Universität Bremen
FB 3 – Informatik
Dr. Karsten Hölscher
Tutor: Karsten Hölscher

Software–Projekt 2 2014

VAK 03-BA-901.02

Testplan

Patrick Hollatz	phollatz@tzi.de	2596537
Tobias Dellert	tode@tzi.de	2936941
Tim Ellhoff	tellhoff@tzi.de	2520913
Daniel Pupat	dpupat@tzi.de	2703053
Olga Miloevich	halfelv@tzi.de	2586817
Tim Wiechers	tim3@tzi.de	2925222

Abgabe: 06. Juli 2014 — Version 1.1

Inhaltsverzeichnis

Version und Änderungsgeschichte

Version	Datum	Änderungen
1.0	04.07.2014	Dokumentvorlage als initiale Fassung kopiert
1.1	05.07.2014	Einführung, Systemüberblick, Merkmale, Abnahme- und Testendekriterien
1.2	06.07.2014	Testplan abgabefertig

Wichtiger Hinweis: Dieser Testplan wurde zu einem Teil aus verschiedenen Dokumententeilen von dem Testplan unserer Gruppenmitglieder aus dem Wintersemester 2013/14 erstellt (Gruppe *IT_R3V0LUTION*). Einige Teile wurden komplett übernommen, andere überarbeitet bzw. angepasst.
Diese Vereinbarung haben wir in der Kick-Off-Veranstaltung für RE SWP 2014 mit dem Veranstalter Dr. Karsten Hölscher getroffen.

1 Einführung

bearbeitet von: Daniel Pupat

1.1 Zweck

Der Testplan bietet einen Überblick über die geplanten Tests und dient u.a. als Anleitung für die Tester. Die Software soll dabei ausführlich auf Funktionalität getestet werden.

Im Testplan wird festgelegt, wie man welche Komponenten testet. Dazu wird außerdem definiert, welchen Umfang die Tests haben sollen und wann ein Test erfolgreich ist und wann nicht.

Während der Implementierungsphase werden wir uns nach dem Testplan richten und ihn gegebenenfalls weiterführen und vervollständigen.

1.2 Umfang

Der Testplan entspricht der vereinfachten Form des *IEEE Standard for Software Test Documentation 829-1998*.

1.3 Beziehungen zu anderen Dokumenten

Dieser Testplan bezieht sich zum einen auf die Anforderungsspezifikation, da dort die Systemeigenschaften und Systemattribute spezifiziert wurden. Die Testfälle werden auf Grundlage der dortigen Anwendungsfälle entwickelt.

Außerdem gibt es Referenzen zur Architekturbeschreibung, da in dieser die Module und Komponenten definiert wurden, die in diesem Dokument getestet werden sollen.

1.4 Aufbau der Testbezeichner

Der Aufbau der Testbezeichner richtet sich nach folgendem Schema:

- Die ersten beiden Buchstaben geben die Art des Tests vor. Dabei unterscheiden wir zwischen vier verschiedenen Testarten:
  - Komponententests = KT
  - Integrationstests = IT
  - Funktionstests = FT
  - Leistungstests = LT
- Die Nummer steht für die jeweilige Testfallnummer
- Optional: in alphabetischer Reihenfolge werden hier Variationen oder untergeordnete Testfälle definiert

Nach diesem Schema sieht ein Testbezeichner nun folgendermaßen aus:

IT-3-A: Integrationstest, Nr. 3, Variante 1

1.5 Dokumentation der Testergebnisse

Zu jedem Testfall wird ein kurzes Testprotokoll angefertigt. Dieses beinhaltet den Ablauf des Testfalls und die möglichen Komplikationen, die bei der Durchführung entstehen können. Dann werden die Resultate des Testfalls bestimmt und eventuell gefundene Fehler beschrieben.

1.6 Definitionen und Akronyme

1.7 Referenzen

IEEE Standard for Software Test Documentation 829-1998
http://standards.ieee.org/findstds/standard/829-1998.html

2 Systemüberblick

bearbeitet von: Daniel Pupat

Das System besteht aus der Server- und der Clientkomponente. Die konzeptionelle Sicht der Architekturbeschreibung (vgl. Abschnitt 3 der Architekturbeschreibung) dient als Grundlage für den Testplan, da dort die verschiedenen Komponenten beschrieben werden.

