HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

FHO Fachhochschule Ostschweiz

User Experience

Studienarbeit

Abteilung Informatik
Hochschule für Technik Rapperswil

Herbstsemester 2015

Autoren:	Dominic Peisker und Matthias Fehr
Betreuer:	Prof. Dr. Peter Heinzmann und Eric Franke
Projektpartner:	cnlab Rapperswil

Danksagung

Zu Beginn wollen wir unseren Betreuern und Experten, Peter Heinzmann und Eric Franke von cnlab AG für die gute Zusammenarbeit danken.

Inhaltsverzeichnis

1 Aufgabenstellung

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL
FHO Fachhochschule Ostschweiz

Studienarbeit
Internet Performance Benchmarking und User Experience

Studiengang: Informatik (I)
Institut: ITA: Internet-Techn. und Anwendungen
Gruppe: Matthias Fehr, Dominic Peisker
Betreuer: Heinzmann, Peter
Industriepartner: cnlab

Ausgangslage

Anbieter von kabelgebundenen und mobilen Internet-Diensten werben mit immer schnelleren Anschlussdatenraten. Die sogenannte «User Experience» bei der Internet Nutzung hängt aber nicht nur von Datenraten ab. Antwortzeiten und Paketverlustwerte, aber auch anwendungsbezogene Faktoren wie beispielsweise die Nutzung von Content Delivery Networks, Server-/Client Optimierungen oder neue Protokolle (z.B. SPDY bzw. HTTP/2.0) spielen eine ebenso wichtige Rolle. Ferner ist zu beachten, dass die Anforderungen für verschiedenen Nutzungsarten (Web, File Sharing, VoIP, Streaming) unterschiedlich sind.

Ziel

Es soll ein System zur Visualisierung der verschiedenen Einflüsse zu Verfügung stehen, mit welchem Anwendungen unter verschiedenen Bedingungen durch Testpersonen getestet werden können. Neben der Steuerung von Datenraten, Antwortzeiten, Paketverlustwerten sollen auch die Einflüsse auf Anwendungsstufe aufgezeigt werden können.

Das System soll die Erfassung von Anwendungsparametern und Benutzerreaktionen unterstützen. Testpersonen sollen ausgewählte Anwendungen bei verschiedenen Netzwerk- und Anwendungs-Qualitäten testen und ihre subjektive Einschätzung dazu abgeben können. Daraus sollen sich Aussagen über die Nutzbarkeit verschiedener Anwendung in unterschiedlichen Situationen (z.B. 2G, 3G und 4G, Wireline/Mobile Verbindungen) ableiten lassen.

Das zu realisierende System umfasst folgende Komponenten:

1. Einen Apposite NetEmulator-zur Simulation verschiedener Netzbedingungen

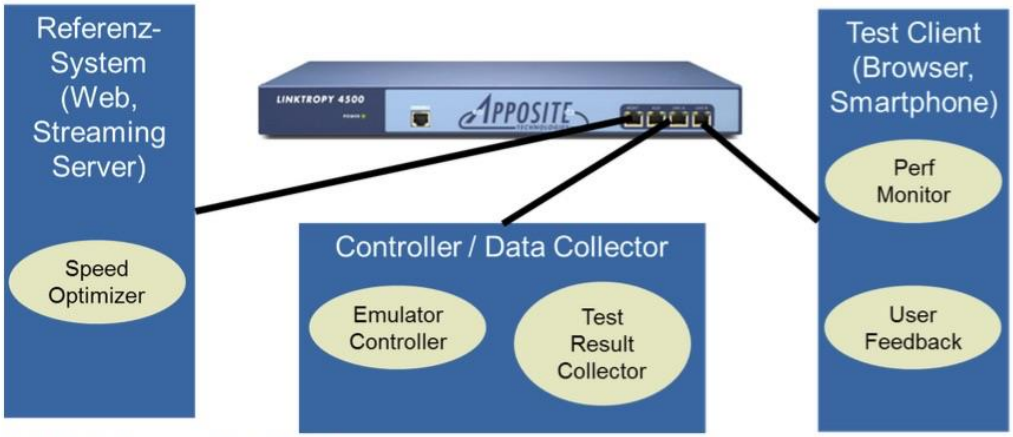


Abbildung 1 Test Setup

2. Ein Referenzsystem als Anwendungsserver (Web und Streaming Server) inkl. Anwendungsoptimierungen zur Bestimmung des Einflusses von verschiedenen Server-/Client Lösungen
3. Test Clients (Browser, Smartphone) mit Performance Monitor Software und User Feedback Möglichkeiten
4. Einen Controller / Data Collector
    a. Testunterstützungsteil zur Steuerung der Tests (Emulator Controller)
    b. Datacollector zur Erfassung der Testdaten inkl. Randdaten zum Test (z.B. Datum, Zeit des Tests; getestete App; Angaben zum Tester wie Name, Vorname, Alter, Geschlecht, Computernutzungstyp) sowie von den Einschätzungen der Probanden.

Teilaufgaben

- Analyse
    - Einarbeitung Mobile User Experience Untersuchungen
    - Einarbeitung Web Performance
    - Analyse des Einflusses verschiedener Faktoren beim Aufruf von Webseiten und bei der Nutzung von Streaming Diensten mit Browser und Mobile Apps
    - Studium des NetEmulators (Apposite)
    - Bestimmung der wichtigsten Performance-Faktoren für verschiedene App-Typen
- Entwicklung Steuerungs-, Monitoring und Feedbackerfassungssystem
    - Anwendung zur Steuerung des Emulators
    - Anwendung zur Erfassung der Qualitätsfaktoren beim Client
    - Anwendung zur Eingabe der Qualitätsbeurteilung (Feedback) durch die User
    - Anwendung zur Erfassung aller Daten auf zentralem System
- Testeinsatz an einem öffentlichen Ort (HSR, Mobile Shop)
    - Zum Abschluss der Arbeit soll eine Testkampagne mit mindestens 10 Probanden durchgeführt werden.

Je nach Vorkenntnissen und Stand paralleler Arbeiten wird das Schwergewicht auf unterschiedliche Teile des Setups gelegt.

Referenzen

1. Hinweise zur Durchführung von Studienarbeiten von P. Heinzmann
2. Ilya Grigorik, Google, „High Performance Browser Networking", O'Reilly, 2013. http://chimera.labs.oreilly.com/books/1230000000545
3. Yelena Nakhimovsky et al, "Mobile User Experience Research: Challenges, Methods & Tools", CHI Workshops, Boston, 2009.
4. Telecom/Networking Testing, Netem, http://testing-in-telecom.blogspot.ch/p/netem.html
5. Apposite Netemulator, http://apposite-tech.com/blog/2012/08/09/a-short-pre-history-of-apposite-technologies/
6. User Experience Erfassungssysteme
    a. http://www.neotys.fr/product/mobile-load-testing.html
    b. Efficient Measurement of the User Experience of Interactive Products. How to use the User Experience Questionnaire (UEQ). Maria Rauschenberger et al, International Journal of Artificial Intelligence and Interactive Multimedia, Vol. 2, Nº 1. http://documat.unirioja.es/descarga/articulo/4221411.pdf
    c. User Experience Blog http://www.user-experience-blog.de/archives/2011/10/ueq---user-experience-quantita.html
    d. Asking the right user experience questions http://www.measuringu.com/blog/ux-questions.php
7. Yusuf Ozuysal, Chief Tab Customizer, Chrome custom tabs smooth the transition between apps and the web, Chromium blog, 2. 9. 2015, http://android-developers.blogspot.ch/2015/09/chrome-custom-tabs-smooth-transition.html, zuletzt besucht am 29.9.2015
8. Android Developers, Up and running with material design, http://developer.android.com/design/index.html, zuletzt besucht am 29.9.2015

15.12.2015

2 Abstract

Die Ladezeit von Webseiten (Page Load Time, PLT) ist ein Qualitätsfaktor, welcher die User Experience, zu Deutsch das Nutzererlebnis beim Surfen im Internet beschreibt. Ladezeiten von wenigen Sekunden gelten als akzeptabel. Steigen die Ladezeiten auf 10 und mehr Sekunden, so sinkt die Zufriedenheit der Internet Nutzer merklich. Die Firma cnlab AG hat ein System zur Erfassung der Page Load Time (PLT) von Webseiten entwickelt, wobei bisher vor allem die Situation beim Einsatz von Desktop Rechnern und Notebooks untersucht wurde. Im Rahmen dieser Studienarbeit ging es darum, die PLT-Untersuchungen auf Smartphones auszudehnen. Neben statistischen Untersuchungen zur PLT bei verschiedenen Webangeboten, sollte auch eine Anwendung entwickelt werden, mit welcher die PLT-Messwerte mit dem subjektiven Empfinden der Nutzer verglichen werden können.

In der Einarbeitungsphase wurde der Einfluss von Datenrate, Round Trip Time (RTT) und Packet Loss auf die PLT bei Smartphones anhand von **manuellen Messungen mit dem Chrome Browser** analysiert. Die Messungen beim Aufruf von neun beliebten Webseiten haben besonders bei Datenraten < 2Mbit/s für die verschiedenen Webseiten grosse Unterschiede der PLT aufgezeigt. Ebenfalls grosse Unterschiede wurden bei RTT > 150ms beobachtet. Bei den schnellen Verbindungen waren die Unterschiede zwischen den einzelnen Webseiten meist kleiner als zwei Sekunden. Die Messungen zum Packet Loss ergaben erst bei unrealistisch hohen Werten (> 5%) signifikante Unterschiede. Bei den weiteren Untersuchungen wurde daher auf die Untersuchung von Packet Loss Einflüssen verzichtet.

In einer zweiten Phase wurde eine **Android Anwendung mit WebView** für automatisierte Page Load Time Messungen entwickelt. Damit liessen sich umfangreichere Messreihen durchführen, welche die Ergebnisse aus den manuellen Tests weitgehend bestätigten.

Schliesslich wurde die **User Experience Testing App** entwickelt, mit welcher Nutzer bei kurzen Tests auf Webseiten zugreifen. Während den Tests werden Daten zum Surferlebnis (URL, Anzahl Elemente, Ladezeiten) aufgezeichnet. Nach Abschluss der Tests geben die Nutzer über die App an, wie gut ihr Surferlebnis war. Die gewonnenen Daten und Nutzerfeedbacks speichert die App zusammen mit Angaben zum Nutzer in einer Datenbank. Die Daten kann die App entweder direkt anzeigen oder sie können exportiert werden. In einem nächsten Schritt wird nun cnlab Testläufe in Mobile-Shops durchführen, um abklären zu können, wie sich die PLT auf das Kundenerlebnis von Smartphone Kunden auswirkt.

3 Management Summary

3.1 Ausgangslage

Anbieter von kabelgebundenen und mobilen Internet-Diensten werben mit immer schnelleren Anschlussdatenraten. Die sogenannte User Experience, also die Kundenzufriedenheit, hängt bei der Internet Nutzung aber nicht nur von Datenraten ab. Antwortzeiten und Paketverlustwerte haben ebenfalls einen Einfluss. Das Unternehmen cnlab AG hat bereits ein System zur Messung der Page Load Time entwickelt. Die Page Load Time ist die Zeit, welche benötigt wird, bis eine Internetseite vollständig geladen ist. Da die Page Load Time die Geschwindigkeit des Surfens angibt, gilt diese als Hauptfaktor für die Benutzererfahrung. Dieselben Messungen sollen nun auch über Smartphones gemacht werden können.

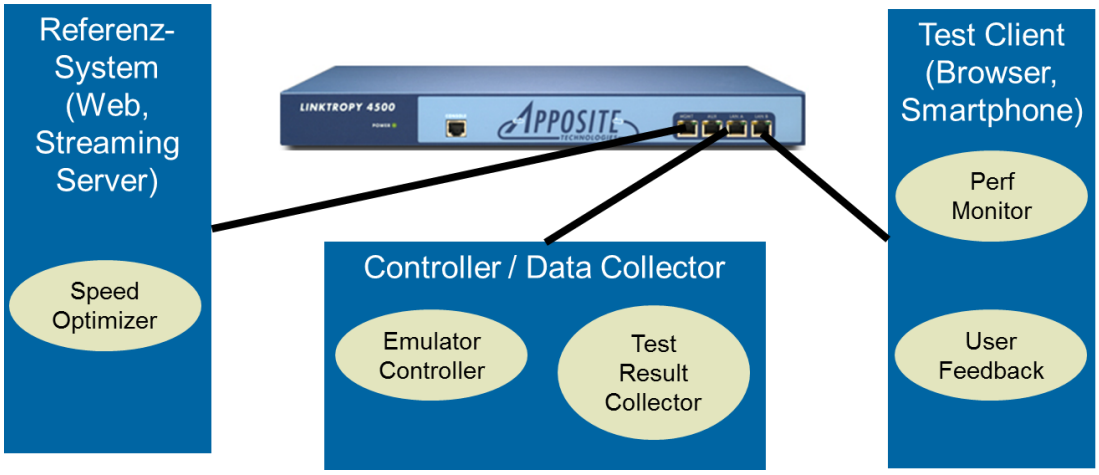
3.2 Zielsetzung

Es soll ein System zur Visualisierung der verschiedenen Einflüsse zu Verfügung stehen, mit welchem Anwendungen unter verschiedenen Bedingungen durch Testpersonen getestet werden können. Neben der Steuerung von Datenraten, Antwortzeiten und Paketverlustwerten sollen auch die Einflüsse auf Anwendungsstufe aufgezeigt werden können.

Das System soll die Erfassung von Anwendungsparametern und Benutzerreaktionen unterstützen. Testpersonen sollen ausgewählte Anwendungen bei verschiedenen Netzwerk- und Anwendungs-Qualitäten testen und ihre subjektive Einschätzung dazu abgeben können. Daraus sollen sich Aussagen über die Nutzbarkeit verschiedener Anwendung in unterschiedlichen Situationen (z.B. 2G, 3G und 4G, Wireline/Mobile Verbindungen) ableiten lassen.

Das zu realisierende System umfasst folgende Komponenten:

1. Einen Apposite NetEmulator-zur Simulation verschiedener Netzbedingungen



2. Ein Referenzsystem als Anwendungsserver (Web und Streaming Server) inkl. Anwendungsoptimierungen zur Bestimmung des Einflusses von verschiedenen Server-/Client Lösungen
3. Test Clients (Browser, Smartphone) mit Performance Monitor Software und User Feedback Möglichkeiten
4. Einen Controller / Data Collector
    a. Testunterstützungsteil zur Steuerung der Tests (Emulator Controller)
    b. Datacollector zur Erfassung der Testdaten inkl. Randdaten zum Test (z.B. Datum, Zeit des Tests; getestete App; Angaben zum Tester wie Name, Vorname, Alter, Geschlecht, Computernutzungstyp) sowie von den Einschätzungen der Probanden.

3.3 Vorgehen

Als erstes werden die Themen Mobile User Experience und Web Performance genauer analysiert. Dazu sollen zum einen die Referenzen in der Aufgabenstellung studiert und zum anderen selber nach weiterführenden Informationen gesucht werden. Der in den später durchzuführenden Messungen verwendete Apposite Netzwerkemulator soll in Betrieb genommen werden. Es sollen zudem die verschiedenen Faktoren analysiert werden, welche die User Experience beim Aufruf einer Webseite beeinflussen können.

In einer zweiten Phase werden diverse Messungen gemacht. Zuerst werden manuelle Messungen über den Chrome Browser des Smartphones gemacht. Dabei gilt es herauszufinden, wie sich verschiedene Einflussfaktoren auf die Qualität der Verbindung auswirken. Diese Messungen sollen analog der bereits von cnlab AG für Desktop PC's durchgeführten Messungen ausgeführt werden. Anschliessend sollen diese Messungen automatisiert auf einem Smartphone erfasst werden. Dazu wird ein kleines Programm entwickelt, welches die zu testenden Seiten automatisiert aufruft und die Page Load Time lokal speichert. Das Programm verwendet die Android WebView.

