Technische Universität Braunschweig

ISF

News Genie

Software-Entwicklungspraktikum (SEP)
Sommersemester 2014

Testspezifikation

Auftraggeber:
Technische Universität Braunschweig
Institut für Informationssysteme
Prof. Dr. Wolf-Tilo Balke
Mühlenpfordstraße 23
38106 Braunschweig

Betreuer: Silviu Homoceanu

Auftragnehmer:

Name	E-Mail-Adresse
Philipp Dittrich	p.dittrich@tu-braunschweig.de
Fabian Heymann	f.heyman@tu-braunschweig.de
Dennis Klose	dennis.klose@tu-braunschweig.de
Adrian Reiß	adrian.reiss@tu-braunschweig.de
Christopher Sontag	c.sontag@tu-braunschweig.de
Nora Widdecke	nora.widdecke@tu-braunschweig.de

Braunschweig, 19. Juli 2014

Versionsübersicht

Version	Datum	Autor(en)	Status	Kommentar
0.1	28.04.2014	P. Dittrich		Einleitung und Testplan, sowie Komponentendiagramm eingefügt
0.2	01.05.2014	F. Heymann		Kapitel 2.2, 2.3, 2.4, 2.5 hinzugefügt
0.3	02.05.2014	N. Widdecke		Kapitel 3 hinzugefügt
0.4	02.05.2014	C. Sontag		Kapitel 3.3.1 hinzugefügt
0.5	02.05.2014	P. Dittrich		Kapitel 3.2 hinzugefügt und 3.1 bearbeitet
0.6	03.05.2014	D. Klose		Kapitel 3.3.3 Test-Case <T201> bearbeitet
0.7	03.05.2014	A. Reiss		Kapitel 3.3.2 Test-Case <T200> bearbeitet
0.8	05.05.2014	Auftragnehmer		diverse Verbesserungen
1.0	05.05.2014	Auftragnehmer	vorläufig	Hinweise des Betreuers eingearbeitet
1.1	16.07.2014	C. Sontag		Unit-Tests hinzugefügt

Inhaltsverzeichnis

1 Einleitung **6**

2 Testplan **7**

2.1 Zu testende Komponenten 7
2.2 Zu testende Funktionen/Merkmale 8
2.3 Nicht zu testende Funktionen 10
2.4 Vorgehen 10
2.5 Testumgebung 11

3 Abnahmetest **12**

3.1 Zu testende Anforderungen 12
3.2 Testverfahren 13
3.2.1 Testskripte 14
3.3 Testfälle 14
3.3.1 Testfall $\langle T100 \rangle$ - Ein- und Ausgabe in natürlichen interaktiven „Gesprächen“ 14
3.3.2 Testfall $\langle T200 \rangle$ - Webinterface – User 16
3.3.3 Testfall $\langle T201 \rangle$ - Webinterface – Administrator 19

4 Integrationstest **21**

4.1 Zu testende Komponenten 21
4.2 Testverfahren 22
4.2.1 Testskripte 22
4.3 Testfälle 23
4.3.1 Testfall $\langle T500 \rangle$ - *QueryHandler* + *Natural Language Processing* + *Analyser* + *Searcher* + *Result Processing* 23
4.3.2 Testfall $\langle T600 \rangle$ - *Query Processor* + *Datenbank* 24
4.3.3 Testfall $\langle T700 \rangle$ - *Datenbank* + *Crawler* 25
4.3.4 Testfall $\langle T800 \rangle$ - *Datenbank* + *Linked Open Data* 26
4.3.5 Testfall $\langle T900 \rangle$ - *Managment Handler* + *Client* 27

5 Unit-Tests **28**

5.1 Zu testende Komponenten 28

5.2 Testverfahren 28
5.2.1 Testskripte 29
5.3 Testfälle 29
5.3.1 Testfall $\langle T1000 \rangle$ - Klasse LinkedOpenData 29
5.3.2 Testfall $\langle T1100 \rangle$ - Klasse Database 30
5.3.3 Testfall $\langle T1200 \rangle$ - Klasse Crawler 31
5.3.4 Testfall $\langle T1300 \rangle$ - Klasse Language 32
5.3.5 Testfall $\langle T1400 \rangle$ - Klasse Analyser und Natural Language Processing . . . 33
5.3.6 Testfall $\langle T1300 \rangle$ - Klasse ManagmentCache 34
5.3.7 Testfall $\langle T1400 \rangle$ - Klasse Client 35

Abbildungsverzeichnis

2.1 Komponentendiagramm, *Architektur von* News Genie 8

1 Einleitung

News Genie ist eine Software zur sprachgesteuerten und interaktiven Auswahl von Nachrichten aus durch den Benutzer vorausgewählten Quellen.

Mit natürlicher Sprache als Eingabemedium setzt *News Genie* auf eine neue Art der Interaktion mit dem Benutzer. Dieses Interface muss deshalb besonders stabil, einfach und intuitiv sein, dabei aber tolerant auf Fehlbedienung reagieren. Umfangreiche Tests sind für die Sicherung und Kontrolle dieser Ziele unerlässlich um Fehler frühzeitig zu erkennen, die Spezifikationen zu verifizieren und die Gesamtqualität der Software zu verbessern.

Die Software wird deshalb in allen Entwicklungsstadien umfangreichen Tests unterzogen, um die im Pflichtenheft als besonders wichtig eingestuften Qualitätsmerkmale wie Angemessenheit und Richtigkeit, Verständlichkeit, Bedienbarkeit und Erlernbarkeit, sowie Zeitverhalten und Analysierbarkeit zu gewährleisten.

Neben der Sprachsteuerung muss auch das Webinterface getestet werden. Es dient dem Benutzer als Eingabeschnittstelle für Nachrichtenquellen und andere Einstellungen. Hier gilt es neben der einfachen Bedienbarkeit auch die Datensicherheit als Qualitätsmerkmal.

2 Testplan

Im Folgenden wird der Testplan für *News Genie* erläutert. Einzelne Komponenten und ihre Funktionsweise, sowie die daraus resultierende Teststrategie werden vorgestellt.

