



X-MAP

ANDROID-APP ZUR QUALITATIVEN ERFASSUNG VON MOBILFUNKNETZEN

Software-Entwicklungspraktikum (SEP)
Sommersemester 2013

Testprotokolle

Auftraggeber
Technische Universität Braunschweig
Institut für Nachrichtentechnik
Prof. Dr.-Ing. Thomas Kürner
Schleinitzstraße 22
38106 Braunschweig

Betreuer: Dennis M. Rose

Auftragnehmer:

Name	E-Mail-Adresse
Sofia Ananieva	s.ananieva@tu-braunschweig.de
Andreas Bauerfeld	a.bauerfeld@tu-braunschweig.de
Ferhat Çinar	f.cinar@tu-braunschweig.de
Andreas Hecker	a.hecker@tu-braunschweig.de
Julia Kreyßig	julia@kreyssig.com
Timo Schwarz	t.schwarz@tu-braunschweig.de
Julian Troegel	j.troegel@tu-braunschweig.de
Deniz Yurtseven	d.yurtseven@tu-braunschweig.de

Braunschweig, 10. Juli 2013

Inhaltsverzeichnis

1 Einfügen in App-Datenbank - Testlauf 1 (2013-06-29)	5
1.1 Testumgebung	5
1.2 Testprotokoll	5
1.3 Zusammenfassung	6
2 Einfügen in App-Datenbank - Testlauf 2 (2013-06-29)	7
2.1 Testumgebung	7
2.2 Testprotokoll	7
2.3 Zusammenfassung	8
3 Einfügen in App-Datenbank - Testlauf 3 (2013-06-29)	9
3.1 Testumgebung	9
3.2 Testprotokoll	9
3.3 Zusammenfassung	10
4 Einfügen in App-Datenbank - Testlauf 4 (2013-06-29)	11
4.1 Testumgebung	11
4.2 Testprotokoll	11
4.3 Zusammenfassung	12
5 Funktionstest Webservice über SoapUI - Testlauf 1 (2013-06-29)	13
5.1 Testumgebung	13
5.2 Testprotokoll	13
5.3 Zusammenfassung	16
6 Daten sammeln und Ergebnisse graphisch darstellen - Testlauf 1 (2013-07-04)	17
6.1 Testumgebung	17
6.2 Testprotokoll	17
6.3 Zusammenfassung	18
7 Applikation an 6 unabhängige Personen verteilt - Testlauf 1 (2013 - 07 - 06)	21
7.1 Testumgebung	21
7.2 Testprotokoll	21
7.3 Zusammenfassung	22

8	Daten messen - Testlauf 1 (2013-07-09)	23
8.1	Testumgebung	23
8.2	Testprotokoll	23
8.3	Zusammenfassung	24
9	Löschen von Datensätzen - Testlauf 1 (2013-07-08)	26
9.1	Testumgebung	26
9.2	Testprotokoll	26
9.3	Zusammenfassung	27
10	Löschen von Datensätzen - Testlauf 2 (2013-07-09)	28
10.1	Testumgebung	28
10.2	Testprotokoll	28
10.3	Zusammenfassung	29
11	GUI-Funktionalität - Testlauf 1 (2013 - 07 - 09)	30
11.1	Testumgebung	30
11.2	Testprotokoll	30
11.3	Zusammenfassung	31
12	Datenübertragungs- und Speichertest - Testlauf 1 (2013 - 07 - 09)	32
12.1	Testumgebung	32
12.2	Testprotokoll	32
12.3	Zusammenfassung	33
13	Nachfolgetest Webservice über SoapUI - Testlauf 1 (2013-07-10)	34
13.1	Testumgebung	34
13.2	Testprotokoll	34
13.3	Zusammenfassung	35
14	Integration von Client und Webservice - Testlauf 1 (2013 - 07 - 10)	36
14.1	Testumgebung	36
14.2	Testprotokoll	36
14.3	Zusammenfassung	37
15	Integration von Client und Webservice - Testlauf 2 (2013 - 07 - 10)	38
15.1	Testumgebung	38
15.2	Testprotokoll	38
15.3	Zusammenfassung	39
16	Integration von Client und Webservice - Testlauf 3 (2013 - 07 - 10)	40
16.1	Testumgebung	40

16.2 Testprotokoll	40
16.3 Zusammenfassung	41

1 Einfügen in App-Datenbank - Testlauf 1 (2013-06-29)

In dieser Testdurchführung wird getestet, dass in der Datenbank des Clients die korrekten Daten gespeichert werden. Dabei sollen Daten nur gespeichert werden, wenn die GPS-Position und der Zeitpunkt bekannt sind.

Dieser Testlauf beschreibt den Versuch einen korrekten Datensatz in der Datenbank zu speichern.

Art des Test: Abnahmetest

Ausgeführte Testfälle: $\langle T400 \rangle$

Beteiligte Tester: Julian Troegel

Abgedeckte Funktionen: $\langle F50 \rangle$, $\langle F60 \rangle$

1.1 Testumgebung

Die Testfälle werden unter Eclipse auf einem Android-Mobilgerät oder dem von Eclipse bereitgestellten Android-Emulator durchgeführt. Die Auswertung der JUnit-Tests erfolgt in Eclipse.

1.2 Testprotokoll

Testfall	$\langle T400 \rangle$
Tester	<i>Julian Troegel</i>
Eingaben	<i>Es müssen zuerst ein Datenbankobjekt, um auf diesen Daten einfügen zu können, und ein DBDataSet, da dieser Datensatz in der Datenbank eingefügt werden, erstellt werden. Damit ein Datensatz gespeichert werden kann, müssen die GPS-Position und der Zeitpunkt bekannt sein. Um dieses zu erkennen, müssen sowohl Longitude, als auch Latitude einen Wert besitzen, der größer als -1000 ist und der Zeitpunkt muss einen positiven Wert besitzen. Longitude: 10.86533 (> -1000); Latitude: -5.686135 (> -1000); Zeitpunkt: 84645351 (> 0)</i>

Soll - Reaktion	<i>Durch das Einfügen erhält man die von der Datenbank automatisch generierte id, des eingefügten Datensatzes. Je nachdem wie viele Datensätze schon vorher in der Datenbank sind, wird die ID einen größeren Wert haben. Wenn der Datensatz nicht eingefügt wird, ist die ID -1. Die ID des Eingefügten Datensatz soll echtgrößer 0 sein.</i>
Ist - Reaktion	<i>Es wurde geprüft, ob $ID > 0$ ist. Das Ergebnis ist true.</i>
Ergebnis	<i>Der Test lief erfolgreich. Die Daten wurden in der Datenbank eingefügt.</i>

1.3 Zusammenfassung

Da die nachfolgenden Testdurchführungen diesem Test entsprechen, nur mit anderen Eingabedaten, wird die Zusammenfassung in Abschnitt 4.3 erläutert.

2 Einfügen in App-Datenbank - Testlauf 2 (2013-06-29)

In dieser Testdurchführung wird getestet, dass in der Datenbank des Clients die korrekten Daten gespeichert werden. Dabei sollen Daten nur gespeichert werden, wenn die GPS-Position ist und der Zeitpunkt bekannt sind.

Dieser Testlauf beschreibt den Versuch einen nicht gültigen Datensatz auch nicht in der Datenbank zu speichern. Dabei ist weder die GPS-Position, als auch die Zeit bekannt.

Art des Test: Abnahmetest

Ausgeführte Testfälle: $\langle T400 \rangle$

Beteiligte Tester: Julian Troegel

Abgedeckte Funktionen: $\langle F50 \rangle$, $\langle F60 \rangle$

2.1 Testumgebung

Die Testfälle werden unter Eclipse auf einem Android-Mobilgerät oder dem von Eclipse bereitgestellten Android-Emulator durchgeführt. Die Auswertung der JUnit-Tests erfolgt in Eclipse.