In der Entwicklungsphase sollen gemäss Aufgabenstellung folgende Anwendungen entwickelt werden.

1. Eine Anwendung zur Steuerung des Apposite Netzemulators. Über diese Anwendung soll es möglich sein, den Apposite Netzemulator zu konfigurieren, ohne dass man als Nutzer das Webinterface aufrufen muss.
2. Eine Anwendung zur Erfassung der Qualitätsfaktoren beim Client. In dieser Anwendung sollen die Nutzer angeben können, was Ihnen beim Surfen im Internet wichtig ist. Diese Angaben helfen dabei, Angebote im Bereich Internet besser an die Kundenwünsche anzupassen.
3. Eine Anwendung zum Testen verschiedener Einflussfaktoren auf die User Experience. Diese Anwendung soll es Nutzern erlauben, verschiedene Einflussfaktoren, welche beim Surfen mit dem Smartphone auftreten, zu testen. Dies kann zum Beispiel dazu verwendet werden, den Nutzer beim Kauf eines Smartphone Abos zu unterstützen, damit er das für ihn optimale Abo findet.
4. Eine Anwendung zur Eingabe der Qualitätsbeurteilung (Feedback) durch die Nutzer. Hier sollen die Nutzer ihr Feedback zu den in Anwendung 3 durchgeführten Tests geben können.
5. Eine Anwendung zur Erfassung aller Daten auf einem zentralen System. Diese Anwendung soll alle relevanten Daten der vorhergehenden Anwendungen managen und deren Speicherung, zum Beispiel in einer Datenbank, regelt.

In der Testphase sollen die entwickelten Anwendungen produktiv getestet werden. Als Beispiel für mögliche Test Orte werden die HSR oder ein Mobile Shop angegeben.

3.4 Resultat

Die geplanten Messungen wurden durchgeführt. Die Erkenntnisse aus den Messungen wurden dokumentiert und auch grafisch dargestellt. Neben den geforderten manuellen und automatisierten Messungen wurden noch Messungen zu den Unterschieden beim Surfen mit Desktop PC und Smartphones gemacht. Auch wurde das Prinzip des Preloadings, welches bei der Analysephase als relevantes Konzept entdeckt wurde, näher beschrieben.
Entwickelt wurde aus diesen Erkenntnissen eine App. Die App erlaubt es, auf definierten Testseiten zu Surfen. Während dem Surfen werden alle als relevant eingestuften Daten gespeichert. Zudem können Angaben zum Nutzer erfasst werden. Am Ende des Tests gibt der Nutzer ein Feedback. Alle gespeicherten Daten werden auf dem Smartphone lokal in einer SQLite Datenbank gespeichert. Diese Datenbank kann auch exportiert werden.

3.5 Ausblick

In weiterführenden Arbeiten könnte ein System zur automatisierten Konfigurierung des Apposite Netzemulators entwickelt werden. Dies würde die Verwendung der App zum Beispiel als Kaufberatung in Mobile Shops ermöglichen.

Weiter könnte die Speicherung der Daten neu nicht mehr lokal auf dem Smartphone, sondern auf einer zentralen Datenbank erfolgen. Durch das exportieren der Datenbank ist es bereits jetzt möglich, Daten von mehreren Geräten zusammenzutragen. Dies würde diese Aufgabe jedoch stark vereinfachen und automatisieren.

Die Feedbackmöglichkeiten der Nutzer sind aktuell noch auf das Nötigste beschränkt. Auch hier könnten bei künftigen Arbeiten Optimierungen vorgenommen werden. Im gleichen Schritt könnte auch überprüft werden, ob die nutzerspezifischen Angaben ausreichen.

Und last but not least kann die App schliesslich für Tests der User Experience genutzt werden.

4 Einleitung

4.1 Ausgangslage

Internetprovider wie Swisscom oder Cablecom werben mit immer schnelleren Datenraten um neue Kunden. Doch bringen immer grössere Datenraten auch ein besseres Nutzererlebnis beim Surfen? Die Firma cnlab AG mit Sitz in Rapperswil hat zur Klärung dieser Frage ein System entwickelt, welches die Page Load Time (PLT) misst und erfasst. Die PLT ist die Zeit, welche benötigt wird, bis eine im Internet aufgerufene Seite vom Browser komplett geladen wurde. Das von cnlab entwickelte System läuft auf dem Firefox Browser und verwendet die Browsererweiterungen (Plug Ins) Firebug [1] und Selenium [2]. Das System ist mittlerweile soweit ausgereift, dass Messungen über einen herkömmlichen Kabelanschluss wie DSL, Cablenet oder über einen Satellitenanschluss möglich sind. Über ein Zusatzgerät, Mobile USB Adapter genannt, welches den Anschluss eines Desktop PC's an das Mobilfunknetz ermöglicht, sind sogar Messungen über das Mobilfunknetz möglich.

Im Rahmen dieser Arbeit werden dieselben Messungen nun auf einem Smartphone nachgestellt. Dazu wird die Chrome App [3] auf einem Android Smartphone verwenden und damit die verschiedenen Websites aufgerufen. Die Zeit, bis die Seite vollständig geladen ist, wird mit dem Chrome Remote Debugging Tool [4] bestimmt. Dieses Tool erlaubt es, ein Android Smartphone via USB an einen Desktop-PC anzuschliessen und über den Chrome Browser des Desktop-PC die Verbindung des Smartphones zu analysieren. Als Testgerät dient ein Huawei Ascend Mate 7 [5] mit Android. Als WLAN Access Point wird ein Netgear N150 Wireless Router [6] verwendet.

Um verschiedene Netzwerksituationen zu simulieren, wird ein Apposite Netzemulator [7] zwischen den WLAN Access Point und dem Anschluss an das HSR Netzwerks geschaltet. Der Apposite Netzemulator ermöglicht die manuelle Einstellung diverser Parameter für jede Verbindungsrichtung wie z.B. Datenrate (Bandwidth), Verzögerungszeit (Delay) sowie Packet-Loss (siehe Abbildung 1).

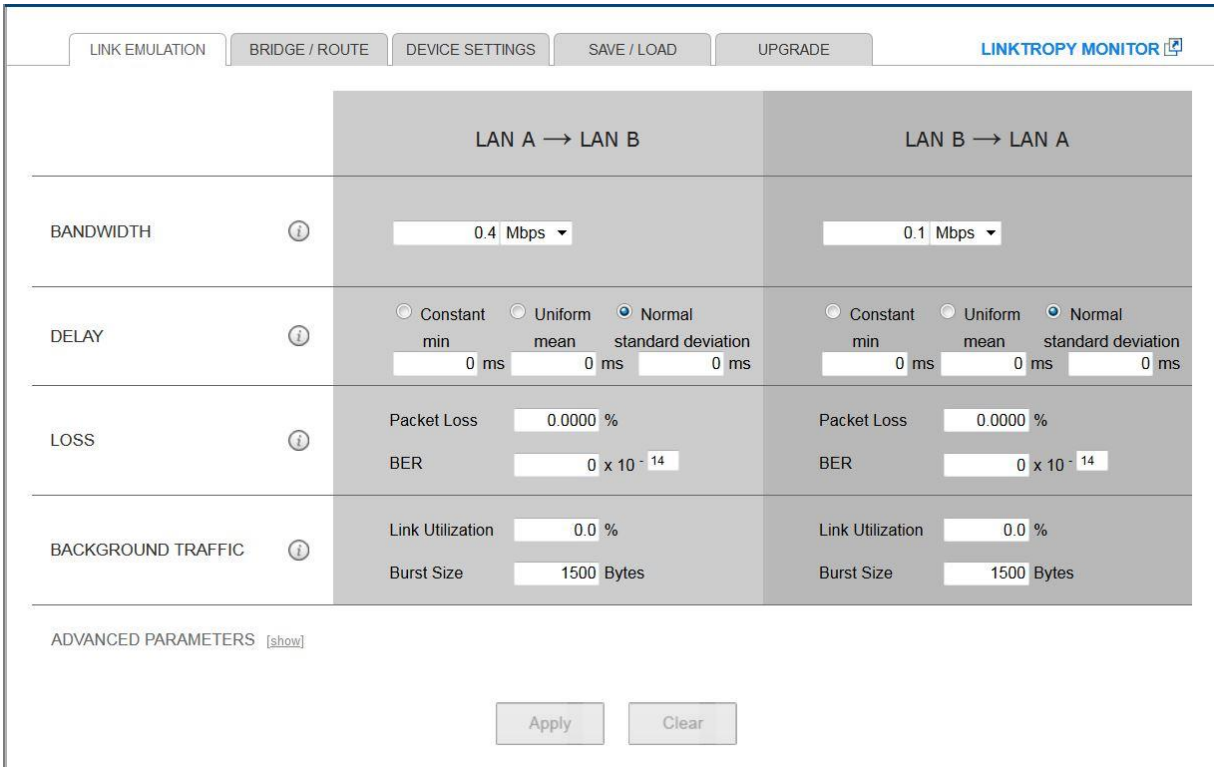


Abbildung 1, Screenshot der Apposite Netzwerkemulator-Benutzeroberfläche

Die Auswahl der zu simulierenden Internet-Datenraten erfolgt entsprechend den aktuell verfügbaren Mobiltechnologien und den Angeboten mit Datenratenlimitierung in der Schweiz. Gegenwärtig gibt es bei Swisscom Abos mit unterschiedlichen maximalen Datenraten [8] und bei Sunrise Beschränkung auf das 3G-Netz bei Überschreitung des monatlichen Datenvolumens. Die zu simulierenden Antwortzeiten entsprechen den bei verschiedenen Verbindungstechnologien typischen Verzögerungszeiten (Cablenet 15ms, DSL 25ms, EDGE 200ms, 3G 40ms, 4G 40ms). Falls dies aufgrund des beobachteten Verlaufs der Messwerte sinnvoll erscheint, werden weitere Messpunkte aufgenommen bzw. Datenraten und Verzögerungszeiten getestet.

Um möglichst aussagekräftige Resultate zu erhalten, wurden für jede Kondition die Testwebseiten fünf Mal aufgerufen und aus den Ergebnissen der Mittelwert gebildet. Falls die Zeitmessung einen Wert weit ausserhalb des Durchschnitts ergab, wurde die Messung jeweils wiederholt, um die Messung nicht aufgrund einer seltenen Ausnahme zu verfälschen.

4.2 Ziel

Das Ziel dieser Messungen ist es, die verschiedenen Einflussfaktoren und deren Auswirkungen auf die Page-Load-Time (PLT) zu analysieren. Auch soll anhand der Resultate abgeschätzt werden können, welche Auswirkungen eine Veränderung der Netzparameter auf die User Experience hat. Diese Informationen könnten für Mobilfunkprovider interessant sein, da insbesondere bei den tiefen Bandbreiten eine kleine Erhöhung bereits eine starke Verbesserung der PLT mit sich bringen kann.

Bevor mit der Entwicklung der App begonnen wird, sollen verschiedene Performance-Faktoren anhand manueller Messungen untersucht werden. Basierend auf diesen ersten Untersuchungen der User Experience sollen die Anforderungen an die zu entwickelnde Anwendung präzisiert werden. Des Weiteren ergänzen die Messungen die theoretischen Untersuchungen im Rahmen der Analyse der User Experience. Im Folgenden Abschnitt werden die Messungen und die daraus resultierenden Erkenntnisse erläutert.

5 Analyse der User Experience

5.1 Manuelle Messungen

Bei den Messungen wurde jeweils nur ein Parameter modifiziert. Dies erlaubt die isolierte Analyse der verschiedenen Einflussfaktoren. Die Auswertung kann so in die drei Kategorien Bandbreite, Round-Trip-Time (RTT) und Packet-Loss eingeteilt werden.

5.1.1 Verschiedene Bandbreite

Bei diesem Versuch wurde der Einfluss der Bandbreite auf die PLT gemessen. Das Verhältnis von Up- zu Downloadrate orientiert sich an den Swisscom Infinity [8] Abos. Dabei liess sich erkennen, dass nur bei den tiefen Datenraten bis etwa 5 Mbps ein merklicher Unterschied der PLT festzustellen war. Eine Downloadrate von 0.2Mbps, wie dies das kleinste Abo (Infinity XS) von Swisscom bietet, ist zum Surfen ungeeignet, da das Internet Erlebnis unter den Ladezeiten von teils über einer Minute leidet. Diese Bandbreite ist höchstens für Messaging-Dienste zu empfehlen. Schon eine Geschwindigkeitssteigerung der Downloaddatenrate auf 0.4 Mbps bringt eine deutliche Verbesserung der PLT auf weniger als 45 Sekunden.

Die Downloadraten wurden anfänglich auf die Werte 0.2, 2, 5 und 100 Mbps eingesetzt. Nach den Erkenntnissen aus diesen Messungen wurden weitere Messungen gemacht. Es wurde entschieden, besonders den Bereich zwischen 0.2 und 2 Mbps genauer zu untersuchen. In diesem Bereich hatte sich die PLT am deutlichsten verändert. Auch die Werte 21 und 42 Mbps wurden noch separat getestet, da diese Technologiegrenzen beim 3G-Netz darstellen.

Die Uploaddatenraten wurden entsprechend den Swisscom Infinity Abos [8] gewählt. Diese liegen, ausser bei den ganz geringen Bandbreiten, meist bei 10 – 20% der Downloadrate. Es wurde somit bei jeder Messung sowohl die Downloaddatenrate wie auch die Uploaddatenrate verändert.

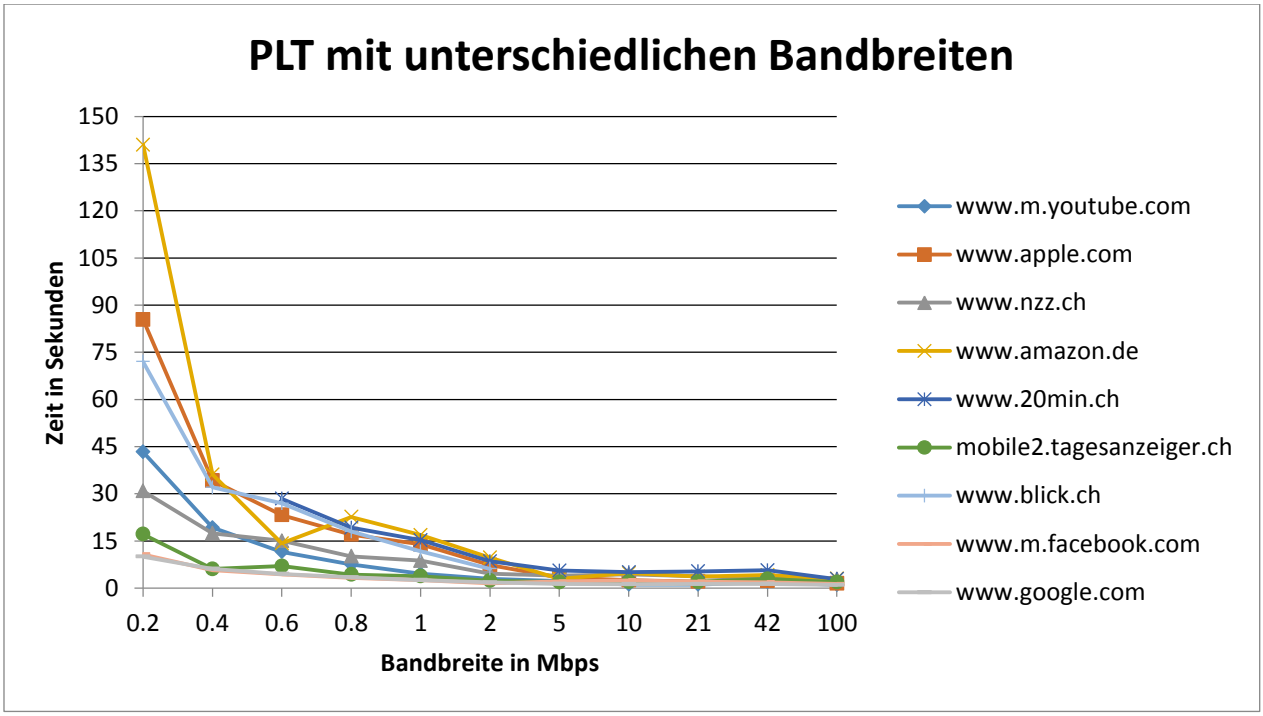


Abbildung 2: PLT Darstellung mit allen getesteten Bandbreiten

Wie in Abbildung 2 ersichtlich, sind oft Seiten mit vielen Medien am stärksten von langsamen Datenraten betroffen. Die Webseite von 20Minuten hatte bei einer Downloadrate unter 0.6Mbps teils nicht vollständig geladen bzw. hatte über drei Minuten benötigt. Die Inhalte waren teilweise dennoch ersichtlich und benutzbar.