News Genie setzt mit Sprachsteuerung auf ein Eingabemedium, für das dem Benutzer keine eigenen Strategien zur Fehlerbehandlung zur Verfügung stehen (am Bildschirm zum Beispiel das Schließen und erneute Öffnen des Programms), deshalb muss besonderes Augenmerk auf die Stabilität, Fehlertoleranz und einfache Bedienbarkeit dieser Schnittstelle gelegt werden.

2.1 Zu testende Komponenten

News Genie wird aus einem Client und verschiedenen Serverkomponenten bestehen. Wie in Abb. 2.1 dargestellt handelt es sich dabei um

Client Läuft auf einem Raspberry Pi Einplatinencomputer und wird die direkte Interaktion mit dem Benutzer übernehmen, sowie dessen Anfragen an den Server weiterleiten

Crawler Eine Serveranwendung, die im Hintergrund die vom Benutzer hinzugefügten RSS-Feeds durchsucht und indexiert. Bereitet die Daten für den QueryProcessor auf.

QueryProcessor Nimmt die Anfragen der Clients entgegen und generiert möglichst präzise Antworten auf die gestellten Fragen. Dabei greift er auf die vom Crawler indexierten Daten zu.

Webinterface Ist für die Registrierung und die Einstellungen der Nutzer zuständig. Außerdem bietet das Webinterface Funktionen zur textbasierten Anfrage des QueryProcessors.

Datenbank Verbindet die einzelnen Komponenten und muss deshalb ein robustes Schema bieten, auf das alle Anwendungen gleichzeitig zugreifen können.

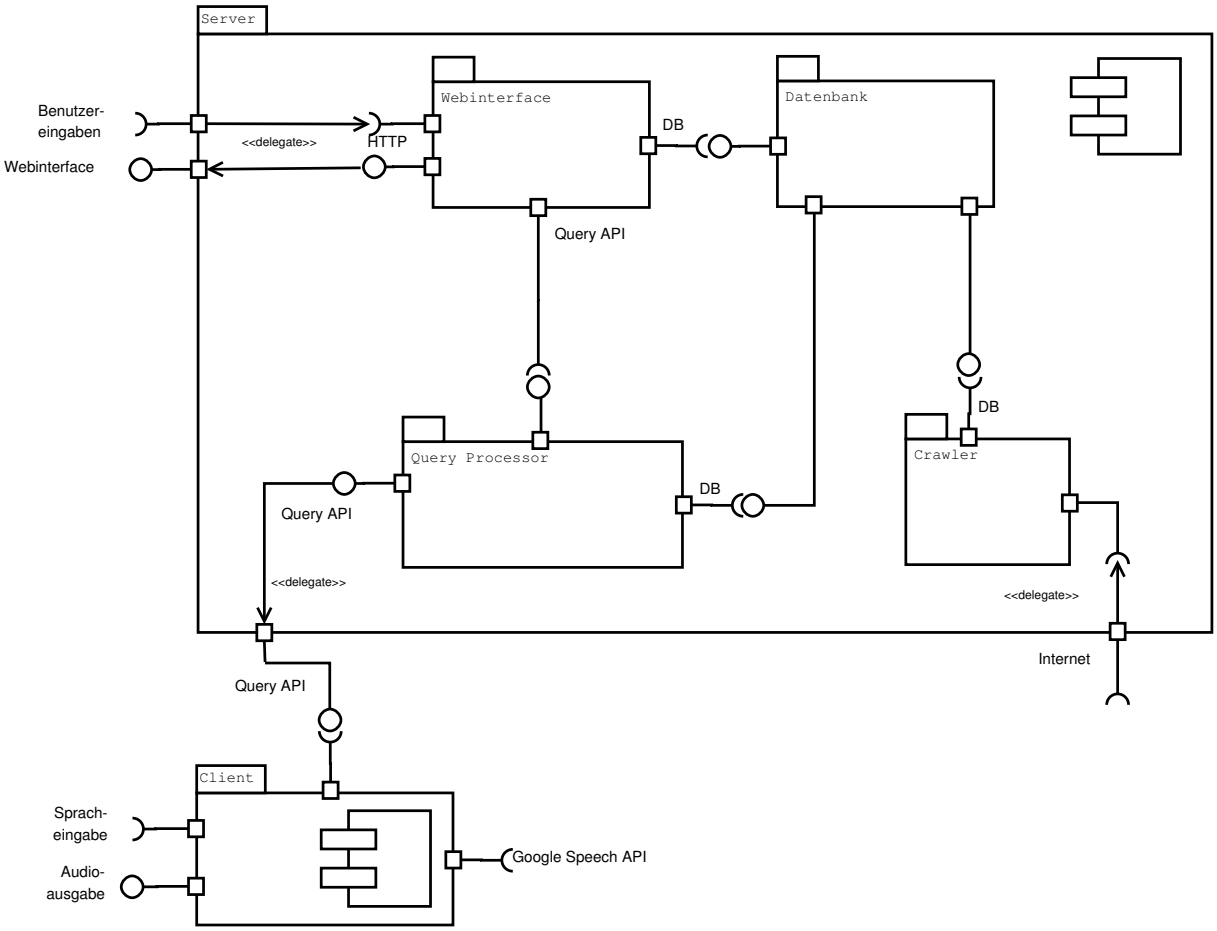

Abbildung 2.1: Komponentendiagramm, *Architektur von* News Genie

2.2 Zu testende Funktionen/Merkmale

- **??** Ein- und Ausgabe in natürlicher Sprache: Das System ist in der Lage verbal mit dem Benutzer zu kommunizieren. Um dies zu ermöglichen, kann es gesprochene Anfragen aufnehmen und verarbeiten sowie Antworten formulieren und leicht verständlich ausgeben.
- **??** Interaktives Verhalten: Das System grenzt Anfragen mit Ja-/Nein-Fragen ein, bricht die Sprachausgabe bei einem Stopp-Befehl ab und erlaubt negatives Feedback.
- **??** Benutzerschnittstelle: Das Webinterface bietet die Möglichkeit, Textanfragen zu stellen und die Resultate manuell zu validieren.
- **??** Backend: Nachrichtendaten werden in einem „Triplestore“ mit „Inverted Indexes“ gespeichert um effiziente Volltextsuche zu ermöglichen. Zu einer Anfrage werden ausschließlich dazu passende Artikel ausgegeben.
- **??** Ausgabe von natürlicher Sprache: Der Client kann natürliche Sprache ausgeben.