2.2 Testprotokoll

Testfall	$\langle T400 \rangle$
Tester	<i>Julian Troegel</i>
Eingaben	<i>Es müssen zuerst ein Datenbankobjekt, um auf diesen Daten einfügen zu können, und ein DBDataSet, da dieser Datensatz in der Datenbank eingefügt werden, erstellt werden. Damit ein Datensatz gespeichert werden kann, müssen die GPS-Position und der Zeitpunkt bekannt sein. Um dieses zu erkennen, müssen sowohl Longitude, als auch Latitude einen Wert besitzen, der größer als -1000 ist und der Zeitpunkt muss einen positiven Wert besitzen. Longitude: -1000; Latitude: -1000; Zeitpunkt: -1 (< 0)</i>

Soll - Reaktion	<i>Durch das Einfügen erhält man die von der Datenbank automatisch generierte id, des eingefügten Datensatzes. Je nachdem wie viele Datensätze schon vorher in der Datenbank sind, wird die ID einen größeren Wert haben. Wenn der Datensatz nicht eingefügt wird, ist die ID -1. Die ID des eingefügten Datensatzes soll -1 sein.</i>
Ist - Reaktion	<i>Es wurde geprüft, ob $ID = -1$ ist. Das Ergebnis ist true.</i>
Ergebnis	<i>Der Test lief erfolgreich. Die $ID = -1$ und kann damit keine in der Datenbank legitime ID sein. Die Daten wurden also nicht in der Datenbank eingefügt.</i>

2.3 Zusammenfassung

Da die nachfolgenden Testdurchführungen diesem Test entsprechen, nur mit anderen Eingabedaten, wird die Zusammenfassung in Abschnitt 4.3 erläutert.

3 Einfügen in App-Datenbank - Testlauf 3 (2013-06-29)

In dieser Testdurchführung wird getestet, dass in der Datenbank des Clients die korrekten Daten gespeichert werden. Dabei sollen Daten nur gespeichert werden, wenn die GPS-Position ist und der Zeitpunkt bekannt sind.

Dieser Testlauf beschreibt den Versuch einen nicht gültigen Datensatz auch nicht in der Datenbank zu speichern. Dabei ist die GPS-Position, nicht bekannt, aber dafür die Zeit.

Art des Test: Abnahmetest

Ausgeführte Testfälle: $\langle T400 \rangle$

Beteiligte Tester: Julian Troegel

Abgedeckte Funktionen: $\langle F50 \rangle$, $\langle F60 \rangle$

3.1 Testumgebung

Die Testfälle werden unter Eclipse auf einem Android-Mobilgerät oder dem von Eclipse bereitgestellten Android-Emulator durchgeführt. Die Auswertung der JUnit-Tests erfolgt in Eclipse.

3.2 Testprotokoll

Testfall	$\langle T400 \rangle$
Tester	<i>Julian Troegel</i>
Eingaben	<i>Es müssen zuerst ein Datenbankobjekt, um auf diesen Daten einfügen zu können, und ein DBDataSet, da dieser Datensatz in der Datenbank eingefügt werden, erstellt werden. Damit ein Datensatz gespeichert werden kann, müssen die GPS-Position und der Zeitpunkt bekannt sein. Um dieses zu erkennen, müssen sowohl Longitude, als auch Latitude einen Wert besitzen, der größer als -1000 ist und der Zeitpunkt muss einen positiven Wert besitzen. Longitude: -1000; Latitude: -1000; Zeitpunkt: 1 (> 0)</i>

Soll - Reaktion	<i>Durch das Einfügen erhält man die von der Datenbank automatisch generierte id, des eingefügten Datensatzes. Je nachdem wie viele Datensätze schon vorher in der Datenbank sind, wird die ID einen größeren Wert haben. Wenn der Datensatz nicht eingefügt wird, ist die ID -1. Die ID des Eingefügten Datensatz soll -1 sein.</i>
Ist - Reaktion	<i>Es wurde geprüft, ob $ID = -1$ ist. Das Ergebnis ist true.</i>
Ergebnis	<i>Der Test lief erfolgreich. Die $ID = -1$ und kann damit keine in der Datenbank legitime ID sein. Die Daten wurden also nicht in der Datenbank eingefügt.</i>

3.3 Zusammenfassung

Da die nachfolgende Testdurchführung diesem Test entsprechen, nur mit anderen Eingabedaten, wird die Zusammenfassung in Abschnitt 4.3 erläutert.

4 Einfügen in App-Datenbank - Testlauf 4 (2013-06-29)

In dieser Testdurchführung wird getestet, dass in der Datenbank des Clients die korrekten Daten gespeichert werden. Dabei sollen Daten nur gespeichert werden, wenn die GPS-Position ist und der Zeitpunkt bekannt sind.

Dieser Testlauf beschreibt den Versuch einen nicht gültigen Datensatz auch nicht in der Datenbank zu speichern. Dabei ist die GPS-Position bekannt, aber nicht die Zeit.

Art des Test: Abnahmetest

Ausgeführte Testfälle: $\langle T400 \rangle$

Beteiligte Tester: Julian Troegel

Abgedeckte Funktionen: $\langle F50 \rangle$, $\langle F60 \rangle$

4.1 Testumgebung

Die Testfälle werden unter Eclipse auf einem Android-Mobilgerät oder dem von Eclipse bereitgestellten Android-Emulator durchgeführt. Die Auswertung der JUnit-Tests erfolgt in Eclipse.

4.2 Testprotokoll

Testfall	$\langle T400 \rangle$
Tester	<i>Julian Troegel</i>
Eingaben	<i>Es müssen zuerst ein Datenbankobjekt, um auf diesen Daten einfügen zu können, und ein DBDataSet, da dieser Datensatz in der Datenbank eingefügt werden, erstellt werden. Damit ein Datensatz gespeichert werden kann, müssen die GPS-Position und der Zeitpunkt bekannt sein. Um dieses zu erkennen, müssen sowohl Longitude, als auch Latitude einen Wert besitzen, der größer als -1000 ist und der Zeitpunkt muss einen positiven Wert besitzen. Longitude: 0 (> -1000); Latitude: 0 (> -1000); Zeitpunkt: -1 (< 0)</i>

Soll - Reaktion	<i>Durch das Einfügen erhält man die von der Datenbank automatisch generierte id, des eingefügten Datensatzes. Je nachdem wie viele Datensätze schon vorher in der Datenbank sind, wird die ID einen größeren Wert haben. Wenn der Datensatz nicht eingefügt wird, ist die ID -1. Die ID des Eingefügten Datensatz soll = -1 sein.</i>
Ist - Reaktion	<i>Es wurde geprüft, ob ID = -1 ist. Das Ergebnis ist true.</i>
Ergebnis	<i>Der Test lief erfolgreich. Die ID ist = -1 und kann damit keine in der Datenbank legitime ID sein. Die Daten wurden also nicht in der Datenbank eingefügt.</i>

4.3 Zusammenfassung

Das Einfügen von Daten in die Datenbank funktioniert wie erwartet. Es werden nur Daten gespeichert, wenn die GPS und eine aktuelle Zeit vorhanden sind. Fehlt eins von beiden oder beides, so werden die Datensätze nicht gespeichert.

In dem Testfall $\langle T400 \rangle$ geht es aber auch noch weiter darum, dass der Benutzer manche Daten auswählen kann, welche dann nicht gespeichert werden sollen. Dabei sollte überprüft werden, dass auch nur die Daten gespeichert werden sollen, die der Benutzer zulässt. Dies wurde aber in den durchgeführten Tests nicht überprüft, da dazu der von Android bereitgestellte “SharedPreferences” genutzt werden. Diese sind dazu da, um Benutzereinstellungen zu schreiben und lesen. Da diese von Android bereitgestellt werden, ist dafür kein Test notwendig.

