Hochschule für Technik Rapperswil

Studienarbeit

Abteilung Informatik

Methode 635 als Cross Plattform App mit Xamarin

Authoren:
Elias Brunner & Oliver Dias

Betreuer:
Prof. Dr. Olaf Zimmermann

26. September 2018

1 Aufgabenstellung

1.1 Ausgangslage

Gerade in der IT-Welt ist das Finden von Lösungen für neu auftretende, aber auch für bekannte Probleme ein wichtiger Bestandteil des Jobs. Durch den Einsatz von Innovationsmethoden kann der Ideenfindung auf die Sprünge geholfen werden. In diesem Bereich finden sich die verschiedensten Ansätze und Methoden; in dieser Studienarbeit soll die Methode 635 verwendet werden. Methode 635 [1] ist eine Kreativitäts- und Brainwriting-Technik, welche die Entwicklung von neuen, ungewöhnlichen Ideen für Problemlösungen in der Gruppe fördert. Die Methode wird im PQM-Modul an der HSR vorgestellt.

Es gibt nach heutigem Kenntnisstand noch keine mobile App, die die Methode 635 unterstützt. Die Motivation für die Studienarbeit besteht darin, eine Cross-Plattform App zu konzipieren und zu implementieren. Dabei sollen moderne Technologien zum Einsatz kommen, welche es den Anwendern ermöglichen, schneller und einfacher eine Lösung für ein Problem zu erarbeiten.

1.2 Ziele der Arbeit und Liefergegenstände

In der Studienarbeit soll die Methode 635 als SmartPhone App umgesetzt werden. Android- und iOS- Support soll durch Verwendung von Xamarin erreicht werden. Es wird erwartet, dass bis zum Ende des Projektes eine lauffähige und getestete Cross-Plattform Applikation umgesetzt wird, welche es Benutzern ermöglicht, die Methode 635 effektiv und effizient auf ihre Probleme anzuwenden.

Damit die App einen Mehrwert gegenüber der Papierversion bietet, soll es z.B. möglich sein, die Anzahl der Teilnehmer variabel zu bestimmen oder verschiedene Medien (Text, Video, Bilder, etc.) zu verwenden bzw. einzubinden. Die persistente Speicherung der bearbeiteten Problemstellungen soll aus Sicht des Kunden einfacher möglich sein als dies mit Papier möglich ist. Ein weiterer Vorteil einer mobilen Anwendung ist, dass Anwender die Methode 635 nutzen können auch wenn sie nicht am selben Ort sind oder die Lösungsvorschläge nicht zur selben Zeit bearbeiten.

Die Vision der Arbeit ist also, die Papierversion für diese Methodik zu funktional und qualitativ zu überbieten. Dabei spielen Erfolgsfaktoren wie einfache und intuitive Bedienung der App und ein unkompliziertes Reporting sowie Robustheit und Stabilität (Bsp. keine Zeit- und Datenverluste) eine wichtige Rolle.

Weitere kritische Erfolgsfaktoren sind:

- Konfigurierbarkeit (z.B. Anzahl Teilnehmer und Schritte) und Erweiterbarkeit (im Hinblick auf Folgearbeiten, die u.U. auch andere Brainstorming Methoden unterstützen)

- sinnvolle Ausnutzung der Smartphone-Fähigkeiten, um einen Mehrwert im Vergleich zur traditionellen, papiergestützten Methode zu erreichen
- Validierung der Konzepte und ihrer Implementierung mit Hilfe von User Tests in mindestens einem Anwendungsbereich (Bsp. Architekturentscheidungen und -optionen).

2 Abstract

Inhaltsverzeichnis

3 Management Summary

3.1 Ausgangslage

3.2 Vorgehen

3.3 Ergebnisse

4 Technischer Bericht

4.1 Einleitung und Übersicht

4.2 Vorstudie

4.2.1 Xamarin.Forms oder Xamarin native

4.2.2 Visual Studio App Center

4.2.3 Backend-Technologie

4.3 Anforderungsspezifikation

4.3.1 Funktionale Anforderungen

4.3.2 Nicht-Funktionale Anforderungen

Beim Thema Nicht-Funktionale Anforderungen halten wir uns an die Standards ISO 9126[?] bzw. dessen Nachfolger ISO 25010[?]. Beide ISO-Normen sind sich sehr ähnlich und liefern eine gute Checkliste für jegliche Art von Systemanforderungen.

