2/10/2014

Christian Janeczek, Wolfgang Mair

5AHITT

DEZSYS01

REMOTING PATTERNS

Contents

[Task Description 2](#_Toc400051010)

[Design consideration 3](#_Toc400051011)

[REQUIREMENT: Identification of Remoting Patterns Usage 3](#_Toc400051012)

[REQUIREMENT: Description of the application 4](#_Toc400051013)

[REQUIREMENT: UML-Diagrams 5](#_Toc400051014)

[REQUIREMENT: Writing a new Test Case 8](#_Toc400051015)

[REQUIREMENT: Improvement Suggestions / Critique 9](#_Toc400051016)

[Apportionment of work with effort estimation 10](#_Toc400051017)

[Final Time Apportionment 11](#_Toc400051018)

[Bibliography 12](#_Toc400051019)

# Task Description

Das Framework für Remoting Patterns finden sie unter dem Thema "Resources"!  
  
Gruppenarbeit: 2 Mitglieder (Server/Client)  
  
Analysieren Sie in einer Gruppe von 2 Leuten die mitgelieferte Implementation der verteilten LeelaApplikation. Identifizieren Sie dabei alle verwendeten Elemente der "Basic Remoting Patterns" und erstellen Sie UML-Klassendiagramme für die Pakete comm, comm.socket, comm.soap, evs2009 und evs2009.mapping  
  
Schließen Sie die unfertigen Tests ab, und dokumentieren Sie etwaige Schwierigkeiten.  
  
Was ist zu tun?

* UML Klassendiagramm
* Erweitern der Testfälle (mind. einen Testfall erweitern)
* Kritik und Verbesserungsvorschläge

*Punkte (16):*

*Identifikation von Basic Remoting Patterns ... 1Pkt  
Beschreibung der Applikation ... 4Pkt  
UML-Diagramme ... 3Pkt  
Schreiben von einem neuen Testfall ... 2Pkt  
konstruktive Verbesserungsvorschläge / Kritikpunkte ... 6Pkt*

# Design consideration

## REQUIREMENT: Identification of Remoting Patterns Usage

The usage of Basic Remoting Patterns in this software design cannot be denied. As mentioned in the documentation, the application was built upon the Broker-Pattern, which doesn't seem to be entirely true. To us it seems like, there has been some kind of merging going on. Not only has the Broker-Pattern been utilized, but also initial parts of the RequestHandler Pattern.

In this case, we have a newborn child with genes of the Father (Broker) as well as the Mother (RequestHandler). The name-choosing is your special task.

## REQUIREMENT: Description of the application

„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.