- **??** Aufnahme von natürlicher Sprache: Der Client kann natürliche Sprache aufnehmen.
- **??** Umwandlung von Sprache in Text: Das System kann Sprachdateien in Text umwandeln.
- **??** Umwandlung von Text in Sprache: Das System kann Text in Sprachdateien umwandeln.
- **??**, **??** Anfrage stellen: Wenn der Benutzer am Client eingeloggt ist, kann er Anfragen stellen und erhält eine sinnvolle Antwort.
- **??** Client Login: Ein Benutzer kann sich am Client einloggen.
- **??**, **??** Client Logout: Nach einem Login am Client kann sich ein Benutzer wieder ausloggen.
- **??** News-Crawling: Neue Artikel werden regelmäßig in das System aufgenommen.
- **??** Registrierung: Neue Benutzer können sich registrieren.
- **??** Webinterface Login: Benutzer können sich am Webinterface einloggen.
- **??**, **??** Webinterface Logout: Nach einem Login am Webinterface kann sich ein Benutzer wieder ausloggen.
- **??**, **??** Feed-Liste anzeigen: Eingeloggten Benutzern wird eine Liste ihrer abonnierten Feeds angezeigt.
- **??**, **??** Feed hinzufügen: Eingeloggte Benutzer können neue Feeds abonnieren.
- **??**, **??**, **??** Feed entfernen: Eingeloggte Benutzer können zuvor abonnierte Feeds wieder entfernen.
- **??**, **??** Passwort ändern: Eingeloggte Benutzer können ihr Passwort ändern.
- **??** Passwort Wiederherstellung: Benutzer können ihr Passwort wiederherstellen.
- **??**, **??** Rolle wechseln: Eingeloggte Administratoren können die Rolle eines anderen Benutzers annehmen.
- **??**, **??** Benutzer-Liste anzeigen: Eingeloggten Administratoren wird eine Liste aller Benutzer angezeigt.
- **??**, **??** Benutzer löschen: Eingeloggte Benutzer haben die Möglichkeit, ihren Account zu löschen.
- **??**, **??** Eingeloggte Benutzer können Textanfragen stellen und die Resultate manuell validieren.
- **??** Maximale Antwortzeit: Auf jede Benutzereingabe erfolgt innerhalb von maximal fünf Sekunden eine Antwort.
- **??** Anwenderfreundlichkeit: Das Produkt ist für Benutzer ohne EDV-Vorkenntnisse intuitiv bedienbar.

- **??** Robustheit: Fehlerhafte Eingaben bringen das Produkt nicht zum Absturz.
- **??** Relevanz: Benutzer erhalten nur für sie relevante Nachrichten.
- **??** Ressourcenverbrauch: Der Client funktioniert ohne Einschränkungen auf einem Raspberry Pi.

2.3 Nicht zu testende Funktionen

Die Anforderung **??** „Das System muss die englische Sprache unterstützen“ wird von uns nicht separat getestet, da sie sich auf alle Funktionen bezieht. Die Erfüllung der Anforderung wird dadurch beim Test der einzelnen Funktionen überprüft.
Außerdem setzen wir hinreichende Tests bei allen von Drittanbietern entwickelten Bibliotheken und Systemen voraus und testen diese nicht selbst.
Insbesondere sind dies:

- Apache openNLP
- Apache Lucene
- akka
- playFramework
- OpenLink Virtuoso
- Google-Speech-API
- Raspbian
- Debian

2.4 Vorgehen

Die Software des *News Genies* kann in vier Bereiche getrennt und im einzelnen getestet werden: Funktionalitäten des Clients, Funktionalitäten des Servers, Funktionalitäten des Webinterfaces und schließlich das Zusammenspiel des Client-Server-Systems. In folgenden Schritten wird beim Testen vorgegangen:

1. Komponententest
   Alle Komponenten von *News Genie* können mit JUnit-Tests überprüft werden. Während der Implementierung werden bereits Blackbox-Testfälle vom jeweiligen Autor erstellt und im Test-First-Verfahren begleitend zur Implementierung durchgeführt. Nach Abschluss

der Implementierung einer Klasse ist so bereits ihre (unabhängige) Funktionalität sichergestellt.

2. Integrationstest
   Wurde eine Komponente fertiggestellt und erfolgreich getestet, folgt nach dem Bottom-Up-Prinzip die Integration in *News Genie*. Dabei wird durch das Ausführen von *News Genie* der Integrationstest sowie die Funktionaliät geprüft. In welcher Reihenfolge die Komponenten unter Berücksichtigung ihrer Abhängigkeit integriert werden, wird zu einem späteren Zeitpunkt konkretisiert.

3. Systemtest
   Die Anforderungen des Systems aus dem Pflichtenheft werden geprüft, indem sie auf dem System mindestens einmal korrekt ausgeführt werden. Dabei wird ein Client und ein gestartetes Webinterface benötigt. Probleme oder Auffälligkeiten werden dokumentiert und gegebenenfalls behoben.

4. Abnahmetest
   Die Anforderung **??** „Backend“ und die damit verbundene Funktion **??** „News-Crawling“ können nicht vom Kunden direkt getestet werden. Um die Korrektheit der Implementierung zu zeigen, legen wir eine Log-Datei vor, die zeigt, wann welche Artikel in die Datenbank aufgenommen wurden.
   Alle anderen Funktionen kann der Kunde selbst vor Ort testen. Dazu wird er sich zunächst einen Account anlegen, sich mit diesem einloggen und alle Funktionen des Webinterfaces sowie des Clients nacheinander ausprobieren. Bei diesem Vorgang erhält der Kunde auch einen guten Eindruck der Qualitätsmerkmale **??** „Maximale Antwortzeit“ **??** „Anwenderfreundlichkeit“ **??** „Relevanz“ und **??** „Ressourcenverbrauch“. Zum Testen von Qualitätsmerkmal **??** „Robustheit“ ist eine größere Anzahl an Probedurchläufen nötig, die wir in Form einer Umfrage gestalten werden. Bei dieser werden sowohl **??** „Anwenderfreundlichkeit“ als auch **??** „Robustheit“ getestet.