Auf der Serverseite gibt es die Komponenten `Communication, BusinessLogic` und `Persistence` (vgl. Abbildung 3: Konzeptionelle Sicht Server; Architekturbeschreibung).

Die Clientseite besteht aus den Komponenten `Communication, Model` und `User Interface` (vgl. Abbildung 4: Konzeptionelle Sicht Client; Architekturbeschreibung).

Da starke Abhängigkeiten zwischen all diesen Komponenten bestehen, ist es wichtig, dass diese Komponenten fehlerfrei funktionieren.

2.1 Module der Anwendungsschicht und deren Funktionen

In der nachfolgenden Tabelle werden die Module verfeinert, die in Punkt 5 visualisiert sind.

GUI	webapp(xhtml)
AndroidApp	com.mygdx.game, com.mygdx.game
Communication	common.net
UserInterface	GUI AndroidApp
Server	Persistence, ServerInbox, ServerDirectory, Server,
Common	entities, common.net
Persistence	Data.java, IPersistence.java

3 Merkmale

bearbeitet von: Daniel Pupat

Zu testende Merkmale sind in erster Linie Funktionen, welche alle Anwendungsfälle abdecken, die die Kundin sich gewünscht haben. Dabei muss sowohl die App, als auch die Website getestet werden.

1. Online

- 1.1 Benutzer registrieren
- 1.2 Benutzer anmelden
- 1.3 Benutzer abmelden
- 1.4 Gegner herausfordern
- 1.5 Angefangenes Spiel weiterspielen
- 1.6 Frage beantworten
- 1.7 Einstellungen ändern

2. Offline

- 2.1 Neues Spiel starten
- 2.2 Frage beantworten

3.Website

- 3.1 Admin anmelden
- 3.2 Admin abmelden
- 3.3 Frage hinzufügen
- 3.4 Frage bearbeiten
- 3.5 Frage löschen
- 3.6 Frageliste importieren
- 3.7 Frageliste exportieren
- 3.8 User löschen
- 3.9 Passwort ändern

3.0.1 Funktionale Anforderungen

Besonders wichtig ist der Spielvorgang bei der App, da die Nutzer diesen hauptsächlich nutzen werden. Dabei muss darauf geachtet werden, dass der Nutzer sowohl online als auch offline spielen kann. Auch das Spielen gegen eine andere Person muss getestet werden.

3.1 Nicht zu testende Merkmale

Da wir ein komplett neues Projekt erstellen, ist es wichtig, dass alle Merkmale getestet werden. Sollte es triviale Funktionen geben, müssen diese nicht getestet werden.

4 Abnahme- und Testendekriterien

bearbeitet von: Daniel Pupat

Fehler werden in eine Kategorie eingeordnet und erhalten entsprechende Fehlerwerte. Aus diesen Fehlerwerten ergeben sich Prioritäten, welche die Reihenfolge der Fehlerbehandlung angeben. Das Testen wird beendet, wenn der berechnete Fehlerwert aller Fehler pro 1000 Zeilen Code unter dem Wert 10 liegt und die Software nicht beeinträchtigt wird, d.h. es keinen Fehler der Fehlerklasse `Mittel` oder höher gibt.

Testabdeckung Die Testabdeckung soll so hoch wie möglich sein. Für ein stabiles System spricht, dass die Testabdeckung in systemkritischen Bereichen soweit vollständig ist. Jeder Fehler in diesem Bereich kann das System zum Absturz bringen und muss somit verhindert werden. In anderen Bereichen, die das laufende System bei einem Fehler weniger beeinträchtigen, wird die Testabdeckung nicht so vollständig sein, wie in kritischen Bereichen.

Fehlerbewertung:
Die nachfolgende Tabelle spezifiziert die Auswirkung eines Fehlers, durch die man diese nach Priorität einordnen kann.