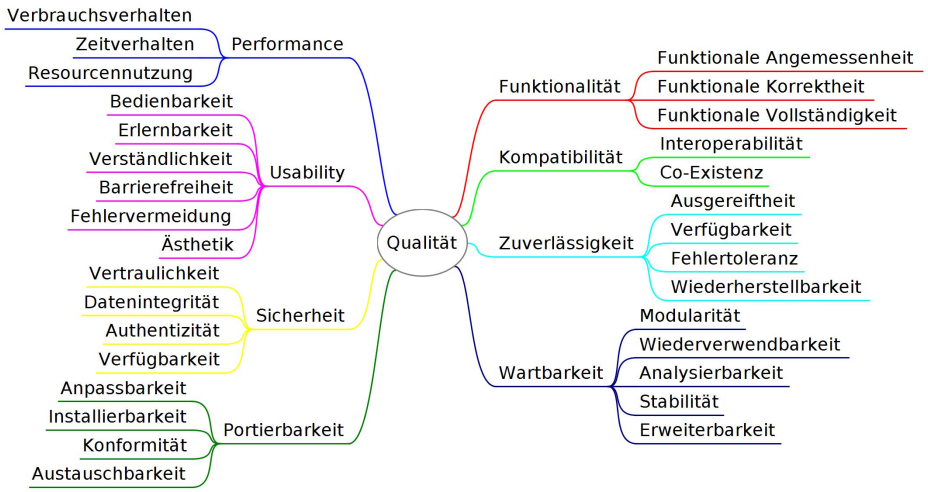

Abbildung 1: Anforderungskategorien nach ISO 25010

Diese Normen sind sehr umfangreich gestaltet. Wir werden uns daher auf die, für uns, wichtigsten Anforderungen konzentrieren.

Ressourcennutzung Die internen Ressourcen wie Kamera, Dateisystem, Batterie usw. müssen sparsam und dürfen nur falls nötig eingesetzt werden.

Bedienbarkeit Die Bedienbarkeit der Applikation muss einfach und intuitiv gestaltet sein.

Fehlervermeidung Dem Benutzer müssen im Falle eines Fehler einfache und verständliche Fehlermeldungen präsentiert werden.

Ästhetik Die Benutzeroberflächen der Applikation dürfen sich nicht wesentlich von den in Xamarin verwendetet Formen unterscheiden.

Vertraulichkeit	Die Daten der Endbenutzer müssen zu jedem Zeitpunkt vertraulich behandelt werden.
Anpassbarkeit	Die Anpassung oder Integration von neuen Brainstorming-Methoden muss gewährleistet sein.
Installierbarkeit	Die Installation der Applikation auf einem Endgerät muss einfach gestaltet sein.
Co-Existenz	Sollte zu einem späteren Zeitpunkt entschieden werden ein Web-Frontend zu programmieren, muss dieses co-existent mit der Xamarin Applikation existieren können.
Wiederherstellbarkeit	Im Falle eines fehlerhaften Features, muss es innerhalb eines Werktages möglich sein, die Applikation wieder auf den alten Stand zurück zu holen und erneut zu deployen.
Wiederverwendbarkeit	Es ist darauf zu achten, dass der geschriebene Code, falls möglich, wiederverwendet wird.
Analysierbarkeit	Es muss möglich sein, die Endnutzer und deren Verhalten mit der Applikation zu analysieren.