5 Funktionstest Webservice über SoapUI - Testlauf 1 (2013-06-29)

Art des Tests: Black-Box-Test/Abnahmetest

Ausgeführte Testfälle: $\langle T1000 \rangle$ mit Simulation der Server-Client-Kommunikation aus $\langle T100 \rangle$

Beteiligte Tester: Timo Schwarz

Abgedeckte Funktionen: $\langle F30 \rangle$, $\langle F80 \rangle$, $\langle F90 \rangle$, $\langle F100 \rangle$, $\langle F110 \rangle$, teilweise $\langle F130 \rangle$

Dieser Test enthält unter Anderem alle Funktionen, die im Testfall $\langle T100 \rangle$ benötigt werden, weshalb dieser Teil als Abnahmetest parallel geführt wird.

5.1 Testumgebung

Die Testfälle wurden unter Windows 7 mithilfe von SoapUI in Version 4.5.2 über eine VPN-Verbindung zum Netz der TU durchgeführt.

Die Funktionen wurden manuell aufgerufen.

5.2 Testprotokoll

Testfall	$\langle T1000 \rangle$ und $\langle T100 \rangle$
Tester	<i>Timo Schwarz</i>
Eingaben	<i>Registrieren eines neuen Users bzw. Login-Daten:</i> <ul style="list-style-type: none">• Benutzername: <i>test2@test.com</i>• Passwort: <i>testpw</i>

Eingaben	<p><i>Registrieren eines neuen Gerätes:</i></p> <ul style="list-style-type: none">• <i>netOp:</i> 33333• <i>netOpName:</i> Purple• <i>countryCode:</i> gb• <i>manufacturer:</i> CTH• <i>model:</i> CTH 6• <i>phoneType:</i> 2 <p><i>Übertragen eines Datensatzes:</i></p> <ul style="list-style-type: none">• <i>gsmSignalStrength:</i> 23• <i>gsmBitErrorRate:</i> 1• <i>networkType:</i> 4• <i>longitude:</i> -23.5• <i>latitude:</i> 34.671• <i>cellID:</i> 4356• <i>lac:</i> 5• <i>networkOperator:</i> 56743• <i>netOpName:</i> TOP• <i>countryCode:</i> de• <i>timestamp:</i> 29.06.2013 20:15 <p><i>Anfordern von Gerätedaten:</i></p> <ul style="list-style-type: none">• <i>Hersteller:</i> CTH <p><i>Das Anfordern von Hersteller- und Providerdaten hat jeweils keine Eingabeparameter.</i></p>
-----------------	---

Soll - Reaktion	<ul style="list-style-type: none"> • Registrieren (A): Eine positive Zahl (UserID) • Login (A): Eine positive Zahl (SessionID) • Registrieren eines Mobilgerätes (M): Eine positive Zahl (UEID) • Datenübertragung (M): true • Liste der Provider (W): Ein Array mit allen gespeicherten Providerdaten • Liste der Hersteller (W): Ein Array mit allen gespeicherten Herstellerdaten • Liste der Geräte (W): Ein Array mit allen gespeicherten Gerätedaten eines Herstellers • endSession (A): true
Ist – Reaktion	<ul style="list-style-type: none"> • Registrieren (A): 14 • Login (A): 1496791628 • Registrieren eines Mobilgerätes (M): 5 • Datenübertragung (M): true • Liste der Provider(W) : Ein korrektes Array mit allen gespeicherten Providerdaten • Liste der Hersteller (W): Ein nicht korrektes Array mit allen gespeicherten Herstellerdaten, das Array enthielt ein Duplikat • Liste der Geräte (W): Eine Exception von SoapUI • endSession (A): true <p>Zuordnung zu Klassen von Kommunikationspartnern des WebService: A = Alle, M = Mobilgeräte, W = WebApp</p>

Ergebnis	<ul style="list-style-type: none">• <i>Alle Funktionen außer dem Anfordern der Liste der Hersteller bzw. der Geräte haben korrekt funktioniert.</i>• <i>Beim Generieren der Herstellerliste wurde bei der Datenbankabfrage das Keyword DISTINCT vergessen.</i>• <i>Als Rückgabewert der Geräteliste kommt ein mit SoapUI inkompatibler Datentyp zum Einsatz.</i>
Nacharbeiten	<ul style="list-style-type: none">• <i>Keyword DISTINCT an der richtigen Stelle in die Datenbankabfrage einfügen.</i>• <i>Primitiven Datentyp zur Rückgabe verwenden.</i>

5.3 Zusammenfassung

Es wurde die Funktionsfähigkeit der vom Webservice bereitgestellten Funktionen getestet.

Ein Großteil der getesteten Funktionen hat korrekt funktioniert. Insbesondere haben alle Funktionen, die für die Kommunikation zwischen Mobilgeräten und Webservice essentiell sind, korrekt funktioniert.

Als Folge des Tests werden die Rückgabetypen des Webservice vereinheitlicht, sodass keine Probleme mit eventuell unbekannten Datentypen mehr auftreten können.

6 Daten sammeln und Ergebnisse graphisch darstellen

- Testlauf 1 (2013-07-04)

In dieser Testdurchführung wird getestet, ob gespeicherte Daten korrekt visualisiert werden. Es wird dabei zwischen den Visualisierungsarten Tabelle, XY-Graph und Google Maps unterschieden.

Art des Test: Integrationstest über die GUI

Ausgeführte Testfälle: $\langle T300 \rangle$

Beteiligte Tester: Sofia Ananieva, Andreas Hecker

Abgedeckte Funktionen: $\langle F20 \rangle$, $\langle F50 \rangle$

6.1 Testumgebung

Die Testfälle werden unter Eclipse auf einem Android-Mobilgerät durchgeführt.

6.2 Testprotokoll

Testfall	$\langle T300 \rangle$
Tester	<i>Sofia Ananieva, Andreas Hecker</i>
Eingaben	<i>Nach dem Start der Applikation wird das Speichern von Datensätzen unter Einstellungen aktiviert und in gewissen Intervallen oder Echtzeit gemessen.</i>
Soll - Reaktion	<i>Die gemessenen Datensätze werden in der Datenbank gespeichert. Sie sind anschließend in allen Visualisierungsarten sichtbar.</i>
Ist - Reaktion	<i>Es wurden 8 Datensätze in der Datenbank gespeichert. Die Datensätze sind in allen Visualisierungsarten sichtbar und korrekt.</i>
Ergebnis	<i>Der Test verlief erfolgreich.</i>

6.3 Zusammenfassung

Die Visualisierung in der Tabelle verlief erfolgreich. Alle gemessenen Daten werden korrekt dargestellt.



Id:	Time:	Signal Strength:	Bit Error Rate:
8	04.07.2013, 17:44:45	-91	0
7	04.07.2013, 17:43:47	-79	0
6	04.07.2013, 17:42:39	-81	0
5	04.07.2013, 17:41:45	-87	0
4	04.07.2013, 17:40:41	-81	0
3	04.07.2013, 17:39:47	-85	0
2	04.07.2013, 17:37:19	-77	0
1	04.07.2013, 17:36:25	-91	0

Abbildung 6.1: Visualisierung in einer Tabelle $\langle F20 \rangle$

X-MAP

Android-App zur qualitativen Erfassung von Mobilfunknetzen

Die Visualisierung im XY-Graph verlief erfolgreich. Die Signalstärke wird zeitlich korrekt dargestellt.