Fehlerkl.[1]	Beschreibung	Wert
Leicht	Unwesentliche Fehler, die den Programmablauf nicht beeinträchtigen, aber trotzdem behandelt werden sollten.	1
Mittel	Fehler in dieser Art haben Auswirkungen auf den Programmablauf. Dieser beeinträchtigt aber nicht die grundlegenden Funktionen.	10
Schwer	Fehler der Klasse ,,Schwer" beeinträchtigen die Funktionsfähigkeit des Systems sehr stark und müssen sofort behandelt werden.	20
Fatal	Diese Fehler machen den Programmablauf unmöglich und können zum Absturz des Systems führen.	100

5 Vorgehensweise

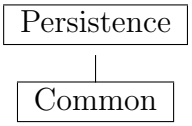
bearbeitet von: Daniel Pupat

5.1 Komponenten- und Integrationstest

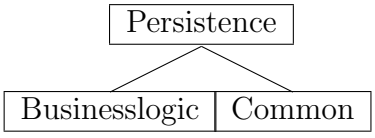
Die beiden Komponenten Server und Client werden bezüglich der Integrationstests zunächst unabhängig voneinander getestet und erst bei wenn das sichere Laufen der einzelnen Komponente gewährleistet ist, wird das System als ganzes getestet.

Server

Es wird zuerst die Persistence mit `Common` und `Persistence` getestet:

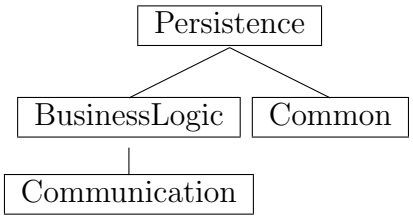


Nun folgt das Zusammenspiel mit der BusinessLogic `BusinessLogic`:



Darauf folgt nun die Kommunikation `Communication` mit den vorigen Komponenten:

[1]=Fehlerklasse

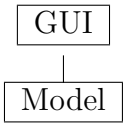


Somit ist der Server als ganzes zu testen, da jede einzelnen Komponenten mit ihren jeweiligen Abhängigkeiten getestet wurden.

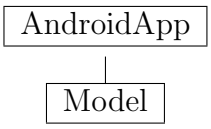
Client

Hier sind nun zwei verschiedene Komponenten für die Darstellung auf dem jeweiligen Endgerät vorhanden:

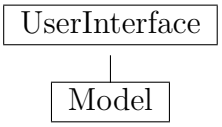
Einmal `GUI` welche zusammen mit dem Model getestet wird und sich an die Browserdarstellung richtet:



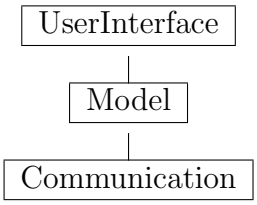
Und einmal `Android App` mit dem Model, welche sich an mobile Geräte richtet:



Diese beiden Tests werden zusammengefasst zu `User Interface` und zusammen mit der nächsten Ebene, dem Model getestet:



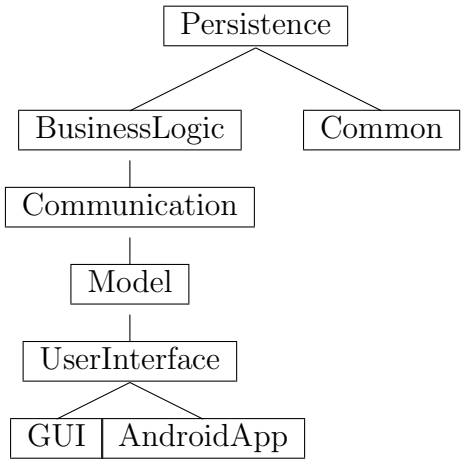
Darauf folgt nun der Test mit der Komponente `Communication`:



Nun ist das Zusammenspiel der Komponenten des Clients gewährleistet.

Server + Client

Um die kompletten Systemkomponenten zu testen werden jetzt der komplette Client und der komplette Server zusammen getestet:



5.2 Funktionstest

Die Funktionstests sind durch jene Anwendungsfälle aus der Anforderungsspezifikation vorgegeben. Jede dieser Funktionen muss durch Tests gedeckt sein.

6 Aufhebung und Wiederaufnahme

bearbeitet von: Daniel Pupat

Wir werden Tests unterbrechen, wenn ein gewisser Wert überschritten wird, welcher über die Tabelle in Abschnitt 4 berechnet wird. In diesem Fall werden wir sofort wieder mit der Implementierung anfangen. Da wir mit der Bottom-up Strategie testen, werden wir bei Fehlern in der unteren Schicht einen niedrigeren Wert nehmen.
Bei Fehlern der Data setzen wir einen Wert von 10, bei Fehlern in der Logik einen Wert

von 20 und bei Fehlern, welche die GUI betreffen, einen von 40 und bei den restlichen Faktoren einen von 100.
Sollten die Fehler behoben sein, testen wir noch einmal alle Komponenten, die mit den veränderten interagieren.

