogin()" wurde in dem AuthenticationTests.java-File getestet:

```
/*
 * A method which tests what happens when there is someone going to login twice
*/
@Test
public void correctDoubleLogin() throws Exception {
	peer.login(Helper.correctPassword, Helper.correctPassword);
	peer.login(Helper.correctPassword, Helper.correctPassword);
	peer.logout();
}
```

- Die Tests „correctCreationAndReadZero()" und "correctCreationAndReadloop" wurden in dem CRUDTests.java-File getestet:

```
/*
* A method to test the reaction of the server when he receives a null object
*/
@Test
public void correctCreationAndReadZero() {
	String identifier = null;
	insertObject(identifier);
	byte[] readBytes = serverPeer.read(identifier);
	assertEquals(getBytes().length, readBytes.length);
	assertEquals(testString, new String(readBytes));
}


/*
* A method to test the reaction of the server when he receives a lot of read
tasks at once
*/
@Test
public void correctCreationAndReadloop() {
	String identifier = "Wow";
	insertObject(identifier);
	byte[] readBytes = null;
	for(inti = 0; i<500; i++){
		readBytes = serverPeer.read(identifier);
	}
	assertEquals(getBytes().length, readBytes.length);
	assertEquals(testString, new String(readBytes));
}
```

2.4 konstruktive Verbesserungsvorschläge / Kritikpunkte

2.4.1 Testcase schlägt fehl

„Nach dem Entpacken funktioniert das Target ant test/ant nicht(debug output wurde hinzugefügt):

```
[junit] 70   [main] DEBUG comm.socket.SocketPluginClient  - Got bytes838
[junit] 70   [main] DEBUG comm.socket.SocketPluginClient  - Read is 838
[junit] before: Fri Sep 26 09:43:06 CEST 2014
[junit] after: Fri Sep 26 09:43:06 CEST 2014
[junit] between: Fri Sep 26 09:43:06 CEST 2014
[junit] ------------ ---------------- ---------------
[junit]
[junit] Testcase: generalTest took 0,21 sec
[junit]     FAILED
[junit]
[junit] junit.framework.AssertionFailedError:
[junit]     at evs2009.ApplicationTest.check(ApplicationTest.java:91)
[junit]     at evs2009.ApplicationTest.generalTest(ApplicationTest.java:33)
[junit]
```

Um das zu fixen, müssen die Aufrufe von check(..) in den Zeilen 33, 45, 47 in der Datei ApplicationTest.java auskommentiert werden." [3]

2.4.2 Kein ordentliches Exceptionhandling

„In diversen Dateien, z.B. PeerReader.java findet kein ordentliches Exceptionhandling statt. Die Exceptions werden zwar abgefangen, der Stacktrace jedoch direkt wieder ausgegeben - keine custom exceptions, kein Logging.

Falls eine Exception auftritt, sollte diese Entweder eine eigene Exception (welche später abgefangen wird) auslösen, oder ein Logging Tool (z.B. Log4j) verwendet werden." [3]

2.4.3 Dokumentation unvollständig gelöscht

„Die Dokumentations-Files welche wohl hätte von uns verborgen werden sollen waren noch über die Git-History auffindbar. Mit den folgenden Befehl wären die Dateien tatsächlich vollständig gelöscht worden:

```
git filter-branch \
--index-filter 'git rm --cached --ignore-unmatch \
README \
documentation/evs028.odt \
documentation/evs028.pdf \
documentation/evsCore.jpg \
documentation/pkgComm.jpg \
documentation/pkgMapping.jpg \
' d0f074f4a20f6b8b68c0ee80b1646e992d8c09ac..HEAD
```

`d0f074f4a20f6b8b68c0ee80b1646e992d8c09ac` *ist hierbei der erste commit.*“ [3]

3 Quellen

[1] Borko, Greifeneder, Motlik (Juni 2009).
Entwurfsmethoden für Verteilte Systeme (Seite 3 von 9).
[zuletzt abgerufen am 16.10.2015]

[2] Janeczek, Mair (Oktober 2014). RemotingPatterns [Online].
Available at: https://github.com/cjaneczek-tgm/RemotingPatterns
[zuletzt abgerufen am 16.10.2015]

[3] Willinger, Klepp (September 2014). RemotingPatterns [Online].
Available at: https://github.com/awillinger/RemotingPatterns
[zuletzt abgerufen am 16.10.2015]

4 Abbildungsverzeichnis

[Abb1] Janeczek, Mair (Oktober 2014). RemotingPatterns [Online].
Available at: https://github.com/cjaneczek
tgm/RemotingPatterns/blob/master/UML/evs2009.png
[zuletzt abgerufen am 16.10.2015]

[Abb2] Janeczek, Mair (Oktober 2014). RemotingPatterns [Online].
Available at: https://github.com/cjaneczek-
tgm/RemotingPatterns/blob/master/UML/comm.png
[zuletzt abgerufen am 16.10.2015]

[Abb3] Janeczek, Mair (Oktober 2014). RemotingPatterns [Online].
Available at: https://github.com/cjaneczek-
tgm/RemotingPatterns/blob/master/UML/mapping.png
[zuletzt abgerufen am 16.10.2015]