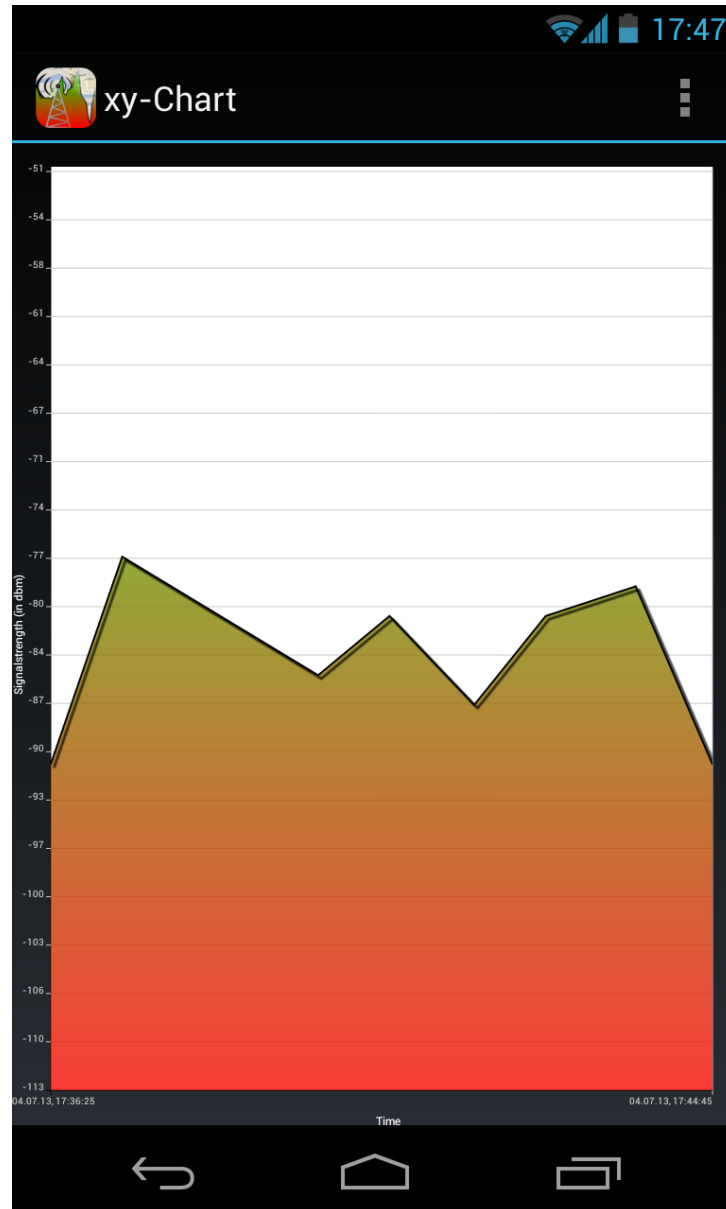


Abbildung 6.2: Visualisierung in einem XY-Graph $\langle F_{20} \rangle$

Die Visualisierung in Google Maps verlief erfolgreich. Die Position der Marker entspricht der Position beim Testen. Die Farbe der Marker wird entsprechend der Signalstärke dargestellt. Beim Anklicken der Marker wird die zugehörige Signalstärke eingeblendet. Die angezeigten Signalstärken entsprechen den Werten der anderen Visualisierungsarten.

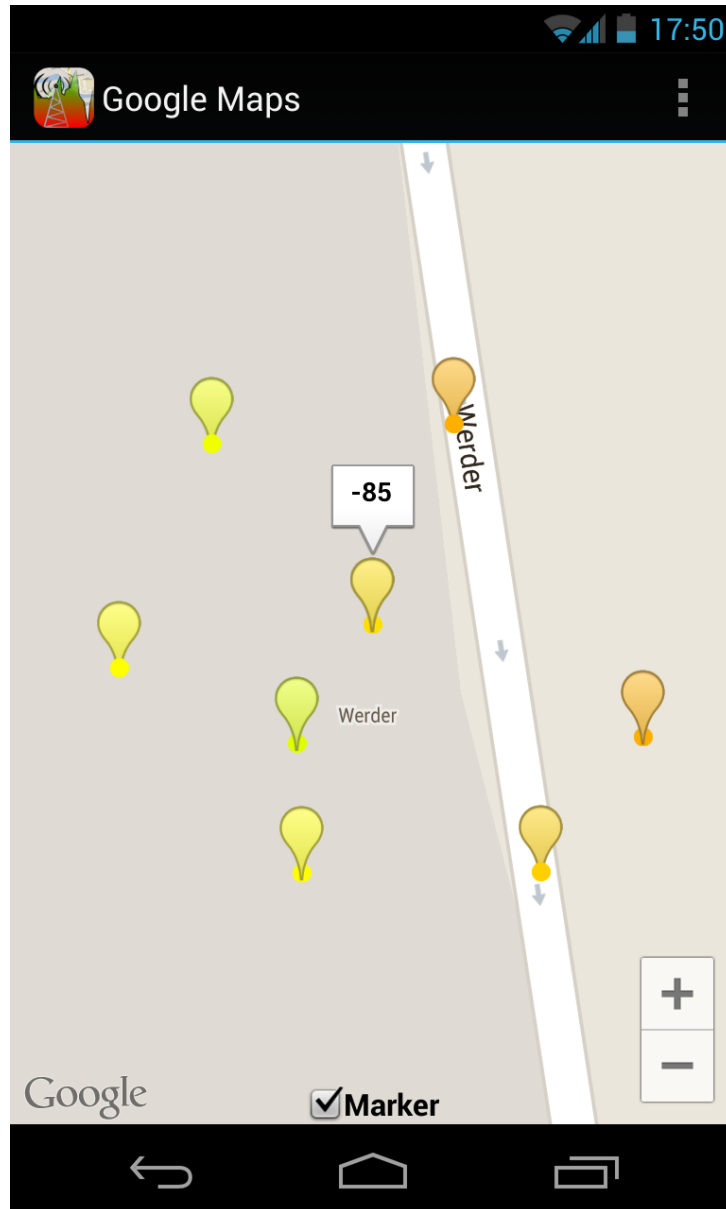


Abbildung 6.3: Visualisierung in Google Maps $\langle F'20 \rangle$

7 Applikation an 6 unabhängige Personen verteilt - Testlauf 1 (2013 - 07 - 06)

In dieser Testdurchführung prüfen externe Nutzer die Funktionalität der Applikation und geben ein abschließendes Feedback. Dabei kommen die unabhängigen Personen aus verschiedenen Gebieten Deutschlands.

Art des Tests: Abnahmetest

Ausgeführte Testfälle: $\langle T100 \rangle$, $\langle T200 \rangle$, $\langle T300 \rangle$ Fall [Anwendung], $\langle T400 \rangle$, $\langle T600 \rangle$, $\langle T800 \rangle$

Beteiligte Tester: (bislang) Sofia Ananieva, Andreas Hecker

Abgedeckte Funktionen: $\langle F10 \rangle$ - $\langle F100 \rangle$

7.1 Testumgebung

Die Testfälle wurden (bislang) auf einem HTC One, Sony Xperia Z, Samsung Galaxy Nexus, Samsung Nexus S, Samsung S2 und Samsung S4 getestet.