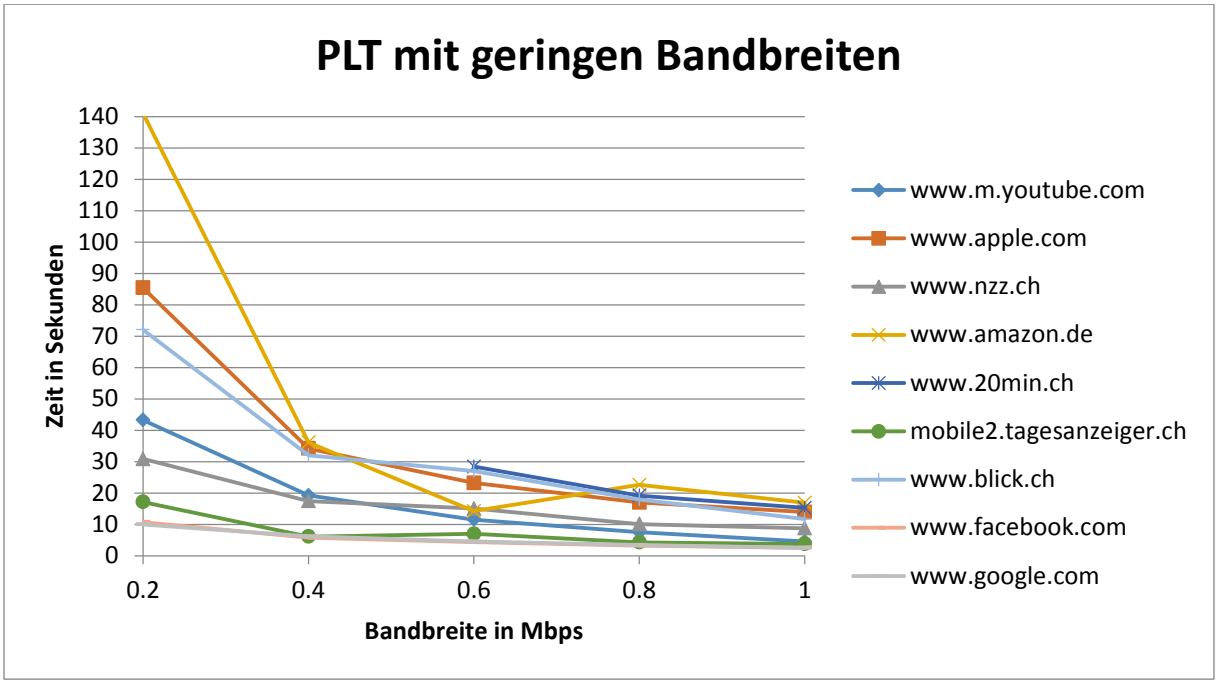


Abbildung 3: PLT mit niederen Bandbreiten

Den Effekt der geringen Bandbreiten ist besonders in Abbildung 3 gut ersichtlich. Eine Verdoppelung der Bandbreite von 0.2 auf 0.4 Mbps bringt bei den meisten Seiten deutlich mehr als eine Halbierung der PLT.

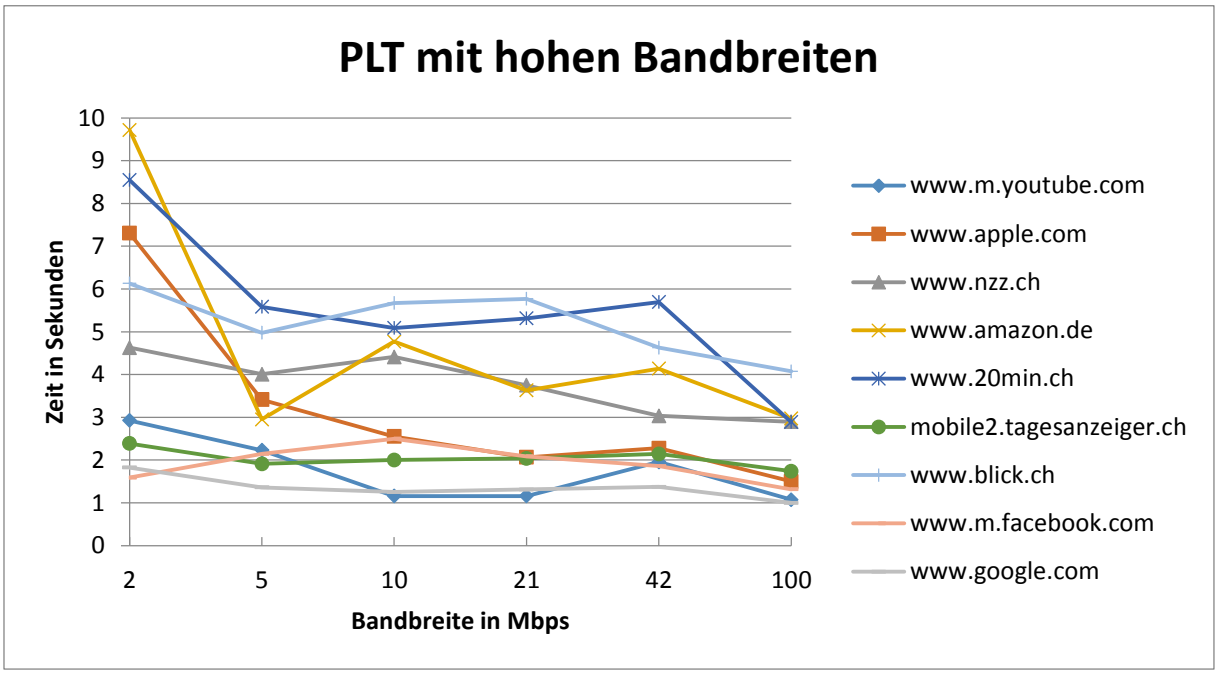


Abbildung 4: PLT Auflistung der hohen Bandbreiten

Bei den hohen Bandbreiten hingegen zeigt sich der gegenteilige Effekt verglichen mit den geringen Bandbreiten, wie Abbildung 4 zeigt. Die PLT sinkt zwischen 2 und 5 Mbps nochmals deutlich, bleibt danach jedoch ziemlich stabil. Aus Kundensicht lohnt sich hier definitiv die Überlegung, ob sich die hohen Kosten für mehr Bandbreite wirklich auszahlen.

5.1.2 Unterschiedliche Round Trip Time

Die Round Trip Time (RTT) beschreibt die Zeit, welche ein Paket braucht, um vom Client zum Server und wieder zurück zu gelangen. Im unserem Falle also die Zeit, welche benötigt wird, bis die Antwort einer Webseite auf dem Smartphone eintrifft. Die RTT steigt, je mehr die Verbindung verzögert wird (zum Beispiel durch die Distanz). Vor allem in der Mobilkommunikation unterscheiden sich die RTT's von den verschiedenen Funktechniken wie GPRS, UMTS und LTE stark, was eine Untersuchung dieses Faktors ebenfalls interessant macht.
Bei der Messung wurden die Pakete um eine vordefinierte Zeit vom Netzwerkemulator verzögert, um so eine erhöhte RTT zu simulieren. Um Festzulegen, welche RTT's getestet werden sollen, wurde auf die Messungen von cnlab zurückgegriffen.

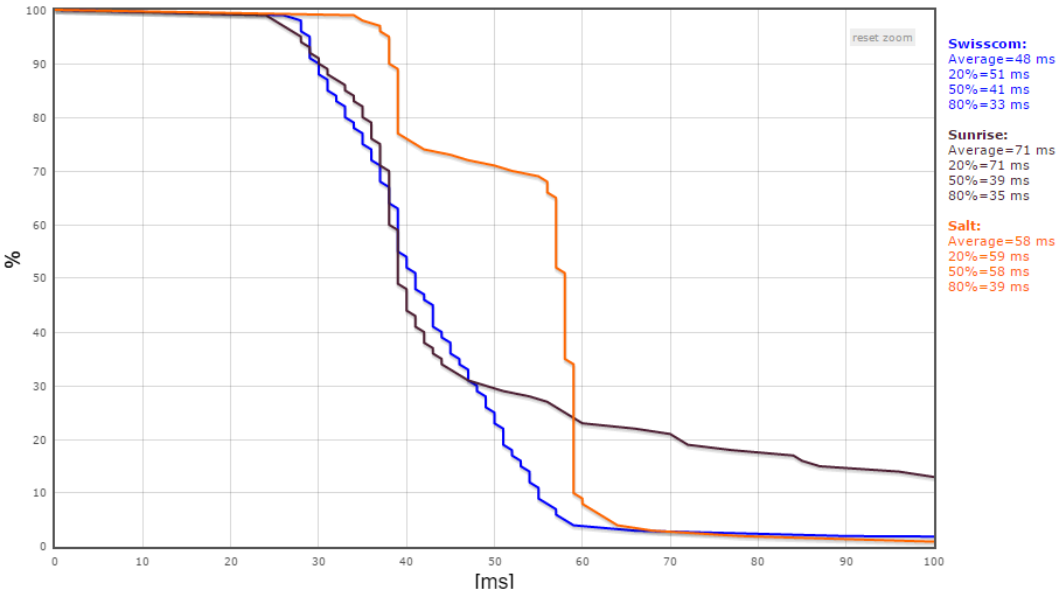


Abbildung 5: RTT's der verschiedenen Anbieter im 4G Netz

In Abbildung 5, welche uns vom cnlab zur Verfügung gestellt wurde, sind die RTT's der verschiedenen Schweizer Mobilfunkprovider im 4G Netz aufgezeigt. Daraus kann abgelesen werden, dass fast 100% der Verbindungsanfragen eine RTT von über 20ms haben. Bei einer RTT von 40ms sind, zumindest bei Swisscom und Sunrise, etwa 50% aller Anfragen bedient. Besonders bei Salt sieht man bei einer RTT von 60ms eine enorme Steigerung der Anzahl Anfragen, welche unter dieser RTT liegen. Im Schnitt haben etwa 90% aller Anfragen eine RTT unter 60ms.
Aus diesem Grund wurde entschieden, mit zusätzlichen Verzögerungen von 0ms, als theoretisches Optimum, 20, 40 und 60ms zu arbeiten. Um die Veränderungen in Extremsituationen zu beobachten, haben wir zusätzlich mit Verzögerungen von 300 und 600ms gemessen. In der Gesamtansicht von Abbildung 6 ist ersichtlich, das Surfen mit einer Verzögerung von 300ms oder mehr, zumindest bei Webseiten mit vielen Inhalten, kein erfreuliches Erlebnis ist. Wichtig zu wissen ist hier, dass nur die Verzögerung zwischen dem Mobiltelefon und der Antenne beeinflusst werden kann. Die anschliessend im Netz anfallende Verzögerung tritt in jedem Falle auf.

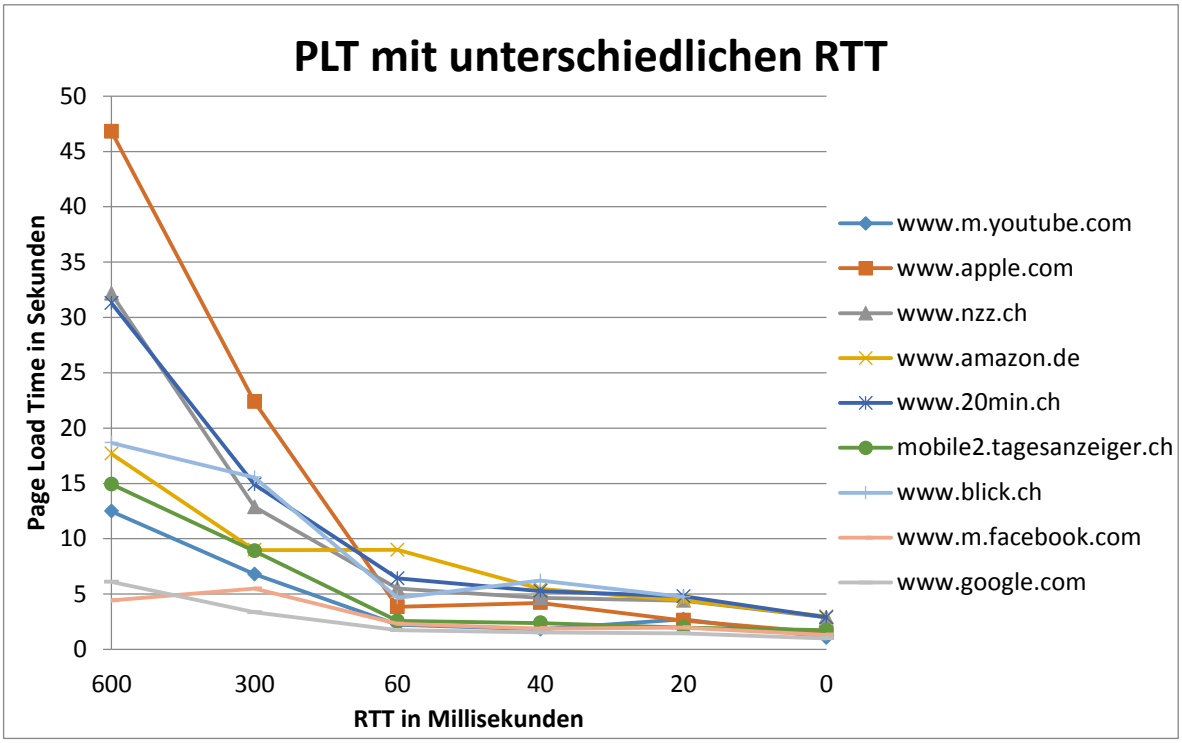


Abbildung 6: PLT mit unterschiedlichen RTT

Im Gegensatz dazu zeigen sich in der Übersicht der geringen Verzögerungen in Abbildung 7, welche gemäss Abbildung 5 im täglichen Gebrauch auftreten, ähnliche Phänomene wie bereits bei den Bandbreiten. Auch hier bei den RTT's wirken sich schlechte Werte vor allem bei Seiten mit vielen Inhalten negativ auf die PLT aus. Seiten wie google.com, welche praktisch keinen Inhalt besitzen, zeigen im Gegensatz dazu nur sehr geringe Unterschiede.