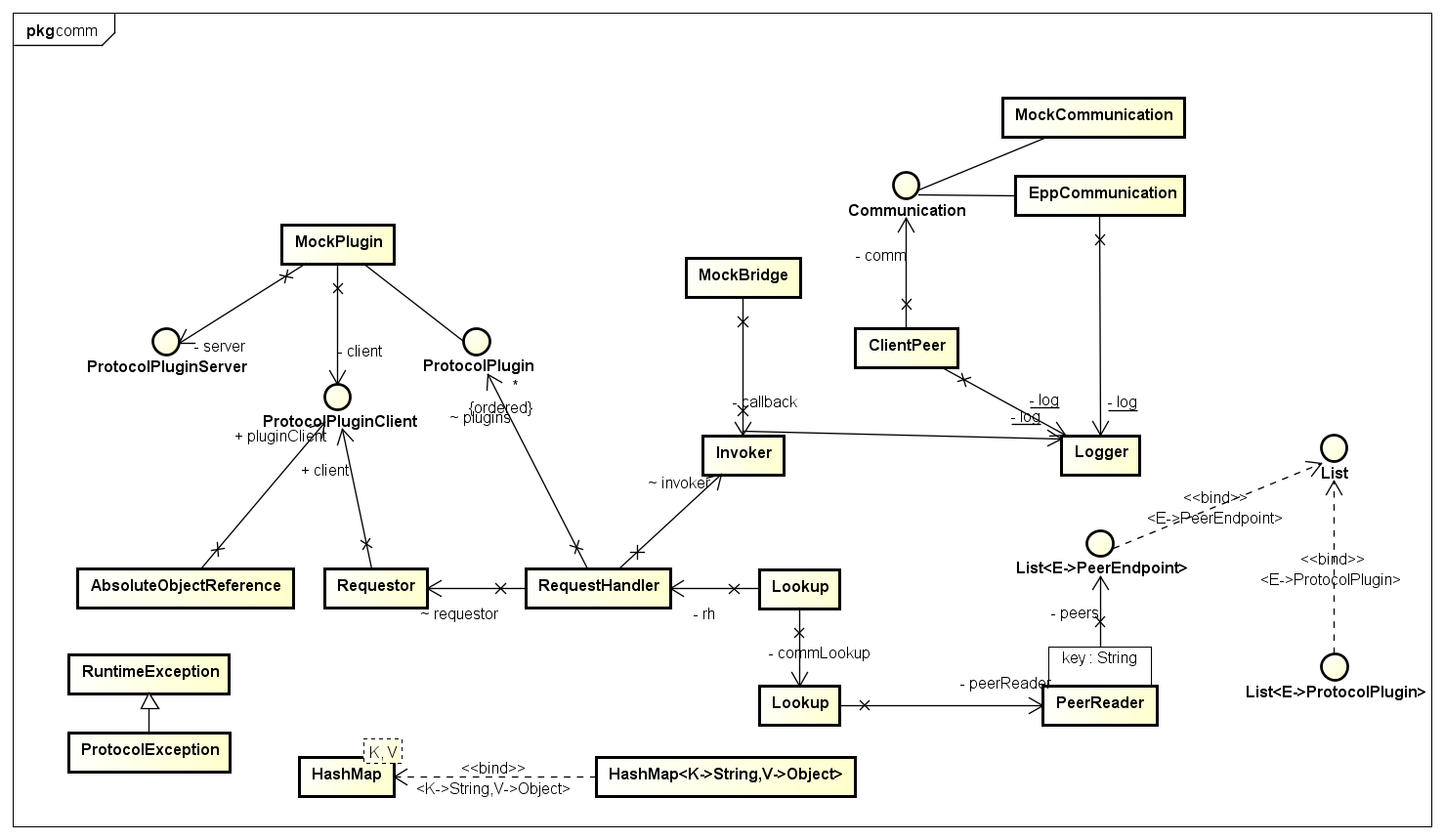
The German explanations have been taken from the **evs028.pdf** documentation presented to you by:

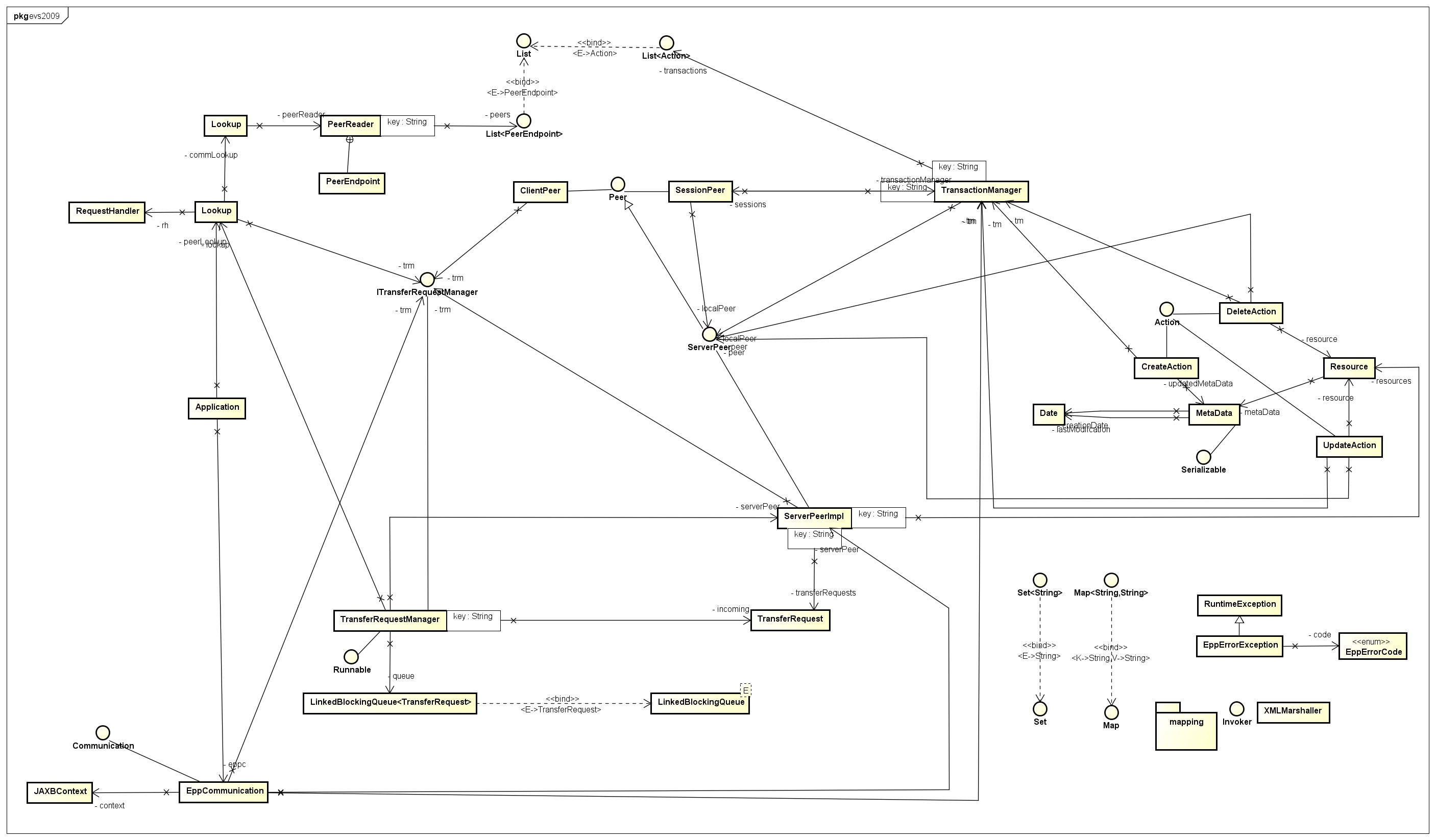
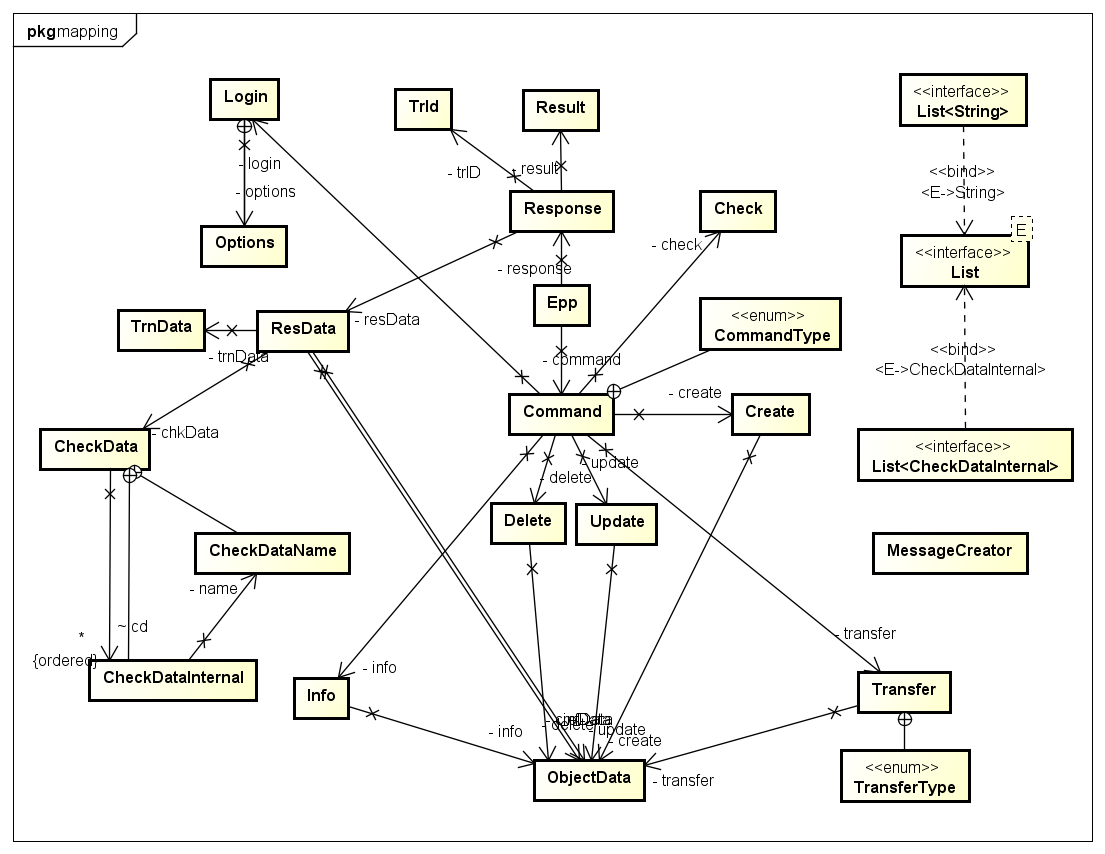
© Borko, Greifeneder, Motlik Seite 3 von 9 07.06.2009

