



Evaluierung von Techniken zur parallel-synchronen Bedienung einer Web-Applikation auf verschiedenen mobilen Endgeräten

vorgelegt von

Adrian Randhahn

EDV.Nr.:744818

dem Fachbereich VI – Informatik und Medien –
der Beuth Hochschule für Technik Berlin vorgelegte Bachelorarbeit
zur Erlangung des akademischen Grades

Bachelor of Science (B.Sc.)

im Studiengang

Medieninformatik

Tag der Abgabe 6. März 2014

Gutachter

Prof. Knabe Beuth Hochschule für Technik

Prof. Dr. Wambach Beuth Hochschule für Technik

Erklärung

Ich versichere, dass ich diese Abschlussarbeit ohne fremde Hilfe selbstständig verfasst und nur die angegebenen Quellen und Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich gemacht. Ich erkläre weiterhin, dass die vorliegende Arbeit noch nicht im Rahmen eines anderen Prüfungsverfahrens eingereicht wurde.

Datum

Unterschrift

Entwurf

Sperrvermerk

Die vorliegende Arbeit beinhaltet interne und vertrauliche Informationen der Firma New Image Systems GmbH. Die Weitergabe des Inhalts der Arbeit im Gesamten oder in Teilen sowie das Anfertigen von Kopien oder Abschriften - auch in digitaler Form - sind grundsätzlich untersagt. Ausnahmen bedürfen der schriftlichen Genehmigung der Firma New Image Systems GmbH.

Entwurf

Inhaltsverzeichnis

1	Einleitung	2
2	Aufgabenstellung	5
2.1	Problemstellung	5
2.2	Annahmen und Einschränkungen	5
2.3	Zielsetzung	5
2.4	Abgrenzungskriterien	5
3	Grundlagen	6
3.1	Begriffsklärung	6
3.1.1	parallel-synchron	6
3.1.2	Web-Applikation	6
3.1.3	Endgeräte	6
3.2	technischer Aufbau	6
3.3	Komponenten	6
3.3.1	Raspberry Pi	6
3.3.2	Hardware	6
4	Technologien	7
4.1	Ghostlab	7
4.2	NodeJS	7
4.3	Zombie.js	7
4.4	W3C Touch Events Extensions	7
4.5	Phantom Limb	7
4.6	jQuery UI Touch Punch	7
4.7	jQuery Touchit	7
4.8	NPM touchit	7
5	Lösungsansätze	8
5.1	Adobe Edge Inspect	8
5.2	Ghostlab	8
5.3	Remote Preview	8
5.4	Browser-Sync	8
5.5	Eigenes Framework	8
6	Evaluation	9
6.1	Adobe Edge Inspect	10
6.1.1	Einführung	10
6.1.2	Vorteile	10
6.1.3	Nachteile	10

6.2	Ghostlab	10
6.2.1	Einführung	10
6.2.2	Vorteile	10
6.2.3	Nachteile	10
6.3	Remote Preview	10
6.3.1	Einführung	10
6.3.2	Vorteile	10
6.3.3	Nachteile	10
6.4	Browser-Sync	10
6.4.1	Einführung	10
6.4.2	Vorteile	10
6.4.3	Nachteile	10
6.5	Eigenes Framework	10
6.5.1	Einführung	10
6.5.2	Systementwurf	10
6.5.3	Vorteile	10
6.5.4	Nachteile	10
7	Helpers	11
7.1	quote	11
7.2	longquote	11
7.3	fussnote	11

Abbildungsverzeichnis

1.1 Entwicklungsprozess 3

Entwurf

Tabellenverzeichnis

Entwurf

1 Einleitung

In der modernen Webentwicklung durchläuft eine Anwendung verschiedene Etappen eines Entwicklungszykluses. Er beginnt bei einem Auftrag oder einer Idee, darauf folgt dann die Spezifikation einzelner Usecases¹. Im Anschluss folgt in der Regel die Entwicklung und Implementation² der einzelnen Komponenten. Am Ende der jeweiligen Implementationsphase durchläuft das Produkt³ die Qualitätskontrolle. Sollten in diesem Abschnitt Fehler auftreten wird das Produkt dem Entwickler zur erneuten Bearbeitung vorgelegt. Dieser Vorgang kann sich beliebig oft wiederholen. Bei großen und komplexen

Softwaresystemen ist es trotz zeitgemäßer Implementierung nicht immer Ausgeschlossen, dass Kaskadierungsfehler⁴ entstehen. Aus Sicht der Qualitätssicherung ist dies ein lästiges Problem, da diese nach jedem erneuten Modifikationsvorganges eines Softwaresegments einen größeren Segmentblock, wenn nicht sogar das gesamte Softwareystem erneut testen muss. Bei der Entwicklung auf und für mobile Endgeräte⁵ kommt noch ein erschwerender

Faktor hinzu, nämlich die diversen, verschiedenen Bildschirmauflösungen. Diese können nicht nur die Darstellung des Inhaltes beeinflussen, sondern auch daraus folgend die Interaktionskonformität beeinflussen.