2.5 Testumgebung

Sowie für die Entwicklung als auch den Abnahmetest und die weitere Verwendung wird die Serversoftware auf einem Server des IfIs ausgeführt und ein Raspberry Pi als Client verwendet. Zum Testen muss auf beiden Systemen eine JUnit-Testsuite installiert sein.

3 Abnahmetest

Der Abnahmetest für den *News Genie* dient dazu, die vom im Pflichtenheft festgelegten Anforderungen an das Endprodukt auf ihre vollständige Erfüllung zu prüfen. Er soll sowohl die durch den Auftraggeber vorgestellten technischen Hintergrundprozesse sowie die eigentliche Bedienung des Produkts bewerten. Dies bedeutet, dass dem Kunden das vollständige Produkt mit all seinen Funktionen vorgestellt wird und er dieses mit seinen Vorstellungen abgleicht. Ziel ist es, dass der Kunde mit dem Endprodukt vollständig zufrieden ist und keine weiteren Anmerkungen hat, sodass das Produkt freigegeben werden kann.

3.1 Zu testende Anforderungen

Der Kunde soll im Verlauf des Abnahmetests alle Funktionen, die der Nutzer später verwendet, ebenfalls ausführen. Dies bedeutet, dass er zunächst ein Benutzerkonto anlegt, um dann in beiden möglichen Arten Anfragen an den *News Genie* zu stellen. Die Reihenfolge soll dabei sein:

1. Registrierung **??**
2. Passwort wiederherstellen **??**
3. Webinterface Login **??**
4. Passwort ändern **??**
5. Feed-Liste anzeigen **??**
6. Feed hinzufügen **??**
7. Feed entfernen **??**
8. Administration: Benutzerliste anzeigen **??**
9. Administration: In Rolle eines Benutzers wechseln **??**
10. Administration: Benutzer löschen **??**
11. Text-Anfrage **??**
12. Webinterface Logout **??**
13. Client Login **??**, umfasst **??** und **??**

14. Anfrage stellen **??** (Sprachinterface)
15. Client Logout **??**

Mit diesem Ablauf werden auch die Kriterien **??**, **??**, **?? ??**, **??**, **??**, **??** abgedeckt.

3.2 Testverfahren

Der Abnahmetest wird in einem Plenum mit allen Stakeholdern erfolgen. Von den Auftragnehmern wird zu Beginn der komplette Funktionsumfang von *News Genie* präsentiert. Dabei werden dem Kunden auch die Backend-Komponenten (Crawler, QueryProcessor und Datenbank) im Detail vorgestellt und Wartungsaufgaben besprochen.

Für den praktischen Teil des Abnahmetests bekommt der Kunde Gelegenheit weitere Mitarbeiter oder externe Teilnehmer einzubeziehen. Diese sollen nun den kompletten Lebenszyklus eines Benutzers aus Benutzer und Administrationssicht im System nachzuvollziehen und zu testen. Dies erfolgt in zwei Teilen: An einem Arbeitsplatz mit Bildschirm und Tastatur wird das Webinterface getestet, anschließend bekommt der Kunde die Gelegenheit an einem bereitstehenden Client das Sprachinterface zu testen.

Den Teilnehmern wird ein Testskript zur Verfügung gestellt, in dem die geplanten Schritte kurz erläutert werden. Zunächst registriert der Kunde sich im Webinterface von *News Genie* **??**. Bevor er sich einloggt, soll die Wiederherstellung eines vergessenen Passwortes von ihm getestet werden **??**. Nach erfolgreicher Registrierung meldet sich der Kunde mit dem neu erstellten Benutzer an der Weboberfläche an **??**. Nun soll er versuchen, sein Passwort zu ändern **??**.

Im nächsten Schritt testet der Kunde die Funktionen zur Verwaltung seiner persönlichen Nachrichtenquellen. Dazu lässt er sich seine RSS-Feed-Liste anzeigen **??**, fügt anschließend einen RSS-Feed hinzu **??** und testet abschließend das Löschen von RSS-Feeds **??**.

Zum Testen der Administrations-Funktionen von *News Genie* wird dem Kunden nun ein Benutzer mit entsprechenden Rechten zur Verfügung gestellt. Nun testet der Kunde die Benutzerliste **??**, wechselt in die Rolle eines normalen Benutzers **??** und kehrt in die Administrationsübersicht zurück, um einen Benutzer zu löschen **??**.

Als letzter Test im Webinterface soll der Kunde nun eine Text-Anfrage an *News Genie* stellen **??** und die Anzeige der Ergebnisse im Webinterface in Augenschein nehmen. Anschließend loggt er sich aus **??**.

Nun beginnt der Test des Clients. Der Kunde drückt zuerst den Sprach-Button um sich per Sprachbefehl am Client einzuloggen **??**. Anschließend stellt er verschiedene Anfragen an *News Genie* **??**. Hierfür wird ein Skript der Möglichkeiten mit Testvorschlägen erarbeitet, so dass der

Kunde Gelegenheit hat, alle Features kennenzulernen und zu testen. Abschließend meldet der Kunde sich, wiederum per Sprachbefehl, am Client ab **??**.

3.2.1 Testskripte

Zur Durchführung des Abnahmetests werden dem Kunden ein erläuterter Ablaufplan wie in 3.1 und Vorschäge zu Sprachanfragen übergeben. Weitere Testskripte sind nicht nötig.

3.3 Testfälle

3.3.1 Testfall $\langle T100 \rangle$ - Ein- und Ausgabe in natürlichen interaktiven „Gesprächen“

Ziel

Zweck des Tests ist es, das Verständnis des *News Genies* bei sprachlichen Eingaben sowie seiner sprachlichen Ausgaben zu überprüfen. Auch wird hierbei das interaktive Verhalten getestet. Gleichzeitig wird es ermöglicht, das Login und Logout-System zu testen.