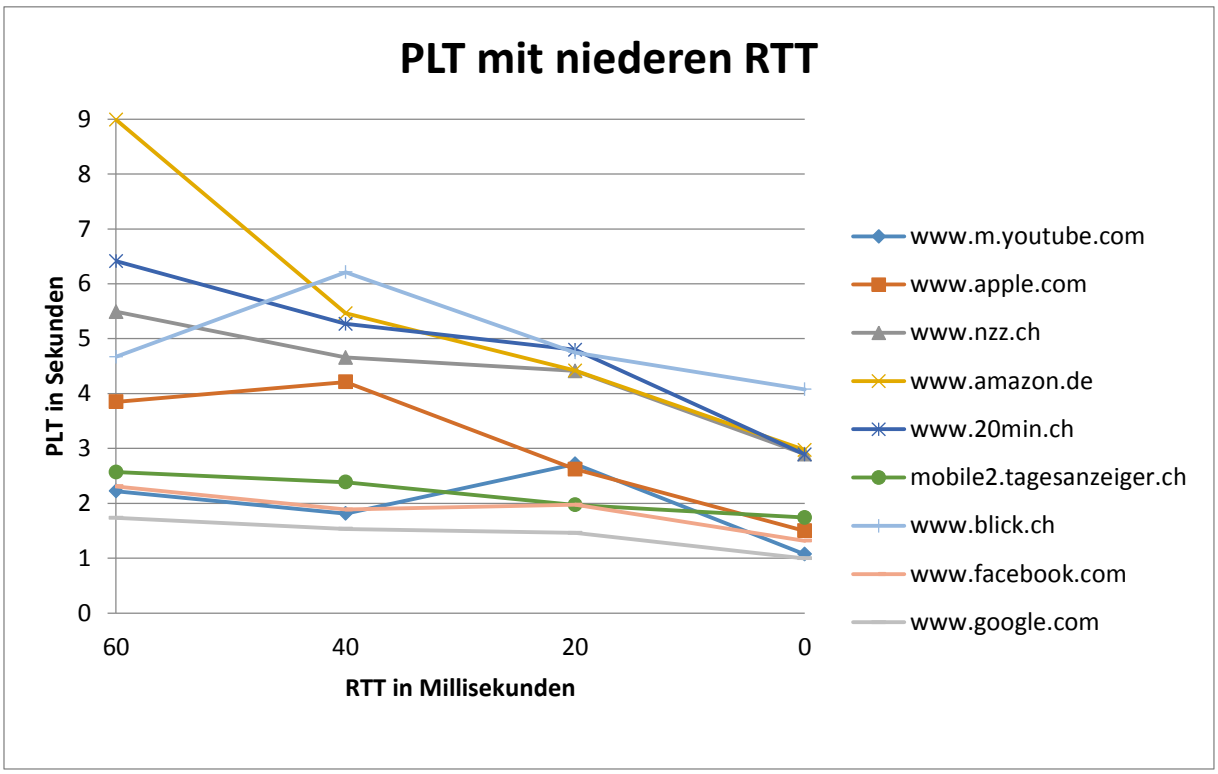


Abbildung 7: PLT mit niederen RTT

5.1.3 Variierender Packet-Loss

Packet-Loss bezeichnet das Phänomen, dass Datenpakete im Netz verloren gehen. Die Endpunkte der Verbindung bemerken, wenn ein Paket fehlt, und fordern das verlorene Paket erneut an. Dadurch entstehen Verzögerungen, welche sich negativ auf die PLT auswirken.

Die Versuche mit dem Packet-Loss sind mit Vorsicht zu geniessen. Die Ergebnisse wichen teils stark von den Erwartungen ab. Zudem ist der Packet-Loss kein Parameter, der sich einfach verbessern liesse. Auch ist die Ursache, wieso ein Paket verloren ging, nur schwer auszumachen. Zudem ist der Packet-Loss unter realen Bedingungen so klein, dass er nur einen verschwindend geringen Einfluss auf die PLT hat. Daher wurde in Absprache mit den Betreuern entschieden, zukünftig auf Messungen mit Packet-Loss zu verzichten und den Fokus auf Bandbreite und RTT zu legen.

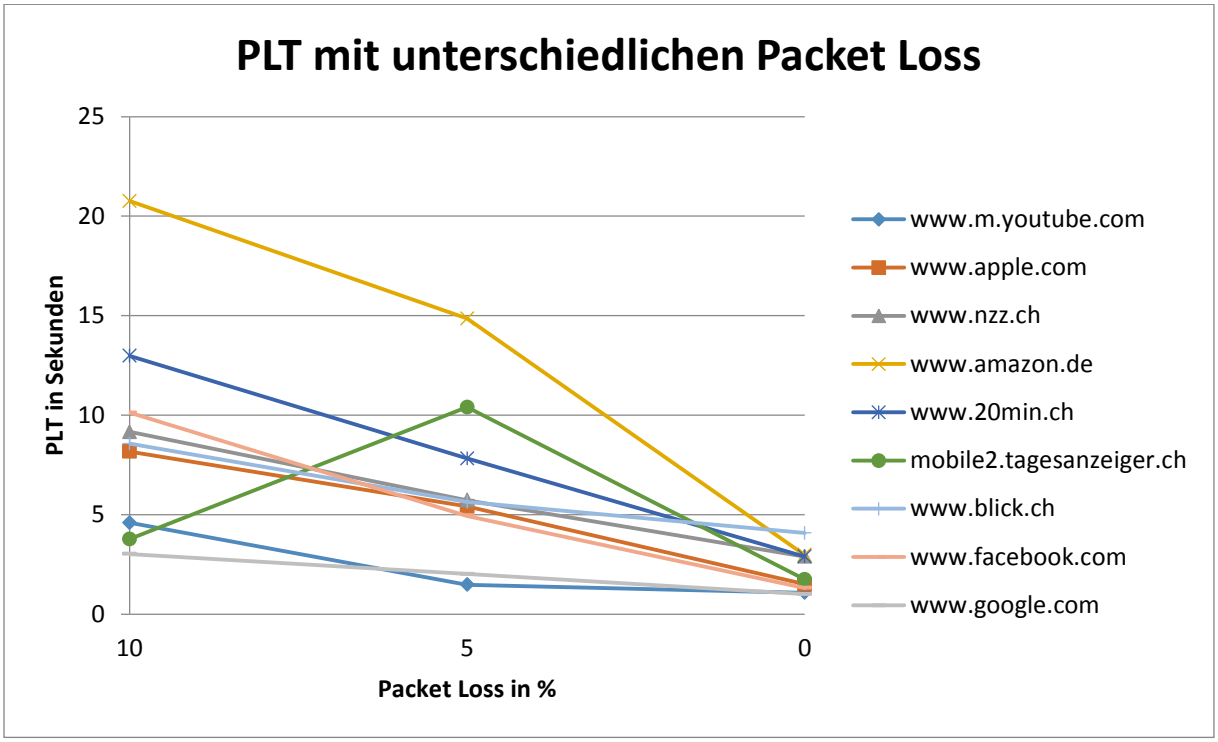


Abbildung 8: PLT mit unterschiedlichen Packet Loss

In Abbildung 8 ist ersichtlich, wie lange das Laden der einzelnen Seiten mit einem gegebenen Packet-Loss dauert. Die Unterschiede zwischen den einzelnen Messdurchgängen waren teils gross. Auch konnte trotz langem Nachforschen nicht erkläret werden, wieso beim Tagesanzeiger die PLT bei einem Packet-Loss von 10% geringer ausfällt als mit einem Packet-Loss von 5%.

5.1.4 Ausblick

Über alle Kategorien hinweg zeigte sich, dass die PLT umso mehr schwankte, je mehr Bilder und Medien von der Webseite geladen werden mussten. Ferner konnte beobachtet werden, wie auf einigen Seiten selbst nach Beendigung des Ladevorgangs (gem. Google Remote Debugging Tool [4]) noch weitere Daten, je nach Seite hauptsächlich Bilder, nachgeladen wurden. Besonders die Seite tagesanzeiger.ch verwendet diesen Nachlademechanismus, aber auch bei anderen Seiten wie amazon.de traten solche Effekte auf. In einem nächsten Schritt wird untersucht, worum es sich dabei handelt.

5.2 Automatisierte Messungen

Als erstes wurde ein Prototyp der App entwickelt. Mit diesem ist es möglich, eine Webseite aufzurufen. Nach Beenden des Ladevorganges wird die benötigte PLT angezeigt.

Dieser Prototyp wurde in einer Sitzung mit den Betreuern besprochen und als Grundlage für die App gutgeheissen. In einem weiteren Schritt wurde der Prototyp dahingehend erweitert, dass automatisierte Messungen durchgeführt werden können.

Nachdem der Prototyp funktionsfähig war, wurden die automatisierten Messungen durchgeführt. Dabei wurden die Messungen wiederholt, welche beim Kapitel 5.1 Manuelle Messungen durchgeführt und beschrieben wurden. Dadurch konnten die manuell, über den Desktop PC durchgeführten Messungen, mit den hier durchgeführten automatisierten Messungen verglichen werden.

Über alle Messungen hinweg wurde eine höhere PLT gemessen als bei den ersten, manuellen Messungen. Grund dafür ist, dass der in der App verwendete Browser, WebView genannt, langsamer ist als der Chrome Browser.

Leider stellte sich heraus, dass die WebView bei vielen Webseitenaufrufen nicht stabil läuft. Bei etwa 50% der Testdurchführungen beendete sich die WebView selbst. Dies führte dazu, dass die automatisierten Tests immer wieder abstürzten und wiederholt werden mussten. Die Tests der besonders tiefen Bandbreiten 0.2 und 0.4 Mbps sowie den RTTs von 300 und 600ms konnten nie erfolgreich beendet werden. Leider konnte der Grund des Fehlers nicht identifizieret werden. Auch in Internetforen wurde keine Lösung des Problems gefunden.

5.2.1 Unterschiedliche Bandbreiten

Wie eingangs erwähnt, wurden wieder dieselben Werte für die Up- und Downloaddatenrate verwendet wie bereits bei den ersten Messungen. Einzig die Tests mit 0.2 und 0.4 Mbps Downloaddatenrate konnten nicht durchgeführt werden.

Wie schon bei den ersten, manuell durchgeführten Messungen, zeigten sich bei den Bandbreiten praktisch nur in den tiefen Bereichen markante Unterschiede in der PLT. Ab einer Downloaddatenrate von 5 Mbps sind fast keine Veränderungen mehr ersichtlich.

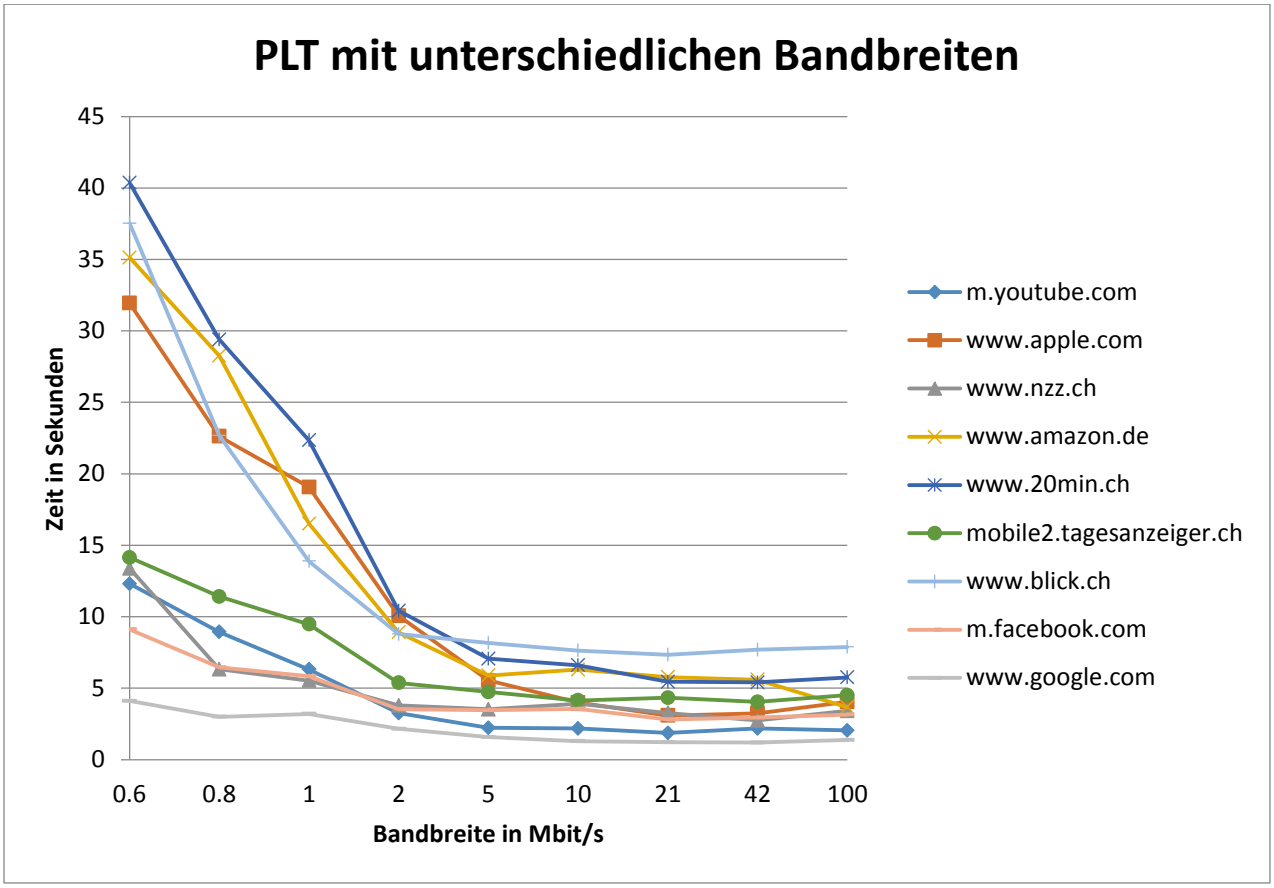


Abbildung 9: PLT mit unterschiedlichen Bandbreiten

5.2.2 Unterschiedliche RTT

Auch bei den Tests mit der RTT wurden dieselben Konfigurationen verwendet wie bereits bei den ersten Messungen.

Wie aus dem Diagramm ersichtlich ist, sind kaum Unterschiede der PLT auszumachen. Mit Ausnahme von Blick und Youtube liegt der Unterschied der PLT einer Webseite unter einer Sekunde.