Im Optimalfall wird die Software erst nach vollständiger Homogenität auf allen unterstützten Geräten freigegeben.

Dieser zyklisch wiederkehrende Prozessablauf ist sehr Zeitintensiv und nimmt linear mit der Anzahl der zu testenden Geräte zu.

Das Ergebnis dieser Forschungsarbeit soll zeigen, wie verschiedene Softwareframeworks die Zeit, die in die Qualitätssicherung investiert wird, beeinflussen können, indem sie die Steuerung diverser Geräte parallel-synchron steuern. Die Evaluierung soll zeigen wo die Vorteile und Nachteile der einzelnen Werkzeuge liegen. Weiterhin soll gezeigt werden ob aktuelle Frameworks erweiterbar sind um Beispielsweise automatisierte Testunits zu implementieren.

¹Szenario oder auch Anwendungsfall

²Einbindung

³hier: einzelne Softwarekomponente

⁴Fehler die nicht im eigentlichen Segment auftreten, sondern eine oder mehr Ebenen weiter unten in der Systemhierarchie

⁵Smartphones, Tablets oder Ähnliche

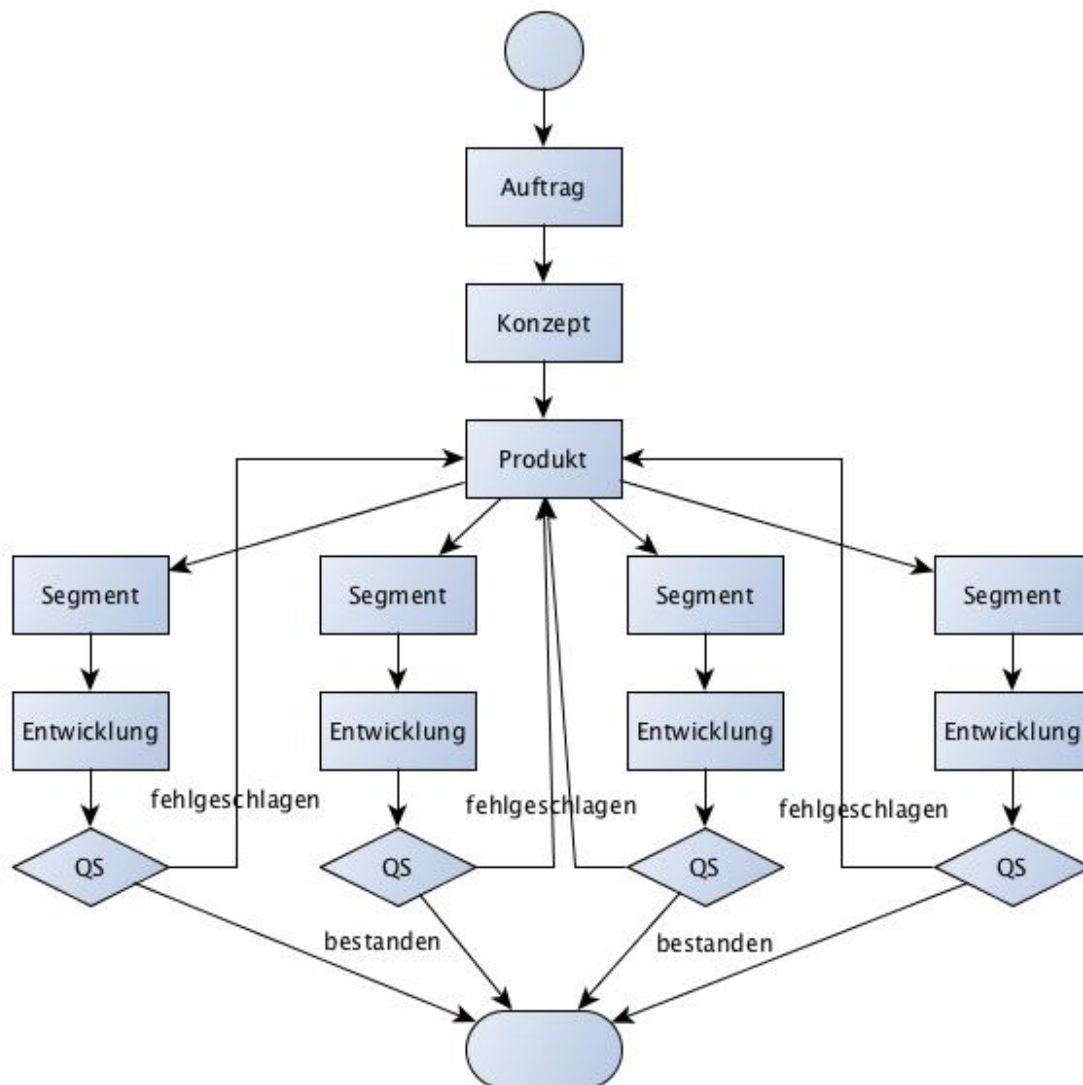


Abbildung 1.1: Vereinfachte Darstellung eines Softwareentwicklungsprozesses

In

Kapitel

TOREMOVE->

KAPITELNUMMER

<-TOREMOVE befrage ich mich ausschließlich mit der Ausformulierung der Aufgabenstellung. Ich ermittle welche Kriterien Notwendig sind für die Durchführung der Evaluation und lege fest welche Wertigkeit die einzelnen Faktoren in Bezug auf die Gesamtbewertung erhalten. Ebenfalls lege ich in diesem Kapitel die Abgrenzungskriterien fest, welche dazu dienen die Bearbeitung der Aufgabe innerhalb eines vordefinierten Rahmens zu halten.

In dem darauf folgenden Kapitel kläre ich alle allgemeinen sowie auch technischen Grundlagen, die Notwendig sind diese Abschlussthesis zu verstehen. Ich werde ausführlich auf verwendete Begriffe eingehen, sowie Begriffe die in dessen Umfeld entstanden sind. Ein weiterer Punkt innerhalb dieses Kapitels ist die Erläuterung technischer Versuchsaufbauten die im Rahmen der Thesis Notwendig waren um die eine Evaluation durchzuführen.

Anmerkung

Aus Gründen der besseren Lesbarkeit wird für alle Personen und Funktionsbezeichnungen durchgängig das generische Maskulinum angewendet und bezieht in gleicher Weise Frauen und Männer ein.

Entwurf

2 Aufgabenstellung

2.1 Problemstellung