7.2 Testprotokoll

Testfall	$\langle T100 \rangle$, $\langle T200 \rangle$, $\langle T300 \rangle$ Fall [Anwendung], $\langle T400 \rangle$, $\langle T600 \rangle$, $\langle T800 \rangle$
Tester	<i>Sofia Ananieva, Andreas Hecker und 6 unabhängige Personen</i>
Eingaben	<i>Nach dem Start der Applikation werden alle bereitgestellten Funktionalitäten der App genutzt.</i>
Soll - Reaktion	<i>Reaktion: Messung, Visualisierung und nutzerdefinierte Einstellungen auf der App sowie Registrierung und Anmeldung am Webservice funktioniert.</i>

Ist - Reaktion	<i>Positives Feedback:</i> <ul style="list-style-type: none">• <i>Funktionalität ist vorhanden.</i>• <i>Optisch stimmige GUI.</i>• <i>Intuitive Nutzung.</i> <i>Negatives Feedback:</i> <ul style="list-style-type: none">• <i>Kein Nutzen bei Registrierung.</i>• <i>'Sync Now' Button schwer zu finden.</i>
Ergebnis	<i>Der Test verlief erfolgreich.</i>

7.3 Zusammenfassung

Die unabhängigen Personen waren zufrieden mit der Funktionalität und Nutzeroberfläche der App. Messung und Visualisierung auf der App sowie Registrierung und Anmeldung am WebService verliefen erfolgreich. Vom Nutzer vorgenommene Einstellungen, beispielsweise zur Intervallhäufigkeit der Messung, wurden übernommen. Gelobt wurden insbesondere die Visualisierungsmöglichkeiten der Messungen sowie die intuitive Nutzung der App.

In kommender Zeit soll es ermöglicht werden, sich nach einer Registrierung per App auf der Webseite der Webapplikation anzumelden und alle eigenen übermittelten Messdaten über GoogleMaps auszuwerten. Administratoren können Messdaten aller Nutzer auswerten.

8 Daten messen - Testlauf 1 (2013-07-09)

In dieser Testdurchführung wird getestet, ob Daten gemessen werden und danach auf dem Smartphone vorhanden sind.

Art des Test: Integrationstest über die GUI

Ausgeführte Testfälle: $\langle T800 \rangle$

Beteiligte Tester: Andreas Hecker, Sofia Ananieva

Abgedeckte Funktionen: $\langle F10 \rangle$

8.1 Testumgebung

Die Testfälle werden unter Eclipse auf einem Android-Mobilgerät durchgeführt.

8.2 Testprotokoll

Testfall	$\langle T800 \rangle$
Tester	<i>Andreas Hecker, Sofia Ananieva</i>
Eingaben	<i>Nach dem Start der Applikation wird das Messen von Datensätzen unter Einstellungen aktiviert und in bestimmten Intervallen oder in Echtzeit gemessen.</i>
Soll - Reaktion	<i>Die gemessenen Datensätze werden in der Datenbank gespeichert. Sie sind anschließend in allen Visualisierungsarten sichtbar. Die Messungen können variabel sein und sind ortsgebunden durch Längen- und Breitengrad. Im Startbildschirm sind "on-the-fly" Änderungen der Statuswerte zum Empfang beobachtbar, beispielsweise die Signalstärke.</i>
Ist - Reaktion	<i>Die gemessenen Datensätze sind variabel und in der Tabelle entsprechend eines bestimmten Intervalls (In diesem Testfall wurde die Messung 'Echtzeit' gewählt) festgehalten. Die Statuswerte im Startbildschirm verändern sich in kurzen Abständen.</i>
Ergebnis	<i>Der Test verlief erfolgreich.</i>

8.3 Zusammenfassung

Das Messen der Daten verlief erfolgreich. Auf der Hauptseite werden alle gemessenen Daten korrekt dargestellt und in Echtzeit aktualisiert.

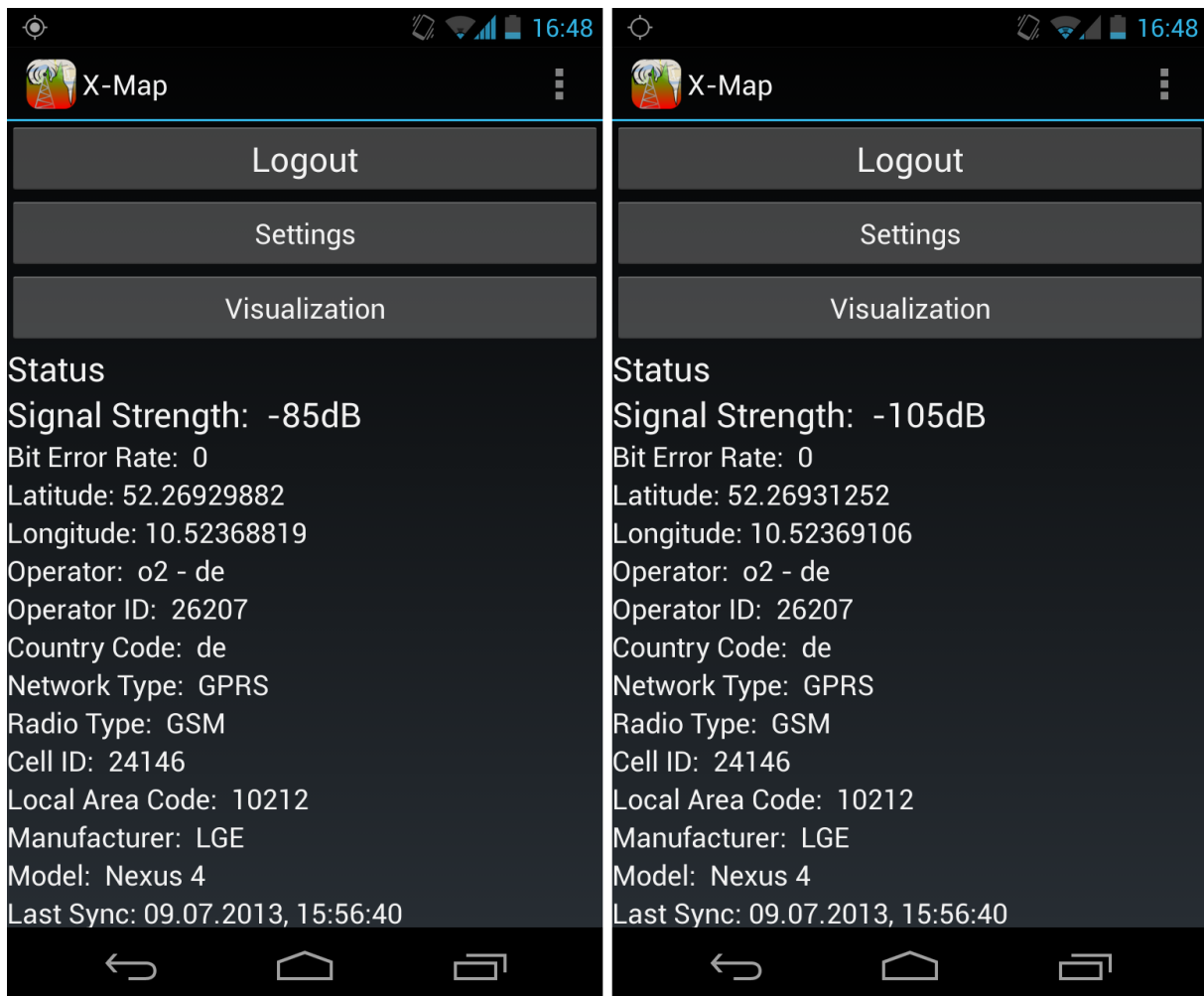
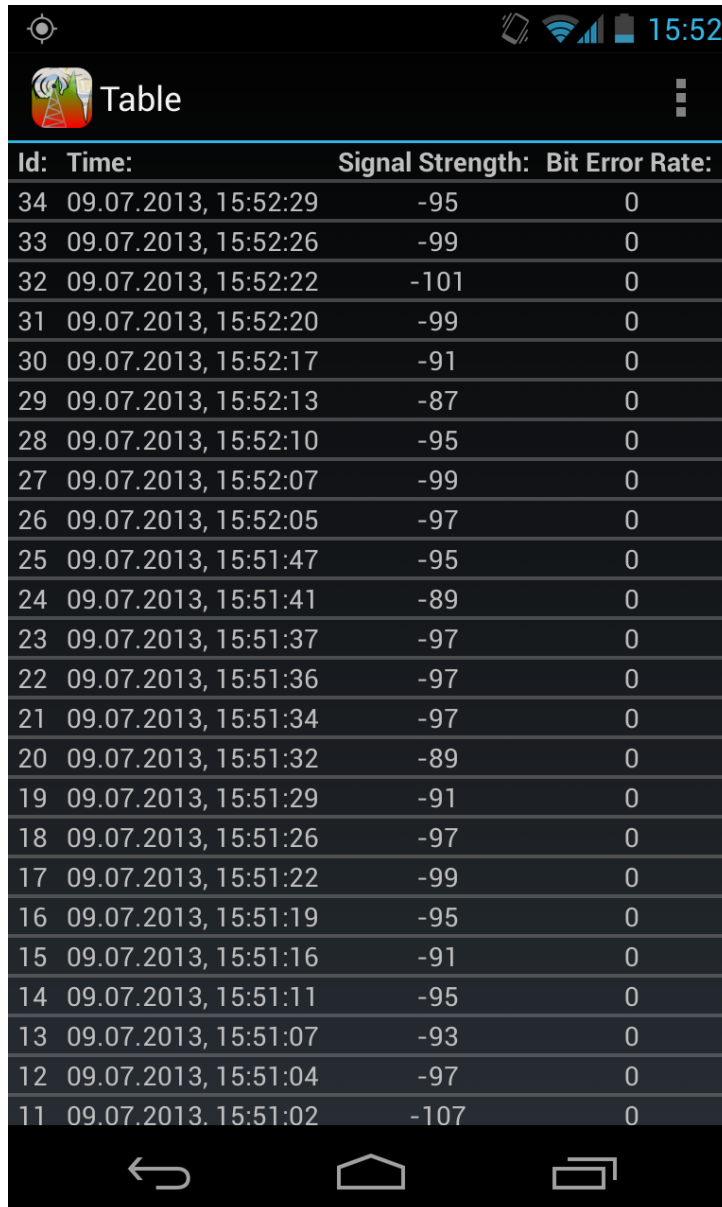