Objekte/Methoden/Funktionen

??, **??**, **??**, **??**, **??**, **??**, **??**, **??**, **??**, **??**, **??**, **??**, **??**

Pass/Fail Kriterien

Bedienung des Systems am Client durch eine nicht involvierte Testperson. Test erfolgreich, wenn sich die Testperson erfolgreich anmelden kann, die Testperson die Ausgabe, Benutzerfreundlichkeit sowie Reaktionszeit als gut bewertet und sich anschließend wieder abmelden kann. Die Benutzerfreundlichkeit wird dabei an der Anzahl der nötigen Interaktionen bis zur Antwort gemessen werden. Dagegen wird die Qualität der Antwort an der Qualität der Erkennung und an dem Übereinstimmungsgrad der Antwort mit der Frage bestimmt. Auch muss die Anfrage in einem fünf Sekunden Zeitraum geschehen.

Vorbedingung

Die Datenbank muss mindestens 100 Einträg besitzen.

Einzelschritte

Eingabe:

1. Anmeldung der Testperson am Client
2. Die Testperson stellt *News Genie* eine verständliche Frage oder unverständliche Frage
3. Die Testperson wartet bis *News Genie* eine Antwort liefert
4. Die Antwort wird von der Testperson auf Korrektheit überprüft.

5. Soll noch eine Frage gestellt werden, so wendet die Testperson nochmals alle Schritte ab Schritt 2 an.

6. Abmelden am Client

Ausgabe: Als Ausgabe erhalten wir eine positive oder negative Rückmeldung der Testperson sowie einen Bogen mit den gestellten Fragen und den von *News Genie* genannten Antworten.

Beobachtungen / Log / Umgebung

Wir beobachten weiterhin die Geschwindigkeit der Antwortsuche durch die Messung der Zeit zwischen Fragestellung und Antwort und notieren die gestellten Fragen und *News Genies* Antworten.

Besonderheiten

Es muss bei der Ausführung beachtet werden, dass jede Testperson eine andere Stimme hat und dadurch Abweichungen entstehen können.

Abhängigkeiten

Es gibt keine besonderen Abhängigkeiten.

3.3.2 Testfall $\langle T200 \rangle$ - Webinterface – User

Ziel

Ziel ist es, das für den User bereitgestellte Webinterface auf seine Funktionalität zu prüfen. Neben der Registration, dem Login und dem Logout, soll es dem User möglich sein Feeds hinzuzufügen und zu löschen. Darüber hinaus werden die Funktionen Passwort ändern, Passwort wiederherstellen, sowie das Löschen des Nutzeraccounts getestet. Auch Textanfragen, welche über das Webinterface an den *News Genie* gestellt werden können, werden auf ihre Funktionalität und Qualität getestet.

Objekte/Methoden/Funktionen

??, ??, ??, ??, ??, ??, ??, ??, ??, ??, ??, ??

Da die Qualitätskriterien ??, ??, ?? bereits im ersten Testfall abgedeckt sind und die Technik der Text-/ und der Sprachanfrage die selbe ist, werden sie in diesem Testfall nicht berücksichtigt.

Pass/Fail Kriterien

Eine nicht involvierte Testperson Registriert sich am Webinterface und Loggt sich ein. Anschliessend testet sie die verfügbaren Funktionen und bewertet die Funktionalität und Benutzerfreundlichkeit.

Vorbedingung Die Testperson muss sich Registrieren und einloggen.

Einzelschritte

Eingabe:

1. Registrierung der Testperson am Webinterface.
   a) Eingabe des Namens der Testperson.
   b) Eingabe der Email-Adresse.
   c) Eingabe des Passwortes.
2. Login der Testperson am Webinterface.
   a) Eingabe des Namens der Testperson.
   b) Eingabe des Passwortes.
3. Logout der Testperson am Webinterface.
4. Passwort wiederherstellen.
   a) Betätigen des „Passwort vergessen“ Buttons.
   b) Eingabe des Namens, sowie der Email-Adresse.
   c) Betätigen des „Neues Passwort generieren“ Buttons.

5. Erneut 2, mit neuem generierten Passwort.
6. Das Passwort ändern.
    a) Betätigen des „Passwort ändern“ Buttons.
    b) Eingabe des alten Passwortes, des neuen Passwortes, sowie Bestätigung des neuen Passwortes.
    c) Betätigen des „Passwort ändern“ Buttons.
7. Feeds hinzufügen.
    a) Die URL des Feeds in das Feld eingeben.
    b) Betätigen des „Feed hinzufügen“ Buttons.
    c) Betrachten der, um den hinzugefügten Feed ergänzte, Feed-Liste. Evtl. erneut ab 7a.
8. Feeds löschen.
    a) Haken in der Liste setzten, bei zu entfernender Feeds.
    b) Betätigen des „Feeds entfernen“ Buttons.
    c) Bestätigen.
    d) Betrachten der um die gelöschten Feeds reduzierte Feed-Liste. Evtl. erneut ab 8a.
9. Stellen einer Textanfrage an den *News Genie*.
    a) Anfrage in das Textfeld eingeben.
    b) Betätigen des „Anfrage stellen“ Buttons.
    c) Manuelles Auswählen einer der vom *News Genie* bereit gestellten Nachrichten.
    d) Evtl. erneut ab 9a.
10. Löschen des Accounts.
    a) Betätigen des „Account löschen“ Buttons.
    b) Bestätigen.

Ausgabe: Als Ausgabe erhalten wir eine Rückmeldung der Testperson bezüglich der Benutzerfreundlichkeit und Verständlichkeit des Webinterfaces, sowie eine Liste mit allen evtl. aufgetretenen unerwarteten Reaktionen oder Fehlern.

Beobachtungen / Log / Umgebung

Die Testperson wird bei der Bedienung beobachtet um so die Bedienbarkeit des Webinterfaces beurteilen zu können. Dabei gilt es darauf zu achten, wie Intuitiv sich die Testperson durch das Webinterface klickt, und wieviele Fragen sie bezüglich der Bedienung hat.

Besonderheiten

Es gibt keine Besonderheiten.