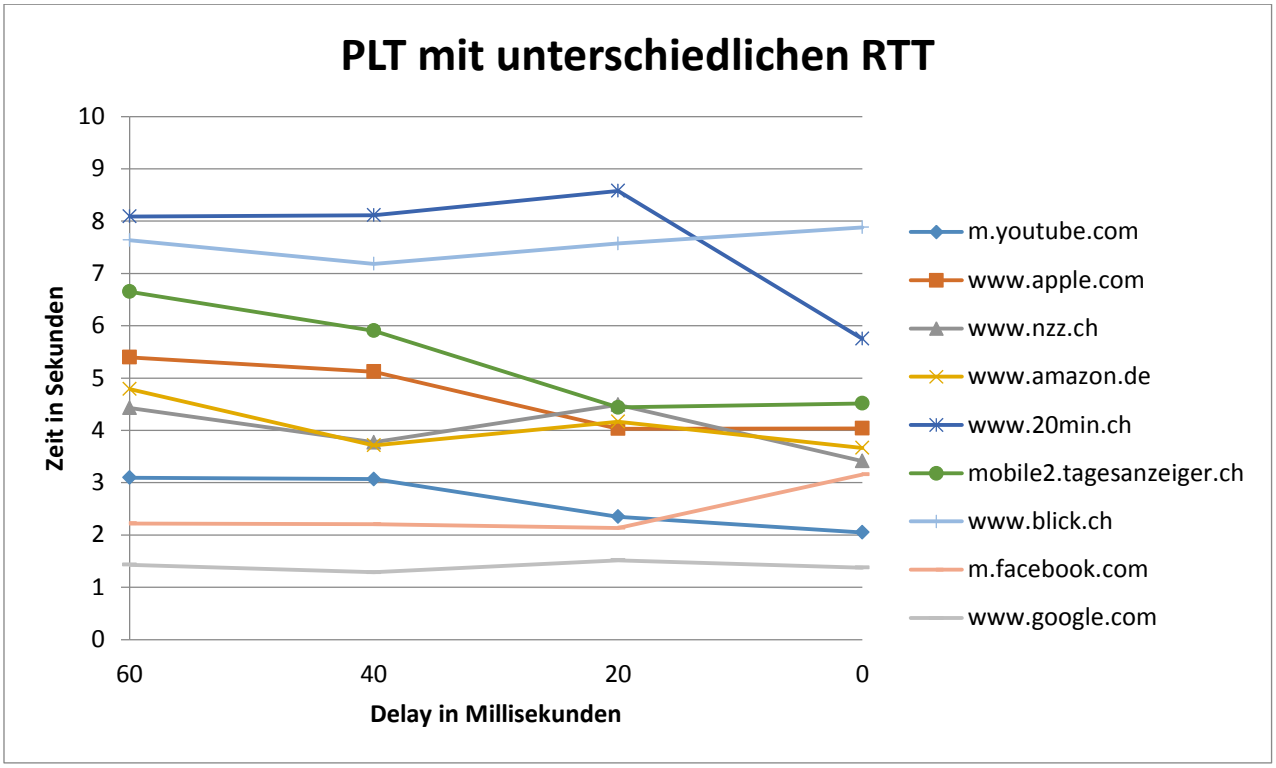


Abbildung 10: PLT mit unterschiedlichen RTT

5.2.3 Erkenntnisse und Ausblick

Die automatisierten Messungen bestätigten die Erkenntnisse aus den manuellen Tests. Auch konnten dank Ihnen die Schwächen der WebView aufgedeckt werden. Da der Ursprung des Fehlers nicht geklärt werden konnte, kann nicht ausgeschlossen werden, dass dieser auch bei Einzeltests auftritt.

5.3 Preloading

Peter Heinzmann hat uns in der Besprechung darauf aufmerksam gemacht, auf ein allfälliges Preloading der einzelnen Seiten zu achten. Was dies ist und welche Erkenntnisse dazu aus den Messungen gezogen wurden, wird in den nächsten beiden Abschnitten beschrieben.

5.3.1 Erläuterung

Preloading, auch Prefetching genannt, bedeutet, dass Webseiten schon im Vorhinein Daten laden, welche der Benutzer gar noch nicht aufgerufen hat. Als Beispiel die Internetseite einer Zeitung, welche auf ihrer Homepage die verschiedenen Artikel aufgelistet hat. Ohne Preloading muss nach Anwählen eines Artikels dieser zuerst vom Server geladen werden. Mit Preloading hingegen werden bereits alle Artikel geladen, noch während der Benutzer auf der Homepage ist. Klickt dieser dann auf einen Artikel, kann dieser sofort angezeigt werden, da er schon vom Server geladen wurde.

Es existieren verschiedene Arten von Preloading. DNS Preloading, Preloading und Prerendering. Beim DNS Preloading sendet die Homepage eine Liste von Seiten, welche der Besucher wahrscheinlich als nächstes besucht, an den Browser. Dieser startet im Hintergrund eine DNS Abfrage für diese Seiten. Auf diese Art werden die Seiten, wenn sie denn tatsächlich aufgerufen werden, schneller geladen. Gewisse Browser bauen beim DNS Preloading auch bereits eine TCP Verbindung auf. Preloading geht bereits etwas weiter. Neben der DNS Abfrage werden einzelne Ressourcen, zum Beispiel Bilder, der nächsten Seite geladen. Klickt der Benutzer nun auf einen weiterführenden Link, bei dem Preloading gemacht wurde, sind diese Ressourcen bereits im Browser zwischengespeichert und die Seite kann schneller angezeigt werden. Noch weiter geht Prerendering. Beim Prerendering wird die ganze Homepage im Voraus geladen. Klickt man nun auf den Link, wird die neue Seite einfach in den Vordergrund geschaltet und ist sofort verfügbar [9] [10].

5.3.2 Erkenntnisse aus den Messungen

Ob beim Tagesanzeiger Preloading gemacht wird, wurde getestet, indem die Homepage geladen und anschliessend der Flugmodus des Smartphones aktiviert wurde. Konnten die Artikel trotz Flugmodus noch angezeigt werden, wurde Preloading gemacht. Dieses Verhalten wurde lediglich bei tagesanzeiger.ch festgestellt, welcher die Artikel jedoch ohne Bilder vorgeladen hatte. Dies erklärt somit auch, weshalb selbst nach Beendigung des Ladevorgangs dieser Webseite noch viele Daten nachgeladen wurden. Dieses Verhalten ist in Abbildung 11 gut ersichtlich.

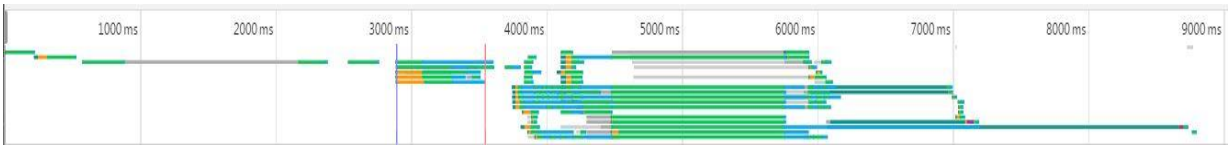


Abbildung 11: Preloading auf tagesanzeiger.ch

Der vertikale, rote Strich zwischen 3000 und 4000ms markiert nach Chrome Developer Tools das Beenden des Ladens der Seite. Es interessierte nun, wie das Preloading bei tagesanzeiger.ch gemacht wird. Dazu wurde wieder mithilfe der Chrome Developer Tools analysiert, was für Daten von tagesanzeiger.ch an das Mobilgerät gesendet werden. Dabei wurde bemerkt, dass die meisten Daten, welche nach dem offiziellen Beenden des Ladevorganges geladen wurden, nicht von der Seite tagesanzeiger.ch stammen. Diese sendet jedoch eine Javascript Datei, welche das nachträgliche Laden der Artikel von anderen Servern steuert.

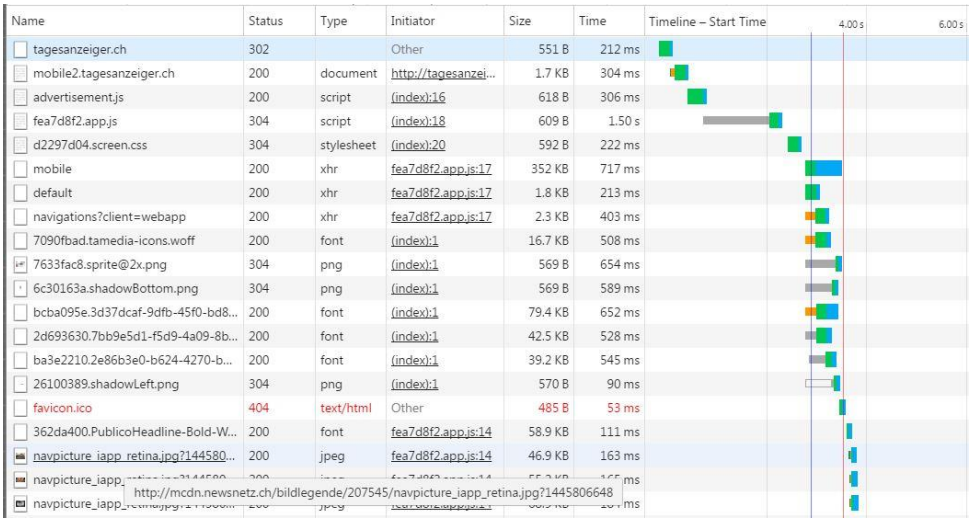


Abbildung 12: Preloading auf tagesanzeiger.ch 2

Abbildung 10 zeigt nochmals den ersten Teil des Ladevorganges von tagesanzeiger.ch auf. Die untersten drei Dateien wurden dabei von der Seite newsnetz.ch geladen.

Bei den Auswertungen der Untersuchungen zeigte sich jedoch, dass das Verhalten der Webseite des Tagesanzeigers nicht die Kriterien des Preloadings erfüllt. Die Artikel werden zwar von einer anderen Seite nachgeladen, jedoch geschieht dies offen über die Javascript Datei.

5.4 Vergleich Desktop-/ Mobile Webseite

Um den Unterschied zwischen der Desktop- und der Mobileversion der untersuchten Webseiten zu ermitteln, wurden die bereits unter Kapitel 5 verwendeten Webseiten getestet. Die Webseiten wurden nacheinander mit Hilfe des Chrome Developer Tools als Mobilgerät und als Laptop aufgerufen. Dabei wurde festgehalten, wie viele Daten empfangen wurden, wie viele Requests der Browser an die Webseite sendete und von wie vielen verschiedenen Servern Daten empfangen wurden. Ebenfalls wurde die PLT notiert. Für die Aufrufe als Mobilgerät wurden im Chrome die drei Smartphones iPhone 6, LG Optimus L70 und das Google Nexus 5 simuliert.

Wichtig zu beachten ist, dass diese Tests bei jeder Seite nur einmal durchgeführt wurden. Das Ziel war es, grob zu sehen, in welchen Bereichen und wie stark die einzelnen Seiten für Mobilegeräte optimiert sind.

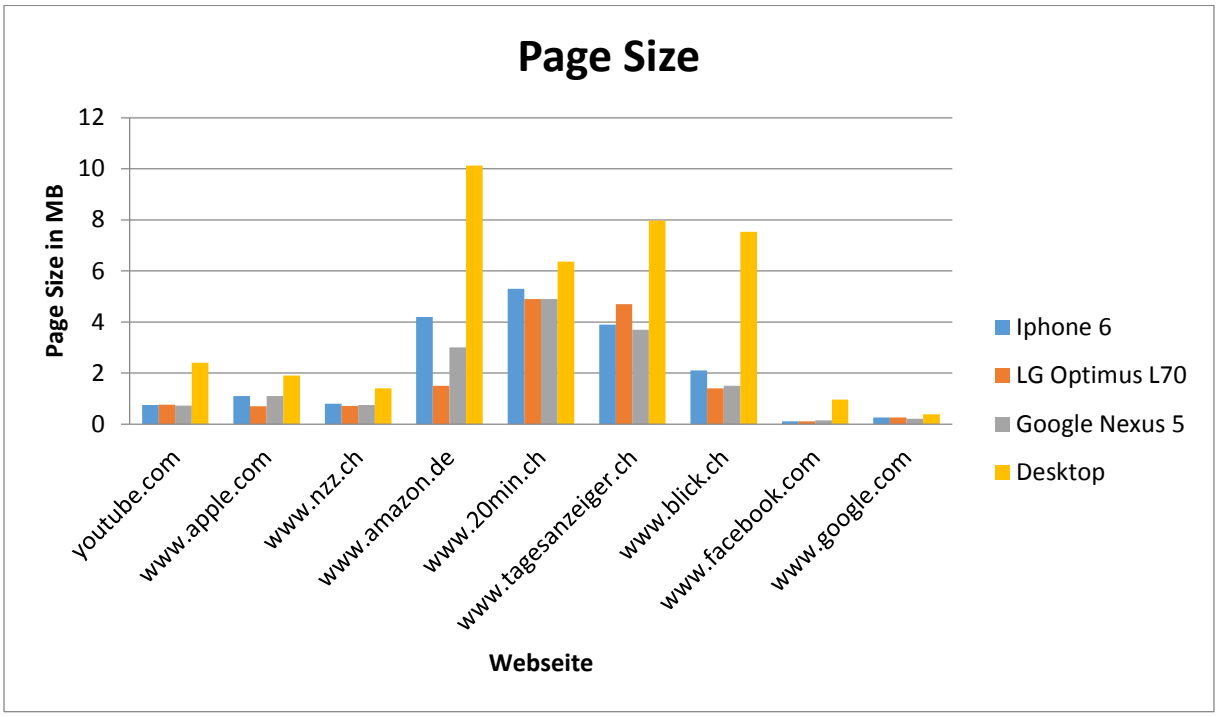


Abbildung 13: Vergleich Desktop - Mobile, Page Size

Bei der Page Size, also der Menge der übertragenen Daten, ist die Optimierung der einzelnen Seiten gut ersichtlich. Alle Seiten sind auf Mobilegeräten kleiner als auf einem Desktop PC. Bis auf amazon.de liegen die Smartphones auch immer nahe zusammen. Aufgrund der Ausreisser bei amazon.de wurden die Messungen für diese Seite noch einige Male durchgeführt. Es zeigte sich, dass die Page Size für das iPhone 6 und das Nexus 5 immer etwa gleich gross ist. Der gemessene Unterschied wird wohl auf unterschiedliche Werbung beziehungsweise Anzeigen auf der Startseite zurückzuführen sein. Die Page Size beim LG Optimus L70 war jedoch konstant tief. Eine gute Erklärung dafür ist, dass Amazon die Displayauflösung der Smartphones abfragt und die Bilder in entsprechender Auflösung sendet, was natürlich die Page Size senkt. Das Optimus L70 hat mit 480 x 800 Pixel mit Abstand die schlechteste Auflösung (Das Nexus 5 besitzt 1920 x 1080 und das iPhone 6 1334 x 750 Pixel).

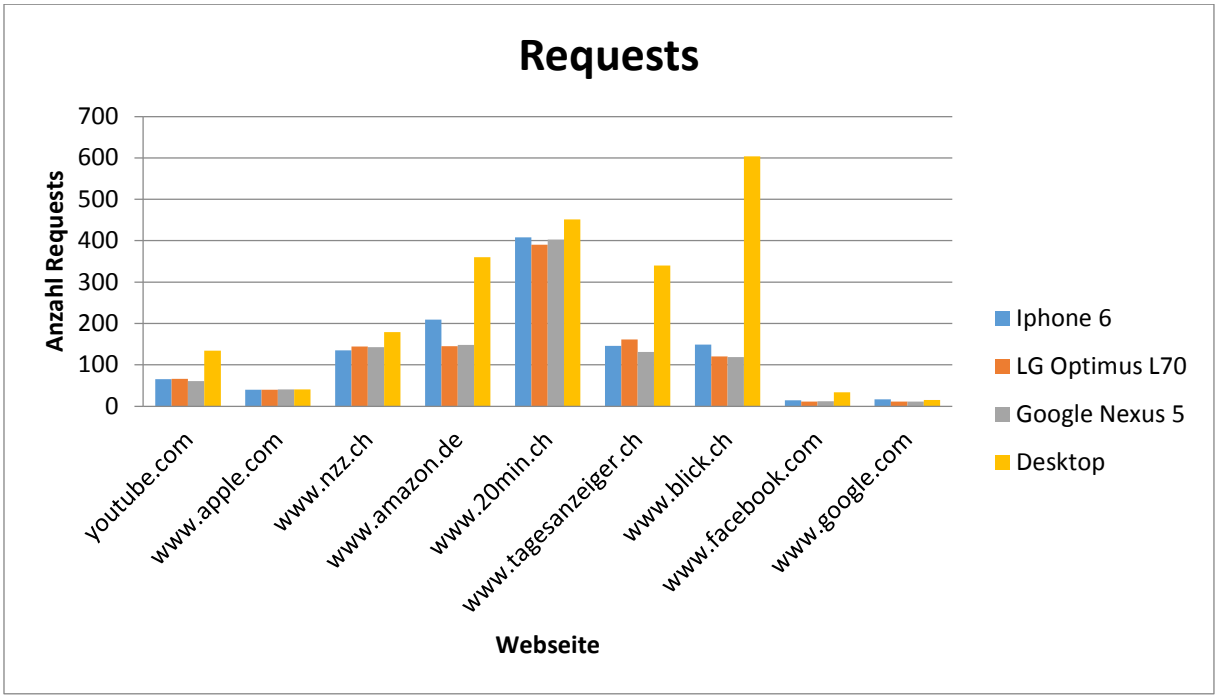


Abbildung 14: Vergleich Desktop - Mobile, Anzahl Requests

Bei der Anzahl Anfragen, welche der Browser an die Webseite sendet, ist die Optimierung auch wieder gut ersichtlich. Die Anzahl Requests ist auf Mobilgeräten immer kleiner als auf dem Desktop PC. Zwischen den einzelnen Smartphones gibt es wieder kaum Unterschiede. Der Ausreisser bei Amazon für das iPhone wurde mit zusätzlichen Messungen untersucht. Dabei zeigte sich, dass die Anzahl Requests beim iPhone immer höher ist als bei den beiden Android Geräten. Es wird daher vermutet, dass Amazon das Betriebssystem erkennt und entsprechend zusätzliche Inhalte lädt.