Abbildung 8.1: MainActivity (F10)

In der Visualisierung 'Tabelle' ist zu erkennen, dass sich die gemessene Empfangsstärke im Laufe der Zeit verändert.



The screenshot shows the 'Table' app interface on an Android device. The status bar at the top displays the time 15:52 and various icons. The app title 'Table' is at the top left. Below the title is a table with four columns: 'Id', 'Time:', 'Signal Strength:', and 'Bit Error Rate:'. The table contains 24 rows of data, showing a decreasing trend in signal strength from -95 to -107 dBm over time. The bottom of the screen shows the Android navigation bar with back, home, and recent apps buttons.

Id	Time:	Signal Strength:	Bit Error Rate:
34	09.07.2013, 15:52:29	-95	0
33	09.07.2013, 15:52:26	-99	0
32	09.07.2013, 15:52:22	-101	0
31	09.07.2013, 15:52:20	-99	0
30	09.07.2013, 15:52:17	-91	0
29	09.07.2013, 15:52:13	-87	0
28	09.07.2013, 15:52:10	-95	0
27	09.07.2013, 15:52:07	-99	0
26	09.07.2013, 15:52:05	-97	0
25	09.07.2013, 15:51:47	-95	0
24	09.07.2013, 15:51:41	-89	0
23	09.07.2013, 15:51:37	-97	0
22	09.07.2013, 15:51:36	-97	0
21	09.07.2013, 15:51:34	-97	0
20	09.07.2013, 15:51:32	-89	0
19	09.07.2013, 15:51:29	-91	0
18	09.07.2013, 15:51:26	-97	0
17	09.07.2013, 15:51:22	-99	0
16	09.07.2013, 15:51:19	-95	0
15	09.07.2013, 15:51:16	-91	0
14	09.07.2013, 15:51:11	-95	0
13	09.07.2013, 15:51:07	-93	0
12	09.07.2013, 15:51:04	-97	0
11	09.07.2013, 15:51:02	-107	0

Abbildung 8.2: Visualisierung in einer Tabelle $\langle F10 \rangle$

9 Löschen von Datensätzen - Testlauf 1 (2013-07-08)

In dieser Testdurchführung wird getestet, dass bei der Überschreitung der maximal eingestellten Datenbankgröße der App so viele der ältesten Datensätze gelöscht werden, bis die Datenbank die maximale Speichergröße wieder unterschreitet.

Beispiel:

Art des Tests: Abnahmetest

Ausgeführte Testfälle: $\langle T600 \rangle$

Beteiligte Tester: Julian Troegel

Abgedeckte Funktionen: $\langle F40 \rangle$, $\langle F50 \rangle$, $\langle F60 \rangle$

9.1 Testumgebung

Der Testfall wird unter Eclipse auf einem von Eclipse bereitgestellten Android-Emulator durchgeführt. Die Datenbank in der App auf dem Emulator wurde mit “Dummy“-Daten gefüllt, damit sie voll ist.

9.2 Testprotokoll

Testfall	$\langle T600 \rangle$
Tester	<i>Julian Troegel</i>
Eingaben	<i>Zu diesem Test gibt es keine direkten Eingaben. Die Datenbank der App muss von der Speichergröße kurz vor der Überschreitung der maximalen Datenbankgröße sein und es muss ein Datensatz eingefügt werden, so dass die Datenbank die maximale Größe überschreitet.</i> <i>In diesem Test betragen die maximal eingestellte und die aktuelle Datenbankgröße 10485760 Bytes (10 MByte). Die Datenbank ist also voll gefüllt. Außerdem wird ein neuer Datensatz in der Datenbank gespeichert.</i>

Soll - Reaktion	<i>Die Datenbankgröße beträgt nach dem Einfügen des neuen Datensatzes wieder die maximal eingestellte Datenbankgröße. Die ältesten Datensätze wurden gelöscht.</i>
Ist – Reaktion	Die Datenbankgröße beträgt nach dem Einfügen 10485760 Bytes. Es wurde neben dem Einfügen ein Datensatz gelöscht. Dass der gelöschte Datensatz der älteste Datensatz ist, ist nicht testbar. Da es sich um einen Android-Methodenaufruf auf die Datenbank handelt, ist die Korrektheit aber sichergestellt.
Ergebnis	<i>Der Test verlief erfolgreich. Die Datenbank überschreitet beim Einfügen eines Datensatzes nicht die Maximalgröße. Es wird die dazugehörige Anzahl von Datensätzen gelöscht.</i>

9.3 Zusammenfassung

Der Test verlief erfolgreich. Die Datenbank überschreitet nicht die maximaleingestellte Datenbankgröße es wird dafür gesorgt, dass sobald die Größe überschritten wird, so lange Daten gelöscht werden, bis sie darunter liegt.

Der Vorgang kann aber sehr lange dauern, da immer nur ein Datensatz auf einmal gelöscht wird. Wenn der Unterschied von aktueller und maximaler Datenbankgröße sehr groß ist, müssen mehrere Datensätze jeweils einzeln und nacheinander gelöscht werden.

10 Löschen von Datensätzen - Testlauf 2 (2013-07-09)

In dieser Testdurchführung wird getestet, dass bei der Überschreitung der maximal eingestellten Datenbankgröße der App so viele der ältesten Datensätze gelöscht, bis die Datenbank die maximale Speichergröße wieder unterschreitet. Dabei stellt der Benutzer die maximale Speichergröße auf eine unter der aktuell liegenden Größe ein.