Abhängigkeiten

Es gibt keine besonderen Abhängigkeiten.

3.3.3 Testfall $\langle T201 \rangle$ - Webinterface – Administrator

Ziel

Es soll geprüft werden, ob ein Administrator eine Liste aller Benutzer angezeigt bekommt und dort Nutzer löschen, sowie auch deren Rolle übernehmen, kann.

Objekte/Methoden/Funktionen

??,??

Pass/Fail Kriterien

Die dem Administrator angezeigte Liste aller Benutzer muss vollständig sein. Nach dem löschen des Benutzers darf dieser nicht mehr aufgelistet sein. Am Ende muss die Rolle zu demjenigen Benutzer gewechselt worden sein, dessen Rolle übernommen werden sollte.

Vorbedingung

Der Administrator muss am Webinterface eingeloggt sein. Es müssen bereits mindestens zwei weitere Accounts registriert worden sein. Einer davon sollte extra für diesen Test, zum löschen, registriert worden sein.

Einzelschritte

Eingabe:

1. Benutzer-Liste Anzeigen.
   a) Den Knopf „Administration“ drücken.

   Erwartete Beobachtung: Eine vollständige Liste aller Benutzer wird angezeigt.

2. Benutzer Löschen.
   a) Das Auswahlkästchen neben dem zu löschenden Nutzer in der Liste anklicken.
   b) Den Knopf „Delete selected“ drücken.

   Erwartete Beobachtung: Der zuvor ausgewählte Nutzer wird nicht mehr aufgelistet.

3. Rolle wechseln.
   a) Bei einem Benutzer in der liste den Knopf „sudo“ drücken.

   Erwartete Beobachtung: Oben rechts, neben „username:“ wird jetzt der Name des Nutzers angezeigt, dessen Rolle übernommen werden sollte.

Ausgabe: Eine Rückmeldung des Administrators, ob nach jedem Schritt, die erwarteten Beobachtungen eingetreten sind.

Beobachtungen / Log / Umgebung

Der Administrator bekommt entweder direkten Zugriff auf die Datenbank, oder eine manuelle Liste aller real bereits registrierten Benutzer, um die Funktion **??** auf Vollständigkeit überprüfen zu können.

Besonderheiten

Es gibt keine Besonderheiten.

Abhängigkeiten

Der Testfall 3.3.3 ist Abhängig vom Testfall 3.3.2. Das Registrieren, sowie das Einloggen muss bereits funktionieren (Siehe Vorbedingung).

4 Integrationstest

Die Integrationstests überprüfen die reibungslose Zusammenarbeit der einzelnen Komponenten des *NewsGenies*.

4.1 Zu testende Komponenten

Um die *Query Processor*-Komponente auf die Korrektheit ihrer Integration zu testen, müssen folgende Unterkomponenten getestet werden:

- *Handler*
- *Natural Language Processing*
- *Analyser*
- *Searcher*
- *Result Processing*

Insgesamt sind im Paket *Server* folgende Komponenten im Zusammenspiel zu testen:

- *Linked Open Data*
- *Datenbank*
- *Query Processor*
- *Webinterface*
- *Crawler*
- *Linked Open Data*
- *Cache*
- *Language*

Letztendlich ist die Kommunikation des *Clients* mit dem *Server* sicherzustellen:

- *Client*

4.2 Testverfahren

Bei den Integrationstests wurde nach dem Bottom-Up Verfahren getestet. Dabei wurden zuerst die Komponenten des *Query Processors* getestet und dieser anschliessend als eigene Komponente betrachtet. Danach werden wird die Integration der Komponenten des *Servers* getestet und dieser ebenfalls anschliessend als eigene Komponente betrachtet. Abschliessend bleibt das Zusammenspiel von dem *Server* mit dem *Client* zu testen. Die Komponenten werden mit Ausnahme der Unterkomponenten des *Query Processors* jeweils paarweise getestet. Diese werden auf Grund der starken Verzahnung und der parallelen Integration im Entwicklungsprozess in einem Testfall behandelt.

4.2.1 Testskripte

Es wurden keine Testskripte verwendet.

4.3 Testfälle

4.3.1 Testfall $\langle T500 \rangle$ - *QueryHandler + Natural Language Processing + Analyser + Searcher + Result Processing*

Ziel Ziel ist die Sicherstellung der Zusammenarbeit der Komponenten des *Query Processors*.

Objekte/Methoden/Funktionen **Objekte:**

- *ClientQuery*
- *AnalysedQuery*
- *SearchRequest*
- *SearchAnswer*
- *ClientAnswer*

Methoden:

- *analyse()*
- *search()*
- *makeClientAnswer()*

Pass/Fail Kriterien Pass: Die Text des *ClientQuerys* durchläuft die Komponenten und gibt eine ClientAnswer zurück.
Fail: Es gibt eine Java Fehlermeldung oder der Text wurde nicht analysiert.

Vorbedingung

- Der *Query Handler* benötigt eine *ClientQuery*, auf welche reagiert wird.
- Es muss die *Datenbank* mit den entsprechenden Daten vorhanden sein.

Einzelschritte

- *QueryHandler* starten.
- Eine *ClientQuery* an den *QueryHandler* senden.
- Die *ClientAnswer* auf Korrektheit prüfen.

Beobachtungen / Log / Umgebung Es muss beobachtet werden ob die *ClientQuery* tatsächlich alle Komponenten korrekt durchläuft. Dabei wird aus der *ClientQuery* im *Analyser* im Zusammenspiel mit *Natural Language Processing* eine *AnalysedQuery*. Anschliessend erstellt der *Searcher* eine *SearchRequest* an die *Datenbank*und erhält eine *SearchAnswer*. Letztendlich wird im *Result Processing* eine *ClientAnswer* generiert. Zur Überprüfung wird nach dem korrekten Durchlauf die *ClientAnswer* überprüft.

Besonderheiten Die *ClientAnswer* kann leer sein, wenn die Datenbank nicht das passende Suchergebnis beinhaltete.