Zur Qualitätsprüfung wird ein Testszenario erstellt, welches möglichst alle, oder zumindest einen Großteil der Anforderungen erfüllt. Dieses Szenario wird nun, von Hand, an jedem vorhandenen Testgerät durchgeführt, und dies möglichst immer konstant. Das Ergebnis wird dem Softwareentwickler mitgeteilt, welcher gegebenenfalls die Software anpasst. Dieser Vorgang wiederholt sich solange bis das erwünschte Ergebnis erreicht ist. Bei einer Vielzahl von Testgeräten entsteht das Problem, dass die Testszenarien durch Nachlässigkeit, Unachtsamkeit oder auch Routine nicht immer vollständig durchlaufen werden, was schlussfolgernd zu verminderter Qualität führt.

2.2 Annahmen und Einschränkungen

2.3 Zielsetzung

Das Ziel dieser Arbeit ist es, bestehende Frameworks auf verschiedene Faktoren zu Evaluieren.

2.4 Abgrenzungskriterien

3 Grundlagen

3.1 Begriffsklärung

3.1.1 parallel-synchron

3.1.2 Web-Applikation

3.1.3 Endgeräte

3.2 technischer Aufbau

3.3 Komponenten

3.3.1 Raspberry Pi

3.3.2 Hardware

Entwurf

4 Technologien

4.1 Ghostlab

4.2 NodeJS

4.3 Zombie.js

4.4 W3C Touch Events Extensions

4.5 Phantom Limb

4.6 jQuery UI Touch Punch

4.7 jQuery Touchit

4.8 NPM touchit

5 Lösungsansätze

5.1 Adobe Edge Inspect

5.2 Ghostlab

5.3 Remote Preview

5.4 Browser-Sync

5.5 Eigenes Framework

Entwurf

Entwurf



6 Evaluation

6.1 Adobe Edge Inspect

6.1.1 Einführung

6.1.2 Vorteile

6.1.3 Nachteile

6.2 Ghostlab

6.2.1 Einführung

6.2.2 Vorteile

6.2.3 Nachteile

6.3 Remote Preview

6.3.1 Einführung

6.3.2 Vorteile

6.3.3 Nachteile

6.4 Browser-Sync

6.4.1 Einführung

6.4.2 Vorteile

6.4.3 Nachteile

6.5 Eigenes Framework

6.5.1 Einführung

6.5.2 Systementwurf

Ablaufdiagramm

Klassendiagramm

6.5.3 Vorteile

6.5.4 Nachteile

7 Helpers

7.1 quote

Dies ist ein Zitat.

7.2 longquote

Dies ist ein längeres Zitat.

7.3 fussnote

Dies ist der Text¹

¹Und dies ist die Fußnote dazu.

Index

Ablaufdiagram, 9

Adobe Edge Inspect, 7, 9

Browser-Sync, 7, 9

Framework, 7, 9

Frameworks, 2, 4

Ghostlab, 6, 7, 9

jQuery, 6

Kaskadierungsfehler, 2

Klassendiagramm, 9

NodeJS, 6

NPM, 6

Phantom Limb, 6

Raspberry Pi, 5

Remote Preview, 7, 9

Softwareframeworks, 2

Systementwurf, 9

Testunits, 2

Touchit, 6

touchit, 6

UI Touch Punch, 6

Usecases, 2

W3C Touch Events Extensions, 6

Zombie.js, 6