Beispiel:

Art des Tests: Abnahmetest

Ausgeführte Testfälle: $\langle T600 \rangle$

Beteiligte Tester: Julian Troegel

Abgedeckte Funktionen: $\langle F40 \rangle$, $\langle F50 \rangle$, $\langle F60 \rangle$

10.1 Testumgebung

Der Testfall wird unter Eclipse auf einem von Eclipse bereitgestellten Android-Emulator durchgeführt. Die Datenbank in der App auf dem Emulator wurde mit "Dummy"-Daten gefüllt, damit sie voll ist.

10.2 Testprotokoll

Testfall	$\langle T600 \rangle$
Tester	<i>Julian Troegel</i>
Eingaben	<i>Die aktuell maximale Datenbankgröße beträgt 52428800 Bytes (50 MByte) und die aktuelle Datenbankgröße beträgt 13557760 Bytes. Die maximale Datenbankgröße wird auf 10 MByte (10485760 Bytes) gesenkt.</i>
Soll - Reaktion	<i>Die Datenbankgröße beträgt 10485760 Bytes. Es wurden nur die ältesten Datensätze gelöscht.</i>

Ist – Reaktion	Die Datenbankgröße beträgt nach dem Einstellen 10485760 Bytes. Es wurden also mehrere Datensätze gelöscht. Dass der gelöschte Datensatz der älteste Datensatz ist, ist nicht testbar. Da es sich um einen Android-Methodenaufruf auf die Datenbank handelt, ist die Korrektheit aber sichergestellt.
Ergebnis	<i>Der Test verlief erfolgreich. Die Datenbank überschreitet bei der Einstellungsänderung der maximalen Datenbankgröße die Maximalgröße. Es wird die dazugehörige Anzahl von Datensätzen gelöscht, bis die Größe unter der eingestellten liegt.</i>

10.3 Zusammenfassung

Der Test verlief erfolgreich. Die Datenbank überschreitet nicht die maximal eingestellte Datenbankgröße es wird dafür gesorgt, dass sobald die Größe überschritten wird, so lange Daten gelöscht werden, bis sie darunter liegt.

Der Vorgang kann aber sehr lange dauern, da immer nur ein Datensatz auf einmal gelöscht wird. Wenn der Unterschied von aktueller und maximaler Datenbankgröße sehr groß ist, müssen mehrere Datensätze jeweils einzeln und nacheinander gelöscht werden.

11 GUI-Funktionalität - Testlauf 1 (2013 - 07 - 09)

In dieser Testdurchführung wird die korrekte Anwendung der GUI getestet.

Art des Tests: Abnahmetest

Ausgeführte Testfälle: $\langle T100 \rangle$, $\langle T200 \rangle$, $\langle T300 \rangle$ Fall [Anwendung], $\langle T400 \rangle$, $\langle T600 \rangle$, $\langle T800 \rangle$

Beteiligte Tester: Julian Troegel, Timo Schwarz

Abgedeckte Funktionen: $\langle F10 \rangle$ - $\langle F100 \rangle$

11.1 Testumgebung

Die Testfälle der GUI wurden auf einem HTC One mit Android 4.1 und auf einem Samsung Galaxy Ace mit Android 2.3.5 getestet.

11.2 Testprotokoll

Testfall	$\langle T100 \rangle$, $\langle T200 \rangle$, $\langle T300 \rangle$ Fall [Anwendung], $\langle T400 \rangle$, $\langle T600 \rangle$, $\langle T800 \rangle$
Tester	Julian Troegel, Timo Schwarz
Eingaben	Nach dem Start der Applikation werden alle enthaltenden GUI-Funktionalitäten der App getestet.
Soll - Reaktion	Jedes GUI-Element vollführt die gewünschte Funktion, z.B. Start einer neuen passenden Benutzeroberfläche.
Ist - Reaktion	Alle GUI-Elemente führen die gewünschten Operationen aus. Die während der aktiven Messung angezeigte Benachrichtigung, in der Benachrichtigungsleiste vom Android-Betriebssystem, reagierte auf den Klick nicht.

Ergebnis	<i>Der Test verlief nicht erfolgreich. Die Benachrichtigung hatte keine Klick-Funktion. Bei Klick sollte der Startbildschirm der Applikation angezeigt werden. Alle anderen GUI-Funktionalitäten wurden erfüllt.</i>
Nacharbeiten	<i>Ein Bildschirm wird in Android durch ein vom Android SDK bereitgestellten Intent gestartet. Der Intent der Benachrichtigung war falsch definiert und muss passend definiert werden.</i>

11.3 Zusammenfassung

Die GUI-Elemente reagieren alle korrekt und erfüllen das Gewünschte. Bei der Benachrichtigung wurden dem Intent weitere Aktionen hinzugefügt, so dass immer der aktuell angezeigte Bildschirm geöffnet wird. Dadurch wirkte die Benachrichtigung, als ob sie keine Reaktion zeigt. Die Aktionen wurden entfernt und die Benachrichtigung öffnet auf Klick die Applikation.

12 Datenübertragungs- und Speichertest - Testlauf 1 (2013 - 07 - 09)

Dieser Test überprüft das korrekte Zusammenspiel aller für die Datenspeicherung auf dem Server notwendigen Funktionen.

Art des Tests: Abnahmetest

Ausgeführte Testfälle: $\langle T500 \rangle$

Beteiligte Tester: Timo Schwarz, Julian Troegel

Abgedeckte Funktionen: $\langle F30 \rangle$, $\langle F110 \rangle$

Hinweis: Das Pass-Kriterium, dass identische Datensätze nicht zwei mal in der Datenbank des Webservice eingetragen werden dürfen, ist nicht mehr gültig. Aus sicherheitstechnischen Gründen ist diese Funktion technisch nicht mehr realisierbar, da sonst die Traces eines Users genau nachvollzogen werden könnten.

12.1 Testumgebung

Der Test wurde nacheinander mit verschiedenen Android-Handys mit der X-Map-App über eine stabile WLAN-Verbindung durchgeführt. Die Datenbank wurde sowohl durch direkten Zugriff als auch durch die Anzeige der zuletzt eingefügten Werte über die WebApp überwacht.

12.2 Testprotokoll

Testfall	$\langle T500 \rangle$
Tester	<i>Timo Schwarz, Julian Troegel</i>
Eingaben	<i>In den Handys wurde zunächst die automatische Datenübertragung unterbunden und der Modus „Echtzeitmessung“ eingestellt, um eine signifikante Menge an Daten anzusammeln. Sobald sich eine angemessene Menge (mindestens 50 Messwerte) angesammelt hatte, wurde die Option „Jetzt synchronisieren“ gewählt.</i>
Soll - Reaktion	<i>Datensätze tauchen, nachdem die Handys das Ende der Übertragung melden, in der Datenbank und in der WebApp auf.</i>

Ist - Reaktion	<i>Keine Abweichung zur Soll-Reaktion</i>
Ergebnis	<i>Test erfolgreich</i>

12.3 Zusammenfassung

Mit verschiedenen Handys wurde nacheinander die Übertragung echter Datensätze getestet. Dabei gab es keinerlei Probleme oder Auffälligkeiten. Alle Tests mit allen Handys waren vollumfänglich erfolgreich.

13 Nachfolgetest Webservice über SoapUI - Testlauf 1 (2013-07-10)

Dieser Test ist als direkter Nachfolgetest zu Kapitel 5, Funktionstest Webservice über SoapUI - Testlauf 1 (2013-06-29), konzipiert. Es werden lediglich die Funktionen für das Anfordern von Hersteller- und Geräte-Arrays getestet, welche im erwähnten Testlauf fehlschlagen.