Entwurfsmethoden für Verteilte Systeme evs028

## REQUIREMENT: UML-Diagrams

The UMLs are located right heeeeeeere



This UML is way too bigThis UML ... I like it

## REQUIREMENT: Writing a new Test Case

This Test was tested in the AuthenticationTests class.

/\*

\* 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();

}

These Tests were tested in the CRUDTests class.

/\*

\* 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**(**int** i = 0; i<500; i++){

readBytes = serverPeer.read(identifier);

}

*assertEquals*(getBytes().length, readBytes.length);

*assertEquals*(testString, **new** String(readBytes));

}

I am a happy new test case and this is my story….

Once upon a time there was a test case

It was written by Wolfgang Mair and lived happily ever after

THE END

## REQUIREMENT: Improvement Suggestions / Critique

* Insufficient documentation/comments inside the code (Zeilenkommentare)
  + It is hard to understand what exactly is going on, easily getting overwhelmed
* If there should be any libraries, which aren’t provided in a library folder, there should be some kind of notification
  + Maybe some kind of download-link and summary of the needed libraries might be useful
* AuthenticationTests.java is not completely bug-free
* Unfortunately there happen to be unhandled warnings (more than just once!)

# Apportionment of work with effort estimation

|  |  |  |  |
| --- | --- | --- | --- |
| **Competent person(s)** | **Task** | **Description** | **Estimated time in h** |
| Janeczek | Design consideration | Which Basic Remote Patterns have been used? | 2 |
| Mair | Design consideration | Creating the UML diagrams | ? |
| Janeczek, Mair | Description of the application | Describing the application | 1 |
| Mair | Test Case | Implementing a new Test Case | 2 |
| Janeczek, Mair | Brainstorming: Improvement Suggestions, Critique | Implementing the State Pattern Solution | 2 |

# Final Time Apportionment

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Competent person(s)** | **Task** | **Estimated time in h** | **Actual time in h** | **Comment** |
| Janeczek | Design consideration | 2 | 1 | Janeczek |
| Mair | Design consideration | ? | ? | Mair |
| Janeczek, Mair | Description of the application | 1 | 1 | Janeczek, Mair |
| Mair | Test Case | 2 | 2 | Mair |
| Janeczek, Mair | Brainstorming: Improvement Suggestions, Critique | 2 | 2 | Janeczek, Mair |

# Bibliography

|  |
| --- |
| **Title:** Communication Framework  **Author:** Borko Michael, Greifeneder Michael, Motlik Florian  **Source:** evs028.pdf  **Last modified:** 2009/06/07  **Last seen:** 2014/10/02 |
|  |