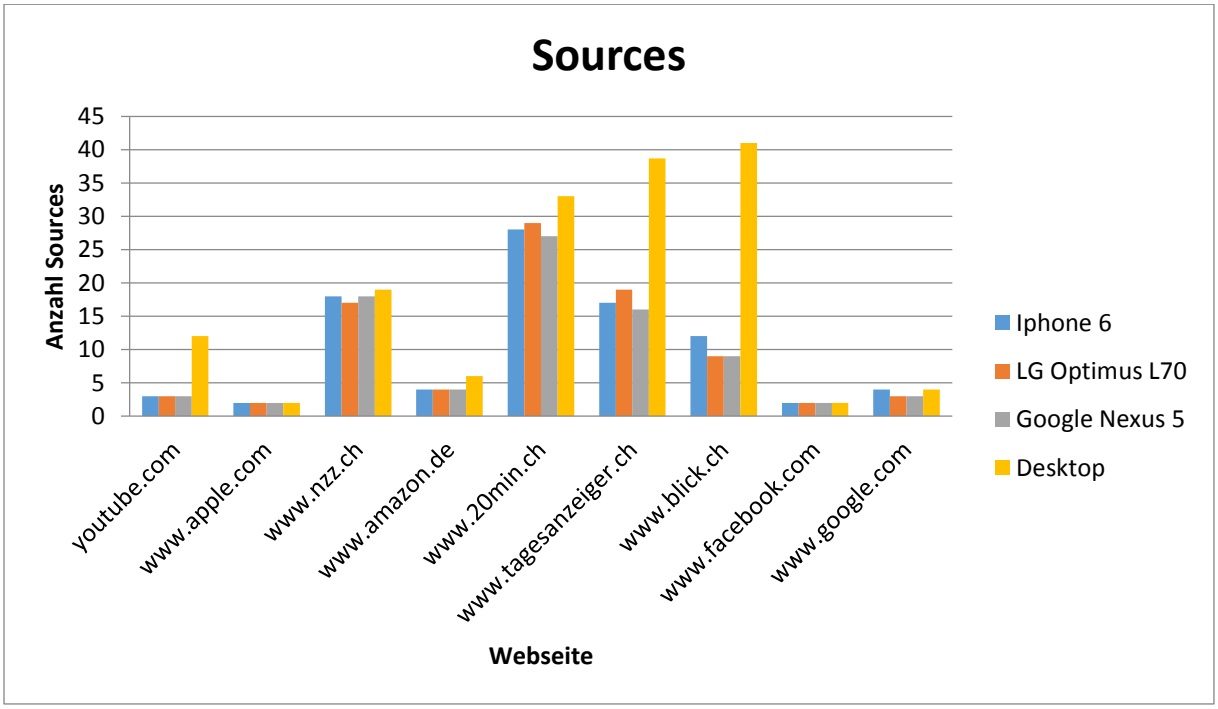


Abbildung 15: Vergleich Desktop - Mobile, Anzahl Sources

Die Anzahl Sources ist meist auf der Desktop Version am höchsten. Hier sieht man jedoch besonders gut, welche Seiten diesbezüglich Optimierungen machten. Während auf der Mobileversion der Webseite von Blick oder Tagesanzeiger deutlich weniger Sources benötigt wurden als auf deren Desktopversion, sind die Anzahl Sources zum Beispiel bei der NZZ auf beiden Versionen nahezu

identisch. Die Anzahl der Sources ist stark von den dargestellten Elementen abhängig, wie zum Beispiel der aktuell dargestellten Werbung. Die teilweise aufgetretene Abweichung zwischen den einzelnen Mobilgeräten kann darauf zurückgeführt werden, dass sich gewisse Elemente zwischen den Messungen geändert haben.

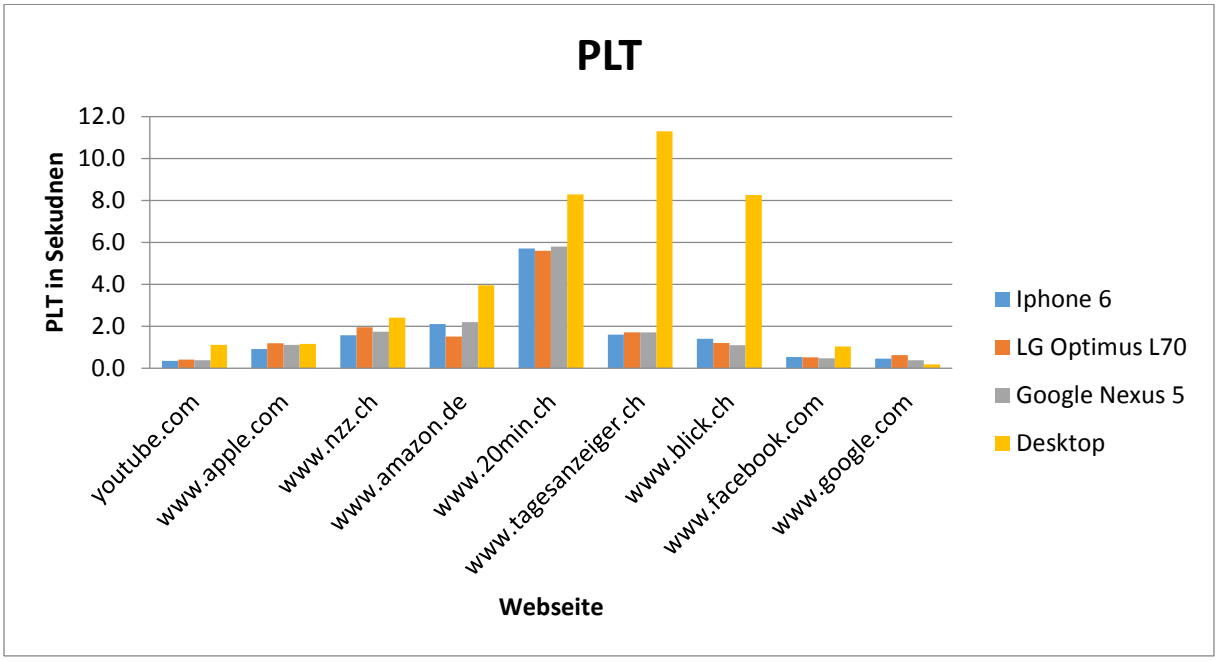


Abbildung 16: Vergleich Desktop - Mobile, PLT

Bei der PLT sind, wohl auch wegen den Optimierungen, die Mobileversionen meist schneller als die Desktopversion. Die Unterschiede sind jedoch auch hier meist minimal.

6 User Experience Testing App

Um die User Experience mit unterschiedlichen Datenraten zu erfassen, wurde eine Android App entwickelt, welche Angaben zu dem jeweiligen Benutzer sowie Messwerte des Surfverhalten in einer Datenbank abspeichert. Die so gewonnenen Daten können später für Auswertungen herangezogen werden.

6.1 Anwendungsszenario

Die App wurde mit dem Ziel entwickelt, diese in verschiedenen Mobilfunkprovider-Shops auf Testgeräten zur Verfügung zu stellen. Dabei können Kunden, welche am Warten sind oder sonst gerade Lust haben, einen Testlauf auf den Geräten starten. Es kann zwischen verschiedenen Datenraten gewählt werden, welche anschliessend mit einem Netzwerkemulator simuliert werden. Der Kunde kann nach einigen persönlichen Angabe über die App auf bestimmten Internetseiten surfen. Dabei sieht der Kunde direkt, welchen Einfluss die Datenrate auf sein Surferlebnis hat. Die App bietet ihm so unter anderem auch eine Hilfestellung für die Auswahl eines entsprechenden Abos. Nach Beendigung des Tests wird nach einem Feedback gefragt, danach ist der Testlauf beendet.

6.2 Testablauf

Ein Testlauf in der App läuft folgendermassen ab: Als Startseite dient ein Infobildschirm, welcher den Ablauf des Tests anzeigt. Startet man den Test, folgt die nächste Seite mit den Angaben zum Benutzer. Dabei wird nach Angaben gefragt, welche einen Einfluss auf die Kundenerwartungen zur Internetgeschwindigkeit haben könnten. Als Beispiel wird nach der Altersgruppe, dem Geschlecht, dem Wohngebiet (Stadt – Land) oder der Selbsteinschätzung zur eigenen Geduld gefragt. Die möglichen Antworten sind in der App vorgegeben, um die spätere Datenauswertung zu vereinfachen. Im nächsten Schritt folgt die Auswahl des zu simulierenden Abos, also welche Datenrate verfügbar ist. Nach Abschluss dieser Angabe wird dem Nutzer eine Auswahl an vorgegeben News-Seiten präsentiert, auf welchen er surfen kann. Während des Surfens werden Daten zur PLT, Seitenname, Anzahl Server sowie geladener Elemente aufgezeichnet.
Hat der Nutzer keine Lust mehr, kann er das Surfen abbrechen. Dabei gelangt er zu einem Feedback Formular, welches nach dem Empfinden zur Surfgeschwindigkeit sowie den Grund für das Beenden des Surfvorgangs fragt. Nach Beantworten dieser zwei Fragen ist der Testlauf abgeschlossen und die Ergebnisse werden der Datenbank hinzugefügt.

6.3 Architektur und Aufbau

Wie für eine App üblich, wurde die Funktionalität auf verschiedene Views verteilt. Die MainActivity steuert dabei, welches Fragment derzeit angezeigt werden soll. Gleichzeitig ist es auch die Schnittstelle zwischen dem GUI und der Business Logik. Da die MainAcitvity, wie bei einem Android App üblich, auch der Programmeintrittspunkt ist, wird von hier aus auch die Datenbank gestartet. Als Datenbank wird die Standard SQLite Datenbank von Android verwendet, weiteres dazu im Anhang 11.1.3 Datenbank.

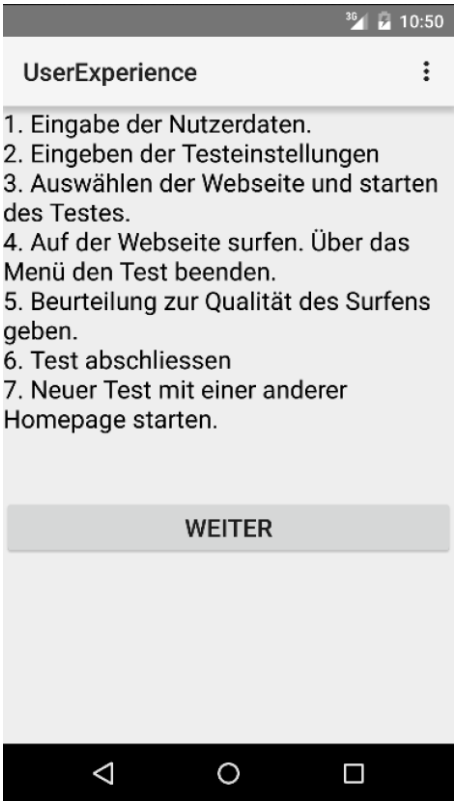


Abbildung 17, Infobildschirm

Auf dem ersten Fragment, welches der Nutzer angezeigt bekommt, steht, wie der Ablauf des Tests ist (Abbildung 17).

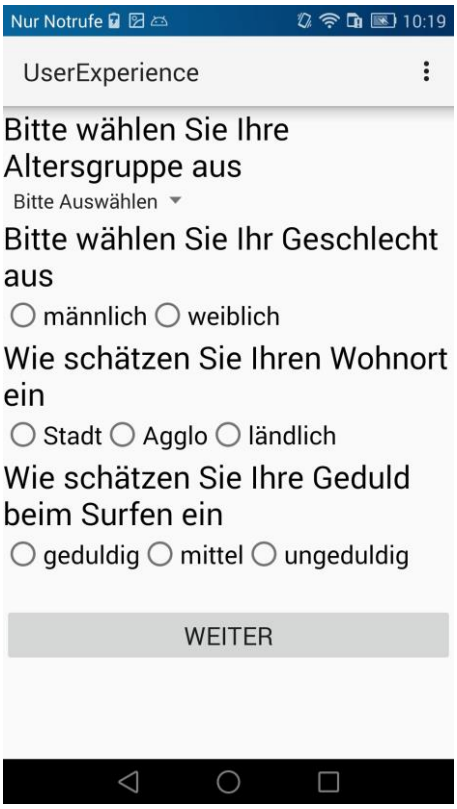


Abbildung 18, Angaben zum Benutzer

Auf dem darauffolgenden Fragment gibt der Nutzer Angaben zu seinem Alter, Geschlecht, Wohnort und seiner Geduld ein (Abbildung 18).

Abbildung 19, Auswahl der Abos

Anschliessend wird das verwendete Abo selektiert (Abbildung 19). Da wir uns bereits bei den vorbereitenden Messungen auf die Swisscom Abos konzentrierten, sind zurzeit nur die Swisscom Infinity Abos verfügbar. Die Abos können jedoch im entsprechenden XML-File sehr einfach ergänzt werden.

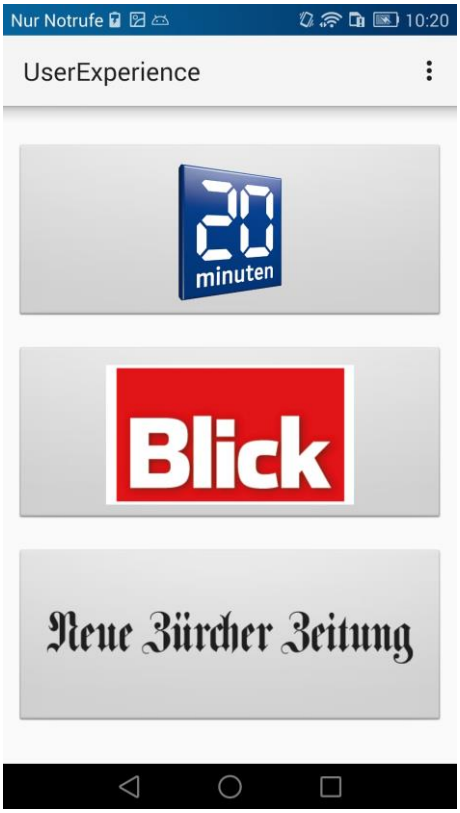


Abbildung 20, Auswahl der Testseite

Nach Auswahl des Abos wählt der Nutzer die Seite aus, auf welcher er surfen will (Abbildung 20). Von den ursprünglich getesteten Seiten haben wir uns hier auf die vier Newsseiten NZZ, 20min, Blick und Tagesanzeiger beschränkt. Während der Entwicklung stellten wir fest, dass die Tests auf der Seite des Tagesanzeigers nicht durchgeführt werden können. Grund dafür ist das bereits bei den ersten Messungen beobachtete Verhalten, dass beim Tagesanzeiger alle Artikel von einer anderen Seite zugeladen werden. Das heisst, sobald das Grundgerüst der Seite geladen ist, wird mitgeteilt, dass die Seite fertig geladen ist, obwohl dies aus Nutzersicht nicht korrekt ist. Selbiges tritt auch auf, wenn auf der Seite ein bestimmter Artikel geöffnet wird. Aus diesem Grund wurde der Test auf die drei anderen Newsseiten, 20Min, NZZ und Blick, beschränkt.