Art des Tests: Black-Box-Test

Ausgeführte Testfälle: $\langle T1000 \rangle$

Beteiligte Tester: Timo Schwarz

Abgedeckte Funktionen: keine komplett, Teilfunktionen von $\langle F130 \rangle$

13.1 Testumgebung

Die Testfälle wurden unter Windows 7 mithilfe von SoapUI in Version 4.5.2 durchgeführt. Im Gegensatz zum vorherigen Test ist kein VPN mehr nötig.

Die Funktionen wurden manuell aufgerufen.

13.2 Testprotokoll

Testfall	$\langle T1000 \rangle$
Tester	<i>Timo Schwarz</i>
Eingaben	<i>Anfordern von Gerätedaten:</i> <ul style="list-style-type: none">• Hersteller: <i>HTC</i> <i>Das Anfordern von Herstellerdaten hat keine Eingabewerte.</i>

Soll - Reaktion	<ul style="list-style-type: none">• Liste der Hersteller: Ein Array mit allen gespeicherten Herstellerdaten• Liste der Geräte: Ein Array mit allen gespeicherten Gerätedaten eines Herstellers
Ist – Reaktion	<ul style="list-style-type: none">• Liste der Hersteller: Ein Array mit allen gespeicherten Herstellerdaten• Liste der Geräte: Ein Array mit allen gespeicherten Gerätedaten eines Herstellers
Ergebnis	<i>Alle Funktionen wurden einwandfrei ausgeführt.</i>

13.3 Zusammenfassung

Die nachgetesteten Funktionen arbeiten nun exakt wie geplant. Somit können die vorgenommenen Änderungen seit dem besagten Test als Erfolg angesehen werden.

14 Integration von Client und Webservice - Testlauf 1 (2013 - 07 - 10)

Bei dieser Testdurchführung wird getestet, ob die Daten, die von der einen Komponente korrekt versendet werden bei der anderen Komponente und bei der Übertragung nicht verändert werden.

Art des Tests: Integrationstest

Ausgeführte Testfälle: $\langle T900 \rangle$

Beteiligte Tester: Andreas Bauerfeld

Abgedeckte Funktionen: $\langle F30 \rangle + \langle F110 \rangle, \langle F80 \rangle + \langle F100 \rangle, \langle F90 \rangle$

14.1 Testumgebung

Dieser Testfall wird durch Logcat von Eclipse auf Clientseite und Wireshark auf Serverseite ausgeführt

14.2 Testprotokoll

Testfall	$\langle T900 \rangle$
Tester	<i>Andreas Bauerfeld</i>
Eingaben	<i>Auf einem Android-Mobilgerät wird X-Map gestartet und im Registerformular die E-Mail-Adresse test@test.de als Benutzerkennung und testpassword als Passwort zweimal zur Registrierung eingegeben.</i>
Soll - Reaktion	<i>Die auf LogCat und Wireshark gleichzeitig schrittweise sichtbaren Nachrichten sind dieselben. Sie enthalten die gleichen Daten.</i>
Ist – Reaktion	<i>Das Benutzerkonto wurde auf dem Webservice erstellt und die eigenen Testdaten wurden dem Webservice korrekt und vollständig übermittelt.</i>
Ergebnis	<i>Test erfolgreich</i>

14.3 Zusammenfassung

Das Anlegen des Benutzerkontos wurde erfolgreich durchgeführt. Während der Registrierung hat die Überwachung von LogCat und Wireshark gezeigt, dass die zuvor auf dem Client eingegebenen Daten denen entsprechen, die hinterher auf dem Webservice ankamen, und bei der Übertragung keinerlei Daten verändert wurden.

15 Integration von Client und Webservice - Testlauf 2 (2013 - 07 - 10)

Bei dieser Testdurchführung wird getestet, ob die Daten, die von der einen Komponente korrekt versendet werden bei der anderen Komponente und bei der Übertragung nicht verändert werden.

Art des Tests: Integrationstest

Ausgeführte Testfälle: $\langle T900 \rangle$

Beteiligte Tester: Andreas Bauerfeld, Julian Troegel

Abgedeckte Funktionen: $\langle F30 \rangle + \langle F110 \rangle, \langle F80 \rangle + \langle F100 \rangle, \langle F90 \rangle$

15.1 Testumgebung

Dieser Testfall wird durch Logcat von Eclipse auf Clientseite und Wireshark auf Serverseite ausgeführt

15.2 Testprotokoll

Testfall	$\langle T900 \rangle$
Tester	<i>Andreas Bauerfeld, Julian Troegel</i>
Eingaben	<i>Auf einem Android-Mobilgerät wird X-Map gestartet und die E-Mail-Adresse test@test.de ist schon registriert. Im Anmeldeformular wird die E-Mail-Adresse test@test.de als Benutzerkennung und testpassword als Passwort zur Anmeldung eingegeben. Nach erfolgreichem Login beginnt die Datenübertragung.</i>
Soll - Reaktion	<i>Die auf LogCat und Wireshark gleichzeitig schrittweise sichtbaren Nachrichten sind dieselben. Sie enthalten die gleichen Daten. Der Benutzer ist angemeldet.</i>
Ist – Reaktion	<i>Der Benutzer wurde angemeldet und die eigenen Testdaten wurden dem Webservice korrekt und vollständig übermittelt.</i>
Ergebnis	<i>Test erfolgreich</i>

15.3 Zusammenfassung

Das Anmelden eines Benutzers wurde erfolgreich durchgeführt. Während der Registrierung hat die Überwachung von LogCat und Wireshark gezeigt dass die zuvor auf dem Client eingegebenen Daten denen entsprechen die hinterher auf dem Webservice ankamen und bei der Übertragung keinerlei Daten verändert wurden.

16 Integration von Client und Webservice - Testlauf 3 (2013 - 07 - 10)

Bei dieser Testdurchführung wird getestet, ob die Daten, die von der einen Komponente korrekt versendet werden bei der anderen Komponente und bei der Übertragung nicht verändert werden.

Art des Tests: Integrationstest

Ausgeführte Testfälle: $\langle T900 \rangle$

Beteiligte Tester: Andreas Bauerfeld, Julian Troegel

Abgedeckte Funktionen: $\langle F30 \rangle + \langle F110 \rangle, \langle F80 \rangle + \langle F100 \rangle, \langle F90 \rangle$

16.1 Testumgebung

Dieser Testfall wird durch Logcat von Eclipse auf Clientseite und Wireshark auf Serverseite ausgeführt

16.2 Testprotokoll

Testfall	$\langle T900 \rangle$
Tester	<i>Andreas Bauerfeld, Julian Troegel</i>
Eingaben	<i>Auf einem Android-Mobilgerät wird X-Map gestartet, der Benutzer ist angemeldet und hat Daten gemessen und lokal gespeichert. Es wird eine Übertragung zum Webservice gestartet. Es beginnt die Datenübertragung.</i>
Soll - Reaktion	<i>Die auf LogCat und Wireshark gleichzeitig schrittweise sichtbaren Nachrichten sind dieselben. Sie enthalten die gleichen Daten.</i>
Ist – Reaktion	<i>Das LogCat und Wireshark zeigen die gleichen Nachrichten. Die Daten wurden dem Webservice korrekt und vollständig übermittelt.</i>
Ergebnis	<i>Test erfolgreich</i>

16.3 Zusammenfassung

Die Überwachung von LogCat und Wireshark zeigen, dass die zuvor auf dem Client eingegebenen Daten denen entsprechen, die hinterher auf dem Webservice ankamen und bei der Übertragung keinerlei Daten verändert wurden. Durch die Visualisierung der vom Client gesendeten Testdaten konnte man auf der Applikation erkennen dass es sich um die zuvor gesendeten Daten handelt.