7 Hardware- und Softwareanforderungen

bearbeitet von: Daniel Pupat

7.1 Hardware

Als Hardware stehen uns unsere Notebooks und Smartphones, sowie die Unirechner zur Verfügung. Dabei haben wir alle geforderten Betriebssysteme mindestens einmal auf unseren Notebooks installiert, sodass wir auf jeden Gerät testen können. Da Android Unterstützung gefordert ist, werden wir die App über unseren vorhandenen Smartphones, die Android haben, testen. Andere Systeme (iOS) werden wir testen, falls noch genügend Zeit vorhanden ist.

7.2 Software

Als Software benutzen wir in der Eclipse Umgebung JUnit-Tests. Diese werden in Form von BlackBox- und WhiteBox-Tests implementiert. Die App werden wir mithilfe eines Android Emulators und unseren Smartphones testen.

8 Testfälle

bearbeitet von: Daniel Pupat

8.1 Komponententest

Wir haben hier alle Klassen aufgelistet, welche wir testen wollen. Dabei werden wir keine abstrakten Klassen und Interfaces testen. Exceptions testen wir nicht einzeln, sondern diese werden mit den zugehörigen Methoden getestet.

Klasse	Implementierer	Tester	Testart
Match	Patrick	Tim E.	Whitebox
Question	Tim W.	Daniel	Whitebox
User	Patrick	Olga	Whitebox
Client	Daniel	Tobias	Whitebox
ClientInbox	Patrick	Tim E.	Whitebox
Data	Tim W.	Olga	Blackbox
Server	Tim W.	Patrick	Whitebox
ServerDirectory	Tim E.	Tobias	Whitebox
ServerInbox	Tobias	Daniel	Whitebox
AuthRequest	Tim E.	Olga	Whitebox
LogoutRequest	Tobias	Patrick	Whitebox
AuthResponse	Daniel	Tim W.	Whitebox
UserListChangedResponse	Daniel	Tobias	Whitebox
NetUtils	Olga	Tobias	Whitebox
QuizGame	Daniel	Tim W.	Whitebox

Tabelle 1: Komponententests

8.2 Integrationstest

Testfallbezeichner	IT-1-a Frage hinzufügen
Testobjekte	Persistence, entities
Eingabe	Frage(Frage, Antwort)
Ausgabe	Erfolgreich Frage hinzugefügt
Umgebungserfordernisse	Server läuft, Datenbank existiert, Admin angemeldet
Anforderungen	keine
Abhängigkeiten	keine

Testfallbezeichner	IT-1-b Frage löschen
Testobjekte	Persistence, entities
Eingabe	Frage
Ausgabe	Erfolgreich Frage gelöscht
Umgebungserfordernisse	Server läuft, Datenbank existiert, Admin angemeldet
Anforderungen	Frage ist in Datenbank vorhanden
Abhängigkeiten	IT-1-a

Testfallbezeichner	IT-2 User löschen
Testobjekte	Persistence, entities
Eingabe	User
Ausgabe	Erfolgreich User gelöscht
Umgebungserfordernisse	Server läuft, Datenbank existiert, Admin angemeldet
Anforderungen	User ist in Datenbank vorhanden
Abhängigkeiten	keine

Testfallbezeichner	IT-3 User registrieren
Testobjekte	entities, Persistence, Communication
Eingabe	Registrierdaten(Nutzername, Passwort)
Ausgabe	Registrierung erfolgreich
Umgebungserfordernisse	App ist gestartet, Server läuft, Datenbank existiert
Anforderungen	korrekte Eingabedaten
Abhängigkeiten	keine

Testfallbezeichner	IT-4 Anmelden
Testobjekte	entities, Persistence, Communication
Eingabe	Anmeldedaten
Ausgabe	Anmeldung erfolgreich
Umgebungserfordernisse	App ist gestartet, Server läuft, Datenbank existiert
Anforderungen	User existiert in Datenbank
Abhängigkeiten	IT-3

8.3 Funktionstest

Bezeichner:	FT-1
Anwendungsfall	App starten
Eingabe	Start-Button wird gedrückt
Ausgabe	App ist gestartet Bei Fehlerfall: Rückmeldung Probleme beim starten der App
Umsetzung	Start-Button wird gedrückt, Benutzer hat die App bereits heruntergeladen