Abbildung 21, Surftest beenden

Der Test wird vom Tester manuell über das Menü beendet (Abbildung 21).

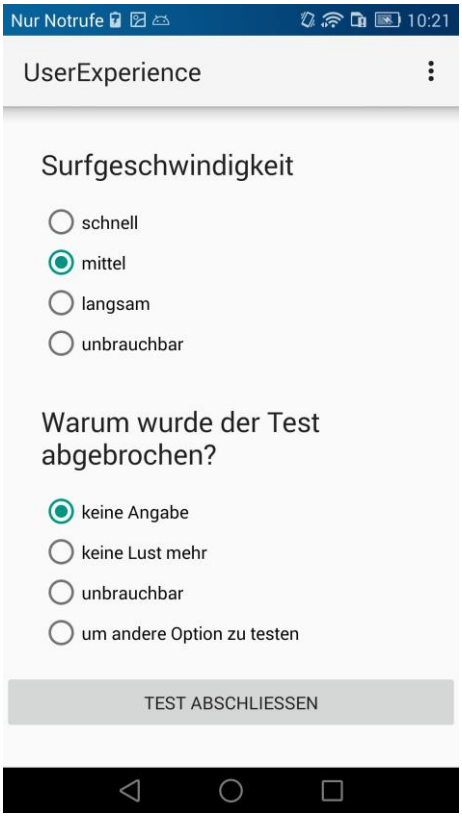


Abbildung 22, Feedback Formular

Sobald der Test beendet ist, wird in einem Feedback Fragment das Nutzerfeedback eingeholt (Abbildung 22). Aktuell wird nur gefragt, als wie schnell der Nutzer das Surferlebnis einstuft und aus welchem Grund der Test abgebrochen wurde. Dies kann bei späteren Erweiterungen noch ausgebaut werden.

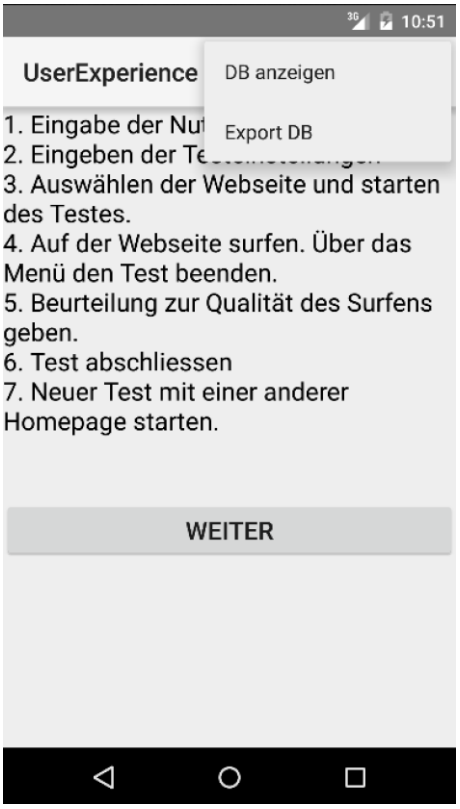


Abbildung 23, Datenbankaktionen

Nach Eingabe und Bestätigung dieser Werte ist der Test beendet. Die Werte werden in die Datenbank geschrieben und wieder die Startseite angezeigt (Abbildung 23).

6.4 Speichern und Auswerten der Daten

Abbildung 24, Auswahl der vorhandenen Tabellen

Wie bereits beschrieben, werden die gespeicherten Daten auf dem Smartphone lokal in eine SQLite Datenbank abgelegt. Im App kann diese über DB anzeigen eingesehen werden (Abbildung 23). Dort werden alle gemäss Datenbankschema existierenden Tabellen aufgelistet (Abbildung 24). Durch Klick auf eine der Tabellen wird deren Inhalt angezeigt.

Über Export DB (Abbildung 23) kann die Datenbank in den Downloadordner des Smartphones exportiert und von dort aus z.B. auf einen Computer übertragen werden.

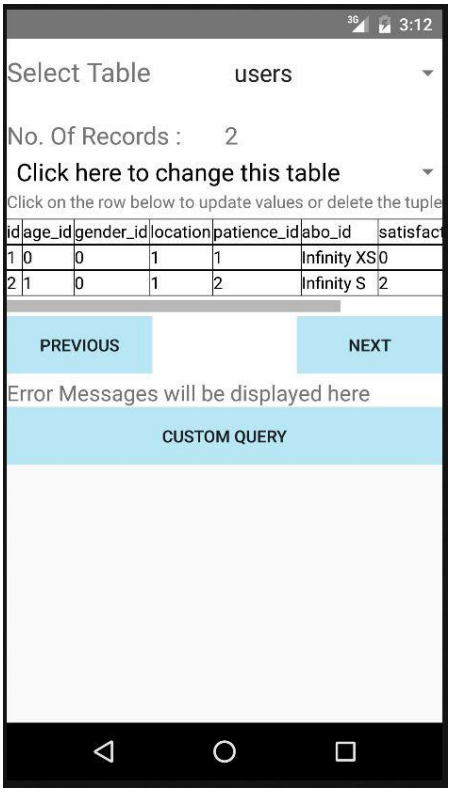


Abbildung 25, "users" - Tabelle

Bis auf „users“ und „tests“ werden alle Tabellen beim Starten der App initialisiert. Die Tabellen „users“ und „tests“ werden nach jedem Testdurchgang mit den entsprechenden Werten ergänzt (Abbildung 25).

Ausser die Abos werden alle Felder mit ID’s abgespeichert, welche auf die jeweiligen Helfertabellen verweisen. Dies erlaubt es später, wenn viele Tests und Users in der Datenbank sind, die Datenbank effizient zu filtern und abzufragen. Die Abos sind als String erfasst, da diese sich schnell ändern.

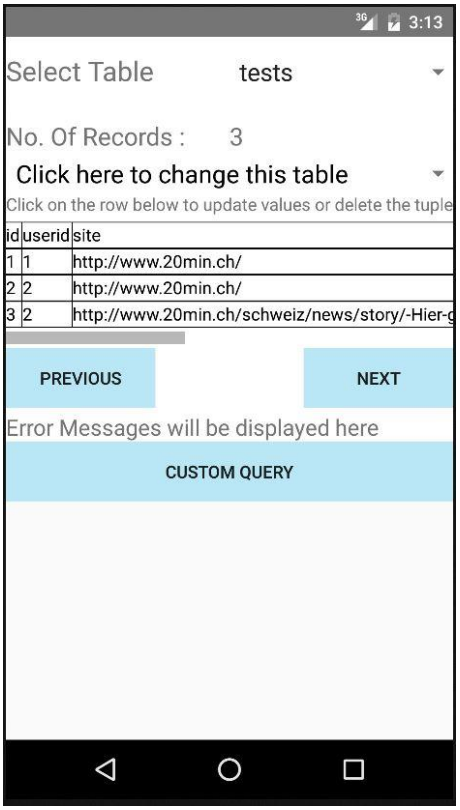


Abbildung 26, "tests" - Tabelle

In der Tabelle „tests“ wird gespeichert, welche Seite aufgerufen wurde, wie lange es dauerte (PLT), wie viele Elemente geladen wurden (zum Beispiel Bilder) und von wie vielen verschiedenen Servern Daten bezogen wurden. Zudem wird der User, welcher den Test durchführte, referenziert (Abbildung 26).

6.5 Testauswertung

Da im Rahmen dieser Arbeit leider keine Zeit mehr blieb, genügend Testläufe durchzuführen, welche für statistisch relevante Auswertungen notwendig gewesen wären (siehe Anhang 11.1.4 Tests), wurde entschieden, einige Tests in Bezug auf die Benutzbarkeit der App durchzuführen. So wurde die soweit entwickelte App zur Benutzung den Testpersonen übergeben. Diese mussten anschliessend einen kurzen Fragebogen ausfüllen.
Dabei kam heraus, dass noch an einigen kleineren Dingen Verbesserungspotenzial in der Benutzbarkeit vorhanden ist. So wurde kritisiert, dass man nicht genau weiss, wie man den Test richtig beendet, wenn man auf den Newsseiten am Surfen ist. Auch würde ein Ladebalken bei der SurfView begrüsst werden. Zur besseren Übersicht und Führung durch den Testlauf wäre ein jeweils angepasster Statusbar-Text sinnvoll, welcher einem auch gleich verrät, wie viele Schritte noch folgen werden (z.B. „Abo auswählen (2/5)“). Zum Schluss des Testlaufs sollte ein Pop-Up dann dem Nutzer anzeigen, dass der Test nun erfolgreich abgeschlossen wurde. Trotz dieser Mängel wurde die Idee hinter der App von allen als gut und nützlich eingestuft.

6.6 Ausblick

In Zukunft könnte die App noch in einigen Hinsichten verbessert werden. Zum einen die in der Testauswertung gewonnen Erkenntnisse, welche vor allem das User Interface betreffen. Dieses könnte noch ansprechender gestaltet werden. Zum anderen sind auch Verbesserungen bzw. Erweiterungen in der Logik der App vorstellbar. So wird es in Zukunft Sinn machen, sobald mehrere Testgeräte im Umlauf sind, dass die Datenbank zentral auf einem Server läuft und alle Geräte ihre Testdaten dann an diese senden werden.

Ein weiterer Schwachpunkt ist die in der App verwendete WebView. Diese ist bekanntlich nicht so schnell und nicht ganz so stabil wie der native Chrome Browser auf der Android Plattform. Vielleicht wird es in Zukunft möglich sein, die WebView durch eine Integration des Chrome Browsers zu ersetzen. Dann wird die Benutzererfahrung auch nicht mehr durch die langsamer arbeitende WebView verfälscht und die Testergebnisse genauer. Leider existierte diese Möglichkeit zum Zeitpunkt der Arbeit noch nicht.

7 Zusammenfassung

Im Rahmen dieser Arbeit wurden als erstes Messungen zur Page Load Time auf einem Android Smartphone gemacht. Im ersten Durchlauf wurden die Werte manuell mithilfe des Chrome Browsers aufgenommen. Dazu wurde mithilfe eines Apposite Netzwerkemulators verschiedene Datenraten und Round Trip Times simuliert. Beim Verhältnis zwischen Download- und Uploaddatenrate orientierte man sich an den Swisscom Infinity Abos. Als Datenrate wurden verschiedene Werte zwischen 0.2 und 100 Mbps gewählt. Die Round Trip Time wurde mit einem zusätzlichen Delay von 0, 20, 40 und 60ms getestet. Aus den Tests wurde erkannt, dass die grössten Unterschiede zwischen den einzelnen Messungen vor allem im tiefen Bereich bis 5 Mbps auftraten. Über 5 Mbps ändert sich die PLT nur noch minimal. Selbiges gilt auch für die Round Trip Time. In beiden Fällen wirken sich schlechte Werte besonders auf Seiten mit vielen Inhalten negativ auf.

Nach Abschluss der ersten Messungen wurden dieselben Messungen über eine eigens entwickelte App automatisiert wiederholt. Dazu wurde ein Prototyp entwickelt, welcher etwas abgeändert auch in der entwickelten App verwendet wurde. Bei dieser Messreihe fiel vor allem auf, dass die verwendete WebView langsamer ist als der Chrome Browser. Zudem lief die WebView bei vielen aufeinanderfolgenden Webaufrufen nicht stabil. Somit konnten die Messungen für extreme Werte nicht durchgeführt werden, da die WebView jeweils abstürzte. Ansonsten wurden die Erkenntnisse der ersten Messungen bestätigt.

Im Anschluss wurde verglichen, was für Unterschiede beim Aufruf von Webseiten zwischen verschiedenen Smartphones und einem Desktop PC gemessen werden konnten. Es zeigte sich, dass die Werte der Smartphones meist nahe beieinanderliegen. Die Aufrufe mit dem Desktop-PC unterschieden sich jedoch meist deutlich.

Aufgrund der Aktualität wurde das Thema Preloading näher behandelt. Preloading bezeichnet das laden von Webinhalten, bevor der Nutzer diese effektiv benötigt. Somit wird die PLT positiv beeinflusst, da zum Beispiel beim Anklicken eines Links gewisse Elemente der dahinterliegenden Webseite bereits geladen wurden.

Mit all diesen Erkenntnissen wurde die Entwicklung der App in Angriff genommen. Mit Hilfe der App soll es möglich sein, verschiedene Mobile Abos zu testen und zu bewerten. Dies soll Kunden beim Kauf und Mobilfunkprovidern beim Entwickeln neuer Abos unterstützen. Bei der Benützung der App werden zuerst Angaben zum Nutzer abgefragt. Anschliessend wird das zu testende Abo ausgewählt. Danach kann auf drei vordefinierten Seiten, 20min, Blick und NZZ gesurft werden. Während dem Surfen speichert die App, welche Seiten aufgerufen wurden, wie gross die PLT war, wie viele Elemente geladen wurden und von wie vielen verschiedenen Servern Daten bezogen wurden. Am Ende wird vom Nutzer ein Feedback zum Test verlangt. All diese Daten werden in einer Datenbank auf dem Smartphone gespeichert. Diese Datenbank kann zum einen über die App eingesehen, zum anderen kann sie direkt in den Downloadordner des Smartphones exportiert und von dort zum Beispiel auf einen Desktop PC kopiert werden. Dadurch können Daten von mehreren Smartphones gesammelt und über SQL einfach ausgewertet werden.

8 Ausblick

Als nächste Schritte nach dieser Arbeit könnten nun, nachdem die kleineren Verbesserungen im Kapitel 6.5 Testauswertung an der App gemacht wurden, die Tests durchgeführt werden. Die so mit der App ausgestatteten Geräte können in Mobilfunkprovider-Shops, aber auch an anderen Orten und Anlässen, zum Einsatz gebracht werden.

Vor den Tests muss jedoch noch geklärt werden, wie man die zu testenden Netzwerkparameter simuliert. Die korrektesten Werte würde man erhalten, wenn man wie in dieser Arbeit vorgeht. Dabei müsste jedoch an jedem Ort, an welchem die Tests durchgeführt werden, ein Netzwerkemulator sowie ein daran angeschlossener WLAN Access Point eingerichtet werden. Der Netzwerkemulator müsste zudem bei jedem Test nach dem ausgewählten Abo konfiguriert werden. Zu prüfen wäre, ob es möglich ist, den Emulator gleich über die App zu konfigurieren. Die würde die Benützung wesentlich komfortabler machen.

Eine weitere Möglichkeit bestünde darin, dass man auf den verschiedenen Testgeräten jeweils eine SIM Karte mit dem entsprechenden Abo einfügt, und so die Tests durchführt. Dadurch entfallen Netzwerkemulator sowie WLAN Access Point. Dies hat jedoch die Nachteile, dass eine gute Netzabdeckung benötigt wird und die Datenraten jeweils von der aktuellen Auslastung der Netzwerkzelle abhängt, was beides die Testergebnisse verfälschen kann.

9 Literaturverzeichnis

[1] mozilla.org, 12 2015. [Online]. Available: https://addons.mozilla.org/de/firefox/addon/firebug/.

[2] selenium, 11 2015. [Online]. Available: http://www.seleniumhq.org/.