4.4 Domainanalyse

4.5 Ergebnisse

4.6 Schlussfolgerungen

4.6.1 Ergebnisbewertung

4.6.2 Ausblick

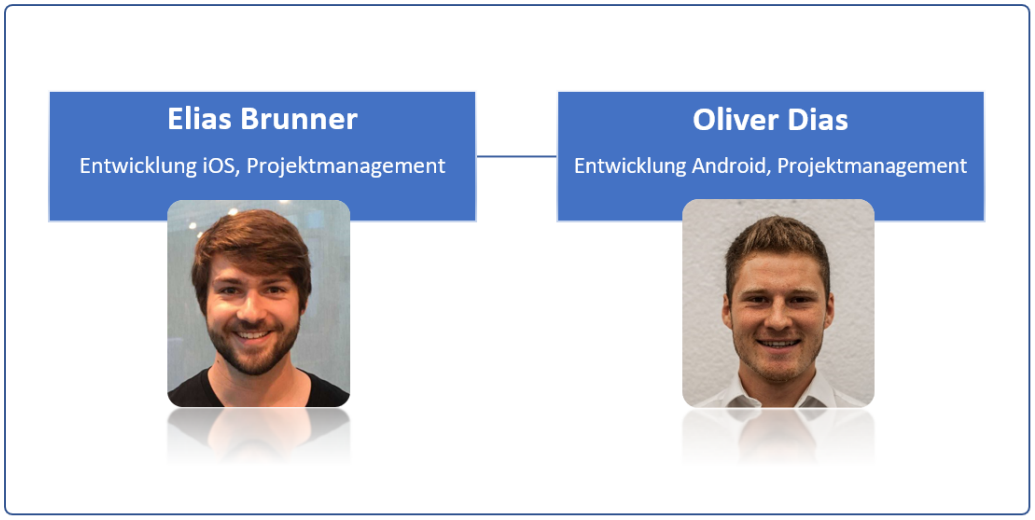

Abbildung 2: Braining Out Of Box Organisation

Ansprechspartner

Im Projekt «Methode 635 als Cross Plattform App mit Xamarin» ist folgender Ansprechpartner vorhanden.

Dozent Prof. Dr. Olaf Zimmermann ist der betreuende Dozent für diese Studienarbeit. Er ist neben der Betreuung auch für die Bewertung des Projekts verantwortlich.

A.3 Projektmanagement

Eine ausführliche Iterationsplanung mit den dazugehörigen Meilensteinen befindet sich auf Jira. Die aufgewendeten Zeiten für ein Issue werden ebenfalls dort erfasst.

Da wir mit einer agilen Entwicklung arbeiten, wird die Planung während dem Projekt laufend aktualisiert und den aktuellen Gegebenheiten angepasst.

Wie in Abbildung **??** zu sehen, ist ein grober Projektablauf ausgearbeitet.

Zur Verwaltung des Codes nutzen wir öffentliche Github Repositories. Die Kommunikation abseits der HSR Anwesenheit erfolgt über einen Whats-App Chat oder alternativ über einen Slack Channel.

Besprechungen

Das Projektteam trifft sich einmal in der Woche das ganze um sich über den aktuellen Stand des Projekts auszutauschen, Fragen zu klären, Probleme anzugehen oder die nächsten Schritte zu planen.

Diese wöchentlichen Besprechungen finden, falls nicht anders vorgesehen, jeden Donnerstagmorgen um 09:00 Uhr statt.

Über jede Besprechung wird Protokoll geführt. Dies mit dem Ziel, die Entscheidungen festzuhalten und Missverständnisse zu vermeiden.

Umgang mit Risiken

Um auch auf unbekannte Risiken vorbereitet zu sein, ist am Ende des Projektes eine Reserve eingeplant. Zudem haben sich alle Teilnehmer bereit erklärt ihr Engagement punktuell zu erhöhen, falls die Situation dies erfordert. Diese Erhöhung sollte jedoch nur Phasenweise sein und in einer folgenden Phase kompensiert werden.

Die häufigsten Risiken wurden mit einer Risikotabelle (Unterkapitel **??**) berücksichtigt, die aktuell gehalten wird und beim Planen in Betracht gezogen wird.

Qualitätsmassnahmen

Das Endprodukt dieses Projekts soll von möglichst hoher Qualität sein. Wie in Tabelle **??** zu sehen ist, treffen wir folgende Massnahmen, um diese Qualität zu erreichen.

Massnahme	**Zeitraum**	**Ziel**
Meeting im Team und mit Betreuer	Jede Woche	Projektstand aufzeigen, allfällige Probleme möglichst früh erkennen.
Code Reviews	Bei jedem Pull Request	Die Qualität des Codes wird durch die Einhaltung der Code Style Guidelines verbessert.

Tabelle 1: Massnahmen

Arbeitspakete

Die gesamte Arbeit ist in Arbeitspakete unterteilt, die auf Jira getrackt sind. Dabei sind relevante Informationen wie die Komplexität, Dauer, der damit verbundene Epic und die Unterteilung in Arbeitskategorie erfasst.

Für die Abschätzung der Komplexität der Arbeiten verwenden wir **Story Points**. Dabei einigen wir uns auf folgendes Schema:

Story Points	Bedeutung
1-3	Niedrige Komplexität
4-6	Mittlere Komplexität
7-9	Komplexe bis sehr komplexe Arbeit

Tabelle 2: Story Points Komplexität

Das Unterteilen in drei Punkte pro Zeile ermöglicht ein genaueres Abschätzen innerhalb der Komplexitätskategorie. Story Points können nicht direkt in zeitlichen Aufwand umgerechnet werden. Für den Aufwand existiert ein separates Feld.

Um die Pakete logisch unterteilen zu können, existieren Arbeitskategorien. Diese werden mit Labels auf den Tickets markiert und können folgende Werte annehmen:

ProjektManagement Alle Aufgaben, die im Zusammenhang mit Projektmanagement stehen, zum Beispiel das Risikomanagement.