Bezeichner:	FT-2
Anwendungsfall	Offline-Modus starten
Eingabe	Offline Modus starten wird gedrückt
Ausgabe	Der Modus wurde gestartet, spielen nun möglich Bei Fehlerfall: Rückmeldung starten nicht erfolgreich
Umsetzung	Auf Offline Modus starten drücken

Bezeichner:	FT-3
Anwendungsfall	Benutzer registrieren
Eingabe	Informationen des Nutzers(Nickname, Passwort)
Ausgabe	Rückmeldung über erfolgreiches Anmelden. Nutzer ist nun in Datenbank gespeichert Bei Fehlerfall: Fehlerrückmeldung
Umsetzung	Manuelles eintragen der Daten, dann auf registrieren klicken

Bezeichner:	FT-4
Anwendungsfall	Benutzer anmelden
Eingabe	Anmeldedaten(Nutzername, Passwort)
Ausgabe	Rückmeldung über erfolgreiche Anmeldung Bei Fehlerfall: Rückmeldung Anmeldung nicht erfolgreich
Umsetzung	Manuelle Eingabe der Nutzerdaten, Benutzer ist registriert

Bezeichner:	FT-5
Anwendungsfall	Benutzer abmelden
Eingabe	Logout-Button wird gedrückt
Ausgabe	Rückmeldung über erfolgreiche Abmeldung Bei Fehlerfall: Rückmeldung Abmeldung nicht erfolgreich
Umsetzung	Manuelles ausloggen, Benutzer ist angemeldet

Bezeichner:	FT-6
Anwendungsfall	Spielen(Offline)
Eingabe	Spielen-Button drücken
Ausgabe	Spiel wird gestartet Bei Fehlerfall: Rückmeldung Spielen nicht möglich
Umsetzung	Manuell, Benutzer spielt Offline

Bezeichner:	FT-7
Anwendungsfall	Spielen(Online)
Eingabe	Spielen-Button wird gedrückt
Ausgabe	Auswahl zwischen Neues Spiel starten und offenes Spiel wird angezeigt Bei Fehlerfall: Rückmeldung Spielen nicht möglich
Umsetzung	Manuell, Benutzer ist angemeldet

Bezeichner:	FT-8
Anwendungsfall	Neues Spiel
Eingabe	Neues Spiel wird gedrückt
Ausgabe	Liste aller Spieler die Online sind erscheint Bei Fehlerfall: Rückmeldung Neues Spiel nicht erfolgreich
Umsetzung	Manuell, Benutzer war auf Spielen

Bezeichner:	FT-9
Anwendungsfall	Gegner herausfordern
Eingabe	Gegner der herausgefordert werden soll
Ausgabe	Spiel wird gestartet Bei Fehlerfall: Rückmeldung herausfordern nicht erfolgreich
Umsetzung	Manuell, Gegner ist Online

Bezeichner:	FT-10
Anwendungsfall	Offenes Spiel spielen
Eingabe	Antwort auf die Frage
Ausgabe	Nachricht ob richtige oder falsche Antwort Bei Fehlerfall: Rückmeldung Fehler
Umsetzung	Manuell, Spiel wurde bereits angefangen

Bezeichner:	FT-11
Anwendungsfall	Einstellungen ändern
Eingabe	Benutzername, Passwort
Ausgabe	Rückmeldung über erfolgreiches Ändern Bei Fehlerfall: Rückmeldung Ändern war nicht erfolgreich
Umsetzung	Manuell, Benutzer ist angemeldet

Bezeichner:	FT-12
Anwendungsfall	Website wird aufgerufen
Eingabe	URL
Ausgabe	Website wird angezeigt Bei Fehlerfall: Rückmeldung anzeigen war nicht erfolgreich
Umsetzung	Manuell, Internet Verbindung vorhanden

Bezeichner:	FT-13
Anwendungsfall	Admin anmelden
Eingabe	Nutzername, Passwort
Ausgabe	Rückmeldung, anmelden erfolgreich Bei Fehlerfall: Rückmeldung anmelden war nicht erfolgreich
Umsetzung	Manuell

Bezeichner:	FT-14
Anwendungsfall	Admin abmelden
Eingabe	Logout-Button drücken
Ausgabe	Rückmeldung abmelden erfolgreich Bei Fehlerfall: Rückmeldung abmelden nicht erfolgreich
Umsetzung	Manuell, Admin ist angemeldet