[3] Google, 16 10 2015. [Online]. Available: https://play.google.com/store/apps/details?id=com.android.chrome&hl=de. [Zugriff am 16 10 2015].

[4] C. DevTools, „Remote Debuggin on Android with Chrome," 15 10 2015. [Online]. Available: https://developer.chrome.com/devtools/docs/remote-debugging. [Zugriff am 15 10 2015].

[5] Huawei, 16 10 2015. [Online]. Available: http://consumer.huawei.com/minisite/worldwide/Ascend-Mate7/. [Zugriff am 16 10 2015].

[6] Netgear, 16 10 2015. [Online]. Available: http://www.netgear.ch/home/products/networking/wifi-routers/WNR1000.aspx. [Zugriff am 16 10 2015].

[7] Apposite, 16 10 2015. [Online]. Available: http://www.apposite-tech.com/assets/pdfs/linktropy-5500-hwguide.pdf. [Zugriff am 16 10 2015].

[8] Swisscom, 05 10 2015. [Online]. Available: https://www.swisscom.ch/de/privatkunden/mobile/abos-tarife/infinity.html?ext-campID=SEA_GDE_INF_LE&gclid=CKaWp8z3qsgCFQbkwgodd0AFUw. [Zugriff am 05 10 2015].

[9] S. Souders, 11 2015. [Online]. Available: http://www.stevesouders.com/blog/2013/11/07/prebrowsing/.

[10] I. Grigorik, „chimera.labs.oreilly.com," 12 2015. [Online]. Available: http://chimera.labs.oreilly.com/books/1230000000545.

[11] cnlab, 11 2015. [Online]. Available: https://www.cnlab.ch/.

[12] mozilla.org, 11 2015. [Online]. Available: https://www.mozilla.org/de/.

[13] Wikipedia, 11 2015. [Online]. Available: https://de.wikipedia.org/wiki/Ethernet.

[14] Wikipedia, 11 2015. [Online]. Available: https://de.wikipedia.org/wiki/Router.

[15] P. D. R. Lackes, 11 2015. [Online]. Available: http://wirtschaftslexikon.gabler.de/Definition/dsl.html.

[16] wikipedia, 11 2015. [Online]. Available: https://en.wikipedia.org/wiki/3G.

[17] wikipedia, 11 2015. [Online]. Available: https://en.wikipedia.org/wiki/4G.

[18] wikipedia, 11 2015. [Online]. Available: https://en.wikipedia.org/wiki/USB.

[19] wikipedia, 11 2015. [Online]. Available: https://en.wikipedia.org/wiki/Enhanced_Data_Rates_for_GSM_Evolution.

[20] wikipedia, 11 2015. [Online]. Available: https://en.wikipedia.org/wiki/General_Packet_Radio_Service.

[21] wikipedia, 11 2015. [Online]. Available: https://en.wikipedia.org/wiki/Universal_Mobile_Telecommunications_System.

[22] wikipedia, 11 2015. [Online]. Available: https://en.wikipedia.org/wiki/LTE_%28telecommunication%29.

[23] wikipedia, 11 2015. [Online]. Available: https://de.wikipedia.org/wiki/Transmission_Control_Protocol.

[24] wikipedia, 11 2015. [Online]. Available: https://de.wikipedia.org/wiki/JavaScript.

[25] sanathp, 11 2015. [Online]. Available: https://github.com/sanathp/DatabaseManager_For_Android.

10 Glossar

PLT	Page Load Time, die Zeit die eine Webseite braucht, um vollständig angezeigt zu werden.
cnlab	Eine auf Netzwerkanalyse spezialisierte Firma mit Sitz in Rapperswil SG. [11]
Firefox	Der von mozilla.org vertriebene Internet Browser. [12]
Ethernet	Ethernet ist eine Technologie, welche Soft- und Hardware für Kabelgebundene Netzwerke spezifiziert. [13]
Router	Ein Router ist ein Netzwerkgerät, welches Daten zwischen mehreren Netzwerken weiterleitet. [14]
DSL	Sammelbegriff, der die Übertragungstechnologien für Daten bezeichnet, mit mit denen sich Breitband-Teilnehmeranschlüsse und damit eine hohe Übertragungsgeschwindigkeit über das "normale" Telefonnetz ermöglichen lassen, ohne dass der Telefondienst eines Anschlusses beeinträchtigt wird. [15]
Cablenet	Kabelnetz
Selenium	Ein Tool, welches Browser automatisiert. [2]
Firebug	Ein Web Development Tool für Firefox [1]
3G	Eine Technologie im Mobilfunk [16]
4G	Eine Technologie im Mobilfunk [17]
Android	Ein von Google entwickeltes Betriebssystem, welches unter anderem auf gewissen Smartphones verwendet wird.
USB	Universal Serial Bus. Ein Standard, welcher Kabel, Verbindung und Protokolle für die Verbindung, Kommunikation und Stromversorgung zwischen Computern und elektronischen Geräten definiert. [18]
Datenrate	Gibt an, wie schnell Daten Herunter- oder Hochgeladen werden können.
Bandwidth	Siehe Datenrate
Packet-Loss	Bezeichnet das Phänomen, dass Pakete im Netzwerk verloren gehen.
EDGE	Eine Technologie im Mobilfunk [19]
User Experience	Nutzererlebnis
Mobilfunkprovider	Firmen, welche Netze und Internetzugang für mobile Geräte zur Verfügung stellen.
RTT	Round-Tripp-Time.
GPRS	Eine Technologie im Mobilfunk [20]
UMTS	Eine Technologie im Mobilfunk [21]
LTE	Long Term Evolution. Bezeichnet eine Technologie im Mobilfunk. [22]
Netzemulator	Ein Gerät, mit dem Einflüsse auf das Netzwerk simuliert werden können.
Preloading	Siehe Kapitel 6 dieser Arbeit. [9]
Prefetching	Siehe Kapitel 6 dieser Arbeit. [9]
DNS	Distributed Name System. Wird verwendet, um den Namen einer Webseite auf deren IP-Adresse abzubilden.
Prerendering	Siehe Kapitel 6 dieser Arbeit. [9]
TCP	Transmission Control Protocol. Ein Protokoll zum Transport von Datenpaketen im Internet. [23]
Javascript	Eine Programmiersprache. [24]
Source	Server, von welchen Elemente der Webseite geladen werden.
Request	Eine Anfrage an einen Webserver

11 Anhang

11.1 Entwicklung der App

11.1.1 Planung

Bei der Entwicklung der App wurde als erstes die Kernfunktion, nämlich das Aufzeichnen der PLT beim Laden einer Internetseite, implementiert. Als dies funktionierte, ging es daran, den Rest der App entsprechend den Anforderungen darum herum zu bauen. Dafür wurde definiert, wie der Test Schritt für Schritt ablaufen soll. Aus diesen Erkenntnissen wurden die einzelnen Views der App mit Hilfe eines Mockup Tools gestaltet und den Betreuern vorgelegt. Aufgrund dieser Informationen wurde der definitive Aufbau der App definiert. Es wurde beschlossen, als erstes eine funktionsfähige App zu programmieren. Danach sollten zusätzliche Features wie eine Datenbankanbindung hinzugefügt und Tests durchgeführt werden.

11.1.2 Anpassungen

Ursprünglich war geplant, dem Nutzer am Ende des Tests ein Fragment anzuzeigen, auf welchem alle Testresultate sowie die Nutzerangaben zusammengefasst anzuzeigen. Dieses Fragment wurde im abgegebenen App weggelassen, da der Nutzen davon als zu gering eingestuft wurde. Um die Testresultate zu sehen, kann man sich im App sehr einfach die Datenbank anzeigen lassen. Die vom Nutzer getätigten Eingaben nochmals anzuzeigen wurde im Nachhinein als sinnlos erachtet, da zu diesem Zeitpunkt im Programmablauf eine Veränderung der Eingaben nicht mehr möglich ist.

11.1.3 Datenbank

Als Datenbank dient die auf Android Geräten standardmässig vorhandene SQLite Datenbank. Vor der Implementierung der Datenbank wurde das folgende Datenbankschema definiert (Abbildung 27, Datenbankschema).

Tabelle 1: User

ID	Alter ID	Geschlecht ID	Wohnort ID	Geduld ID	Abo	Zufriedenheit ID	Abgebrochen-Grund ID

Tabelle 2: Testresults

ID	User-ID	Seite	PLT

Tabelle 3: Age

ID	Altersgruppe
0	0-15
1	16-30
2	31-45
3	46-

Tabelle 4: Gender

ID	Geschlecht
0	Männlich
1	weiblich

Tabelle 5: Location

ID	Wohnort
0	Stadt
1	Agglomeration
2	ländlich

Tabelle 6: Patience

ID	Geduld
0	geduldig
1	mittel
2	ungeduldig

Tabelle 7: Satisfaction

ID	Zufriedenheit
0	Schnell
1	Mittel
2	Langsam
3	unbrauchbar

Tabelle 8: Aborted

ID	Grund
0	Keine Angabe
1	Keine Lust mehr
2	Unbrauchbar
3	Um andere Option zu testen

Abbildung 27, Datenbankschema

Für die Steuerung der Datenbank wurden zwei Klassen erstellt. Die Klasse DbHelper erstellt die einzelnen Tabellen und fügt die entsprechenden Werte ein. Aufgerufen wird diese Klasse von der MainActivity aus. Die Klasse AndroidDatabaseManager, welche von einem öffentlichen GitHub Projekt bezogen wurde [25], ist für die Darstellung der Datenbank zuständig.

Das Speichern der Daten wird über die MainActivity gesteuert. Während der Test läuft, wird immer beim Wechseln eines Fragments die innerhalb dieses Fragments gesammelten Daten entweder in einer Klasse User, wenn es den Nutzer betrifft, oder in einer Klasse TestResults, wenn es Daten wie die PLT sind, gespeichert. Sobald der Test beendet wird, ruft die MainActivity die für das Schreiben der Werte in die Datenbank verantwortliche Methode der DbHelper Klasse auf. Diese liest die Werte aus der Klasse User und TestResults aus. Somit kann garantiert werden, dass nur erfolgreich beendete Testdurchläufe in die Datenbank gespeichert werden.

11.1.4 Tests

Zu Beginn der Entwicklungsarbeit wurde geplant, mit der App durch reale Benutzer die einzelnen Abos zu testen. Die daraus gewonnenen Erkenntnisse sollten analysiert werden. Im Verlauf der Arbeit kristallisierte sich jedoch heraus, dass dafür zu wenig Zeit ist. Die Anzahl Personen, welche die App für eine statistisch relevante Aussage nutzen müssten, wäre ebenfalls zu gross für diese

Studienarbeit. Damit diese Tests in Zukunft einfach durchgeführt werden können, haben wir bereits die folgenden Hypothesen aufgestellt:

1. Alte Personen sind geduldiger als Junge.
2. Junge Personen werden vor allem 20min und Blick besuchen.
3. Personen, welche sich selbst als ungeduldig einstufen, drücken öfters auf „Test abbrechen“.
4. Personen, welche sich als geduldig einstufen, sind bereits mit tiefen Bandbreiten zufrieden. Ungeduldige nicht.
5. Personen stufen sich selbst als geduldiger ein als sie dann effektiv sind.
6. Man wird ein unterschiedliches Surfverhalten zwischen alt/jung und geduldig/ungeduldig feststellen.
7. Zwischen schnellen und sehr schnellen Abos wird man kaum Unterschiede in den Kundenbewertungen sehen.
8. Bei den Abos mit tiefen Bandbreiten werden die Tests oft durch den Nutzer manuell abgebrochen.
9. Die User Experience wird schlechter, je mehr Elemente geladen werden müssen.
10. Die Personalisierung der Webseite wirkt sich auf die Kundenzufriedenheit (Antwortzeit) aus.
11. Zufriedenheit wird mehr durch den Typ Mensch als durch die effektiv gemessene Antwortzeit bestimmt.
12. Ab einer bestimmten Datenrate (ca. 5Mbit/s) wird die Zufriedenheit nicht mehr stark ansteigen.
13. Bei der RTT wird man im Bereich bis etwa 80ms keine grossen Unterschiede feststellen.
14. Bei sehr niedrigen Datenraten wird die Zufriedenheit mit nur leicht steigender Datenrate stark steigen.

Auch haben wir die notwendige Anzahl Testpersonen berechnet. Aus dem oben beschriebenen Datenbankschema ist ersichtlich, dass es 4 Altersgruppen, 2 Geschlechter, 3 Wohnsituationen und 3 Geduldsstufen gibt. Dies ergibt total 84 verschiedene Personengruppen. Bei 10 Personen für eine Gruppe bedeutet dies, dass etwa 840 Personen Tests mit der App durchführen sollten.

11.2 Erklärung Urheberschaft und Verwendung

Erklärung zur Urheberschaft[1]

Die vorliegende Arbeit basiert auf Ideen, Arbeitsleistungen, Hilfestellungen und Beiträgen gemäss folgender Aufstellung:

Gegenstand, Leistung	Person	Funktion
Bericht, Messungen, Implementierung der App	Matthias Fehr	Autor der Arbeit
Bericht, Messungen, Implementierung der App	Dominic Peisker	Autor der Arbeit
Idee, Aufgabenstellung, allgemeines Pflichtenheft, Betreuung während der Arbeit	Prof. Dr. P. Heinzmann	Verantwortlicher Professor
Industriepartner, Betreuung während der Arbeit.	E. Franke, cnlab AG	Industriepartner

Ich erkläre hiermit,

- dass ich die vorliegende Arbeit gemäss obiger Zusammenstellung selber und ohne weitere fremde Hilfe durchgeführt habe,
- dass ich sämtliche verwendeten Quellen erwähnt und gemäss gängigen wissenschaftlichen Zitierregeln korrekt angegeben habe,
- dass in der Arbeit verwendete urheberrechtlich geschützte (Copyright) Inhalte (insbesondere Fotografien und Grafiken) klar gekennzeichnet und mit Quellenhinweis versehen sind,
- dass Inhalte die unter Creative-Commons-Lizenz veröffentlicht wurden, klar gekennzeichnet sind.

Rapperswil, den 14.12.15 .....
Student

Rapperswil, den 14.12.15 .....
Student

[1] Diese Erklärung basiert auf der Muster-Erklärung in den *Richtlinien der HSR zur Durchführung von Projekt-, Studien-, Diplom- oder Bachelorarbeiten* vom 16. Februar 2009.

Vereinbarung zur Verwendung und Weiterentwicklung der Arbeit[1]

1. Gegenstand der Vereinbarung

Mit dieser Vereinbarung werden die Rechte über die Verwendung und die Weiterentwicklung der Ergebnisse der Studienarbeit „User Experience" von Matthias Fehr und Dominic Peisker unter der Betreuung von Prof. Dr. P. Heinzmann (für die Arbeit verantwortlicher Professor) geregelt.

2. Urheberrecht

Die Urheberrechte stehen der Studentin / dem Student zu.

3. Verwendung

Die Ergebnisse der Arbeit dürfen sowohl von allen an der Arbeit beteiligten Parteien, d.h. von den Studenten, welche die Arbeit verfasst haben, vom verantwortlichen Professor sowie vom Industriepartner verwendet und weiter entwickelt werden. Die Namensnennung der beteiligten Parteien ist bei der Weiterverwendung erwünscht, aber nicht Pflicht.

Rapperswil, den 14.12.15 .....
Student

Rapperswil, den 14.12.15 M. Fe
Student

Rapperswil, den 14.12.15 .....
Verantwortlicher Professor

Rapperswil, den 14.12.15 Eric Fr...
Industriepartner

[1] Diese Vereinbarung basiert auf den Muster-Vereinbarungen in den *Richtlinien der HSR zur Durchführung von Projekt-, Studien-, Diplom- oder Bachelorarbeiten* vom 16. Februar 2009.