Planung Planungsaufgaben. Zum Beispiel steht jede Woche ein Planungsmeeting an, welches dieser Kategorie zugeordnet ist.

Dokumentation Arbeiten an der Dokumentation des Projektes.

Infrastruktur Diejenigen Arbeitspakete, die für die Entwicklung und für den Betrieb des Projekts notwendig sind. Ein Beispiel dafür kann das Einrichten eines Codequalitätstools sein.

Entwicklung Arbeitspakete welche mit der Programmierung der Applikation in Zusammenhang stehen.

Testing Arbeitspakete wie zum Beispiel das Schreiben von Testfällen kann dieser Arbeitskategorie zugewiesen werden.

Eingesetzte Werkzeuge

Um ein gutes Arbeiten zu ermöglichen, stehen viele Tools zur Verfügung, die im Folgenden beschrieben sind. Die primäre Entwicklungsumgebung ist Visual Studio und Visual Studio for Mac.

A.4 Entwicklung

Der Entwicklungscode wird in öffentlichen Github Repositories unter der Organisation **BrainingOutOfBox** gehalten. Für alle einzelnen Teile des Projekts gibt es

ein eigenes Repository.

Doc Dieses Repository enthält alle relevanten Dateien, welche für die Dokumentation von Relevanz sind.

App Dieses Repository enthält den gesamten Code für die Xamarin Applikation.

Vorgehen bei der Entwicklung

Jedes Teammitglied verfügt über eine lokale Kopie der Repositories von Github. Für jede Aufgabe/Issue wird ein eigener Branch erstellt. Darin werden die Änderungen für diese Aufgabe vorgenommen. Die Änderungen sollen mit sinnvollen und präzisen Commit-Notizen festgehalten werden. Um ein Tracking der Änderung möglichst effizient zu gestalten, gilt es möglichst früh, möglichst viel zu commiten.

Code Guidelines

Da Xamarin auf .Net bzw. C# aufbaut, werden die Code Guidelines von .Net verwendet. [**?**]

Builden und testen der App

Für das automatisierte Builden und Testen nach einem Commit wird auf Visual Studio App Center von Microsoft gesetzt. Zum einen ermöglicht es eine einfache Integration von Github und zum anderen bringt es alles mit, um Xamarin Apps automatisch zu builden, testen und deployen.