Bezeichner:	FT-15
Anwendungsfall	Frage hinzufügen
Eingabe	Frageinformationen(Frage, Antworten)
Ausgabe	Rückmeldung über erfolgreiches Hinzufügen Bei Fehlerfall: Rückmeldung Hinzufügen war nicht erfolgreich
Umsetzung	Manuell, Benutzer ist Admin

Bezeichner:	FT-16
Anwendungsfall	Frage bearbeiten
Eingabe	Frageinformationen(Frage, Antworten)
Ausgabe	Rückmeldung über erfolgreiches Ändern Bei Fehlerfall: Rückmeldung Ändern nicht erfolgreich
Umsetzung	Manuell, Benutzer ist Bibliothekar

Bezeichner:	FT-17
Anwendungsfall	Frage löschen
Eingabe	zu löschende Frage
Ausgabe	Rückmeldung über erfolgreiches Löschen Bei Fehlerfall: Rückmeldung Löschen war nicht erfolgreich
Umsetzung	Manuell, Benutzer ist Admin

Bezeichner:	FT-18
Anwendungsfall	Frageliste importieren
Eingabe	Frageliste
Ausgabe	Rückmeldung über erfolgreiches importieren Bei Fehlerfall: Rückmeldung Importieren war nicht erfolgreich
Umsetzung	Manuell, Benutzer ist Admin

Bezeichner:	FT-19
Anwendungsfall	Frageliste exportieren
Eingabe	Frageliste
Ausgabe	Rückmeldung über erfolgreiches exportieren Bei Fehlerfall: Rückmeldung Exportieren nicht erfolgreich
Umsetzung	Manuell, Benutzer ist Admin

Bezeichner:	FT-20
Anwendungsfall	User löschen
Eingabe	User
Ausgabe	Rückmeldung über erfolgreiches Löschen Bei Fehlerfall: Rückmeldung Löschen war nicht erfolgreich
Umsetzung	Manuell, User existiert

Bezeichner:	FT-21
Anwendungsfall	Passwort ändern
Eingabe	Passwort
Ausgabe	Rückmeldung über erfolgreiches Ändern Bei Fehlerfall: Rückmeldung Ändern war nicht erfolgreich
Umsetzung	Manuell, Admin ist angemeldet

8.4 Leistungstest

8.4.1 Härtetest

Bezeichner:	LT-1
Beschreibung	100 Nutzer starten gleichzeitig ein Spiel
Ziel:	Robustheit der Datenbank mit vielen Anfragen umzugehen wird getestet
Bei Erfolg:	Datenbank kann Anfragen bearbeiten; es gibt keine langen Wartezeiten
Fehler:	TimeOut, Absturz

8.4.2 Volumentest

Bezeichner:	LT-2
Beschreibung	Eine sehr große CSV-Datei wird importiert
Ziel:	Robustheit der Datenbank mit großen Datenmengen umzugehen wird getestet
Bei Erfolg:	Datenbank kann mit der Verarbeitung umgehen; es gibt keine Fehler
Fehler:	TimeOut, Absturz

8.4.3 Sicherheitstest

Bezeichner:	LT-3
Beschreibung	Ein Nutzer gibt ein falsches Passwort ein(App und Website)
Ziel:	Korrekte Authentifizierung wird getestet
Bei Erfolg:	Nutzer kann sich nicht einloggen; System gibt Fehlermeldung
Fehler:	Benutzer kann sich anmelden

8.4.4 Erholungstest

Bezeichner:	LT-4
Beschreibung	Ein Nutzer gibt mehrmals ein falsches Passwort ein
Ziel:	Erholt sich das System; Kann man sich danach problemlos mit dem richtigen Passwort einloggen
Bei Erfolg:	Bei richtiger Eingabe der Logindaten, ist man eingeloggt
Fehler:	Man kann sich nicht mehr einloggen, Absturz

Bezeichner:	LT-5
Beschreibung	Der Server geht offline und startet die Verbindung neu
Ziel:	Ist das System nach neuer Verbindung wieder funktionstüchtig
Bei Erfolg:	System produziert keine Fehler
Fehler:	Funktionen werden nicht mehr unterstützt, Datenverlust, TimeOut, Absturz

9 Testzeitplan

Komponententests: Woche ab 28.7.2014
Integrationstests: Woche ab 28.7.2014
Funktionstests: Woche ab 28.7.2014
Leistungstests: Woche ab 04.8.2014