Abhängigkeiten -

4.3.2 Testfall $\langle T600 \rangle$ - *Query Processor + Datenbank*

Ziel Ziel ist die Sicherstellung der Zusammenarbeit des *Searchers* in der *Query Processor*-Komponente und der *Datenbank*.

Objekte/Methoden/Funktionen **Objekte:**

- *ClientQuery*
- *ClientAnswer*

Methoden:

- *search()*
- *onReceive()*

Pass/Fail Kriterien Pass: Die Text des *ClientQuerys* durchläuft die Komponenten des *Query Handlers* und gibt anschließend eine passende *ClientAnswer* aus der Datenbank aus.
Fail: Es gibt eine Java Fehlermeldung oder die Datenbank reagiert nicht.

Vorbedingung

- Der *Query Handler* benötigt eine *ClientQuery*, auf welche reagiert wird.
- Die *Datenbank* muss die entsprechenden Daten enthalten.

Einzelschritte

- *QueryHandler* starten.
- Eine *ClientQuery* an den *QueryHandler* senden.
- Die *ClientAnswer* auf Korrektheit prüfen.

Beobachtungen / Log / Umgebung Es muss beobachtet werden, ob die *ClientAnswer* tatsächlich die Daten aus der *Datenbank* enthält.

Besonderheiten

- Die *ClientAnswer* kann leer sein, wenn die Datenbank nicht das passende Suchergebnis beinhaltete.

Abhängigkeiten - *Query Processor*

4.3.3 Testfall $\langle T700 \rangle$ - *Datenbank + Crawler*

Ziel Ziel ist die Sicherstellung der Zusammenarbeit der *Datenbank* und des *Crawlers*.

Objekte/Methoden/Funktionen **Objekte:**

- *Articles*

Methoden:

- *startCrawler()*
- *crawl()*

Pass/Fail Kriterien Pass: Der *Crawler* arbeitet korrekt.
Fail: Es gibt eine Java Fehlermeldung oder Feeds werden nicht hinzugefügt.

Vorbedingung
- Die *Datenbank* muss Feeds enthalten.

Einzelschritte
- Starten des *Crawlers*.
- *Crawler* nach 24 Stunden stoppen.
- Artikel überprüfen.

Beobachtungen / Log / Umgebung Es muss beobachtet werden, ob eine Anzahl von Artikeln hinzugefügt wurden.

Besonderheiten -

Abhängigkeiten -

4.3.4 Testfall $\langle T800 \rangle$ - *Datenbank + Linked Open Data*

Ziel Ziel ist die Sicherstellung der Zusammenarbeit der *Datenbank* und *Linked Open Data*.

Objekte/Methoden/Funktionen **Objekte:**

- *FactAnswer*
- *FactRequest*

Methoden:

- *searchFor()*

Pass/Fail Kriterien Pass: *Linked Open Data* liefert das korrekte Ergebnis.
Fail: Es gibt eine Java Fehlermeldung oder einen Timeout.

Vorbedingung -

Einzelschritte

- Senden einer *FactRequest* an die *Datenbank*.
- *FactAnswer* überprüfen.

Beobachtungen / Log / Umgebung -

Besonderheiten -

Abhängigkeiten -

4.3.5 Testfall $\langle T900 \rangle$ - *Managment Handler + Client*

Ziel Ziel ist die Sicherstellung der Zusammenarbeit des *Managment Handlers* und des *Clients*.

Objekte/Methoden/Funktionen **Objekte:**

- *ClientQuery*
- *ClientAnswer*

Methoden:

- *apply()*
- *onReceive()*

Pass/Fail Kriterien Pass: Die *ClientAnswer* wird ausgegeben.

Fail: Es gibt eine Java Fehlermeldung, oder eine der folgenden Fehlermeldungen:

- *"Remote actor not available [...]"*
- *"I'm sorry, that did not work."*
- *"Not ready yet"*

Vorbedingung Die *Server* Architektur muss korrekt arbeiten.

Einzelschritte

- Senden einer *ClientQuery* an den *Managment Handler*.
- *ClientAnswer* überprüfen.

Beobachtungen / Log / Umgebung -

Besonderheiten -

Abhängigkeiten -

5 Unit-Tests

Das Ziel der Unit-Tests ist es, jede Komponente auf ihre korrekte Funktionsweise zu testen. Hierbei werden besonders die Teile des Systems, die für den Nutzer bzw. für die Verarbeitung seiner Anfagen relevant sind, ausgiebig getestet. In den Tests werden mögliche Anfragen und Methodenparameter an die entsprechenden Methoden übergeben und die Ausgabe mit einem erwarteten Ergebnis verglichen.

Nicht getestet werden vorallem die Semantik und Funktionen, welche aus fertigen Bibliotheken als fertige .jar in das Projekt importiert werden. Aber auch das Webinterface wird nicht gesondert getestet, da Typesafe Activator nach jeder Veränderung die Website nochmals erstellt und überprüft wird, sowie der Searcher und Result-Processor, da diese keine Verarbeitung sondern nur Verteilung und Aufbereitung von Daten übernehmen.

5.1 Zu testende Komponenten

Im Paket *Server* sind folgende Komponenten zu testen:

- *Linked Open Data*
- *Datenbank*
- *Query Processor*
- *Crawler*
- *Language*
- *Cache*

Weiterhin ist das Verständnis für Eingaben und die Ausgabe des *Servers* sicherzustellen.

5.2 Testverfahren

Alle zu testenden Komponenten und Methoden werden mit JUnit-Tests getestet. Diese werden automatisch bei jeden Build durchgeführt und brechen bei Nichterfüllung den Build-Vorgang ab.

5.2.1 Testskripte

Folgende Tests sind für den Server auszuführen:

- AnalyserTest.java
- DatabaseTest.java
- CrawlerTest.java
- LanguageTest.java
- LinkedOpenData.java
- ManagmentCacheTest.java

Für den Server sollen folgende Tests durchgeführt werden:

- BasicAnalyzedRequestTest.java

5.3 Testfälle

5.3.1 Testfall $\langle T1000 \rangle$ - Klasse LinkedOpenData

Ziel Das Ziel des Tests ist es, die Erstellung einer korrekten Suchanfrage mit den Daten des Analyser zu überprüfen.