A.5 Zeitliche Planung

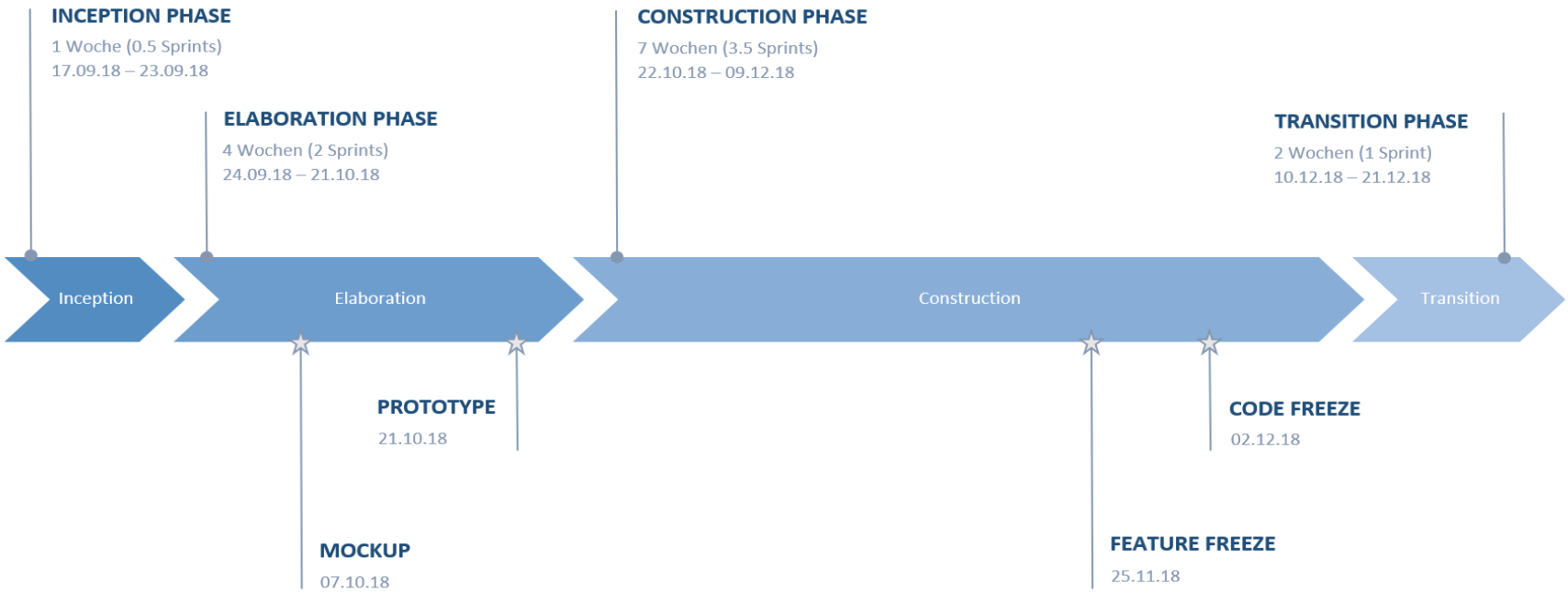

Abbildung 3: Projektplan

A.6 Risikotabelle

Risikomanagement

Projekt:	Methode 635 als Cross Plattform App mit Xamarin
Erstellt am:	21/09/2018
Autor:	Elias Brunner
Gewichteter Schaden:	46,95

Nr	Titel	Beschreibung	max. Schaden [h]	Eintrittswahrscheinlichkeit	Gewichteter Schaden	Vorbeugung	Verhalten beim Eintreten
R1	Anforderungen	Stark wechselnde Anforderungen	20	20%	4	Genaue Definition während der Elaboration und gute Planung der Iterationen	Änderungen einschätzen, überprüfen und eventuell Vornehmen
R2	Managment Tool	Arbeiten mit Jira zeitaufwändiger als erwartet da noch nie verwendet	20	10%	2	Früh genug mit Jira vertraut machen	Zusätzliche interne Schulung und erfahrene Kollegen fragen
R3	Kommunikation	Informationen werden nicht gut genug den Teammitgliedern mitgeteilt	15	15%	2,25	Frühzeitig abklären und genau mitteilen, wer was übernimmt	Zusätzliche Meetings um Ungenauigkeiten abzuklären
R4	Komplexität	Die Komplexität wurde unterschätzt. Der effektive Zeitaufwand übersteigt die Planung um ein Vielfaches	50	25%	12,5	Genaue Abschätzung der Komplexität mittels Story Points und der benötigten Zeit	Rücksprache mit Betreuer über weiteres Vorgehen. Allenfalls Funktionalitätsumfang anpassen, verringern
R5	Schlechtes Zusammenspiel der Komponenten (Technologie Stack)	Der angedachte Technologie Stack kann nicht wie angenommen umgesetzt werden, da inkompatible Komponenten/Packages existieren	16	20%	3,2	Internetanalyse. Gibt es bereits Projekte, die die angedachte Kombination bereits so einsetzen?	Inkompatible Komponenten ersetzen
R6	Entwicklungsumgebung	Kompatibilität der Entwicklungsumgebung oder Kentnisse derselben nicht aussreichend	10	10%	1	Auswahl der Umgebung für alle Mitglieder in Ordnung	Aushilfe bei Problemen, zusätzlich informieren
R7	Qualität	Code Guidelines, Qualitätsmanagment werden nicht eingehalten	20	10%	2	Guidelines einhalten, Tools für Überprüfung verwenden. Kontrolle bei Code-Reviews	Mehr Reviews und Gespräch mit Entwicklern
R8	Dokumentation	Erstellte Arbeiten werden nicht gut genug dokumentiert	20	20%	4	Alles verständlich dokumentieren und kommunizieren	Sensibilisierung der Mitglieder für Dokumentation

R9	Architektur skaliert nicht	Bei vielen Benutzer verhält sich das System sehr langsam und träge	40	20%	8	Genügt Zeit in die Architekturanalyse investieren und bereits bei den ersten Prototypen mehrere Benutzer und höhere Last simulieren	Anpassen der Architektur. Alternativ Ausbau der Hardware Infrastruktur
R10	Know-How	Fehlendes Know-How in den gewählten Programmiersprachen oder Arbeitsumgebungen	30	20%	6	Know-How vertiefen über verwendete Sprachen und Umgebungen	Know-How aufbauen
R11	Schwierige Umsetzung der Wireframes	Die in der Evaluations Phase erstellten Wireframes lassen sich mit Mobiletechnologien nur schwer umsetzen	10	20%	2	Durch die Ausbildung ist den Mitgliedern relativ gut bekannt, wie ein gutes GUI auszusehen hat	Alternative GUIs besprechen und umsetzen
Summe			251		46,95		