Objekte/Methoden/Funktionen Mit diesem Test wird die Methode searchfor getestet. Dafür wird eine neue *LinkedOpenData*-Instanz, wenn nicht schon vorhanden angelegt.

Pass/Fail Kriterien Der Test war erfolgreich, wenn der JUnit-Test ohne Failure durchgelaufen ist.

Vorbedingung Es besteht eine Internetverbindung

Einzelschritte Starten des JUnit-Testes und Überprüfung der Ausgabe

Beobachtungen / Log / Umgebung Es wird eine Entwicklungsumgebung für Java benötigt.

Besonderheiten -

Abhängigkeiten -

5.3.2 Testfall $\langle T1100 \rangle$ - Klasse Database

Ziel Das Ziel des Tests ist es, die Datenbank auf ihre Funktionalität zu prüfen.

Objekte/Methoden/Funktionen Mit diesem Test werden die wesentlichen Funktionen der Datenbank überprüft sowie die Antwortzeit. Dafür wird eine neue *Database*-Instanz mittels Akka-Actors erstellt.

Pass/Fail Kriterien Der Test war erfolgreich, wenn der JUnit-Test ohne Failure durchgelaufen ist.

Vorbedingung Es besteht eine Verbindung zur Datenbank.

Einzelschritte Starten des JUnit-Testes und Überprüfung der Ausgabe

Beobachtungen / Log / Umgebung Es wird eine Entwicklungsumgebung für Java benötigt.

Besonderheiten Der Test erstellt Daten in der Datenbank, löscht diese allerdings als einen weiteren Testpunkt wieder.

Abhängigkeiten Der Test ist abhängig von der Datenbank

5.3.3 Testfall $\langle T1200 \rangle$ - Klasse Crawler

Ziel Das Ziel des Tests ist es, den Crawler auf seine Crawl-Eigenschaft zu überprüfen.

Objekte/Methoden/Funktionen Mit diesem Test wird die wesentliche Funktion des Cawlers überprüft. Dafür wird eine neue *Crawler*-Instanz, ein englischer und ein deutscher Feed erstellt .

Pass/Fail Kriterien Der Test war erfolgreich, wenn der JUnit-Test ohne Failure durchgelaufen ist.

Vorbedingung Es besteht eine Internetverbindung.

Einzelschritte Starten des JUnit-Testes und Überprüfung der Ausgabe

Beobachtungen / Log / Umgebung Es wird eine Entwicklungsumgebung für Java benötigt.

Besonderheiten Der Test überprüft nicht die Semantik der gecrawlten Artikel, sondern nur, ob der Crawler Artikel empfängt.

Abhängigkeiten -

5.3.4 Testfall $\langle T1300 \rangle$ - Klasse Language

Ziel Das Ziel des Tests ist es, die Sprachabfrage aus der Sprachdatei auf ihre Funktionalität zu prüfen.

Objekte/Methoden/Funktionen Mit diesem Test werden die wesentlichen Funktionen für die Sprachdateiauslese überprüft. Dafür werden alle Sprachen durch die *Language*-Klasse geladen.

Pass/Fail Kriterien Der Test war erfolgreich, wenn der JUnit-Test ohne Failure durchgelaufen ist.

Vorbedingung -

Einzelschritte Starten des JUnit-Testes und Überprüfung der Ausgabe

Beobachtungen / Log / Umgebung Es wird eine Entwicklungsumgebung für Java benötigt.

Besonderheiten -

Abhängigkeiten Der Test ist abhängig von den Sprachdateien.

5.3.5 Testfall $\langle T1400 \rangle$ - Klasse Analyser und Natural Language Processing

Ziel Das Ziel des Tests ist es, die Funktion der Erkennung der Eingaben des Benutzers auf ihre Funktionalität zu prüfen.

Objekte/Methoden/Funktionen Mit diesem Test werden die Funktionen des Analysers sowie des Natural Language Processing überprüft. Dafür werden Instanzen von *Standfort-NLP* und des *Analysers* neu erstellt, wenn diese nicht bereits vorhanden sind.

Pass/Fail Kriterien Der Test war erfolgreich, wenn der JUnit-Test ohne Failure durchgelaufen ist.

Vorbedingung -

Einzelschritte Starten des JUnit-Testes und Überprüfung der Ausgabe

Beobachtungen / Log / Umgebung Es wird eine Entwicklungsumgebung für Java benötigt.

Besonderheiten -

Abhängigkeiten -

5.3.6 Testfall $\langle T1300 \rangle$ - Klasse ManagmentCache

Ziel Das Ziel des Tests ist es, den Cache auf seine Funktion zu überprüfen.

Objekte/Methoden/Funktionen Mit diesem Test werden die wesentlichen Funktionen des Caches überprüft. Dafür werden verschiedene Akka-Messages erstellt und in den Cache geladen, damit die Funktionen des Caches benutzt werden können.

Pass/Fail Kriterien Der Test war erfolgreich, wenn der JUnit-Test ohne Failure durchgelaufen ist.

Vorbedingung -

Einzelschritte Starten des JUnit-Testes und Überprüfung der Ausgabe

Beobachtungen / Log / Umgebung Es wird eine Entwicklungsumgebung für Java benötigt.

Besonderheiten -

Abhängigkeiten -

5.3.7 Testfall $\langle T1400 \rangle$ - Klasse Client

Ziel Das Ziel des Tests ist es, die Übereinstimmung eines gegebenen Srings mit einer Anzahl von Strings zu vergleichen

Objekte/Methoden/Funktionen Mit diesem Test wird die wesentliche Funktion für den Vergleich von mehreren Strings überprüft. Dafür wird eine Instanz der Klasse *BasicAnalyzedRequest* erstellt und genutzt.

Pass/Fail Kriterien Der Test war erfolgreich, wenn der JUnit-Test ohne Failure durchgelaufen ist.

Vorbedingung -

Einzelschritte Starten des JUnit-Testes und Überprüfung der Ausgabe

Beobachtungen / Log / Umgebung Es wird eine Entwicklungsumgebung für Java benötigt.

Besonderheiten -

Abhängigkeiten -