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

6. Oktober 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 [1] verwendet werden. Methode 635 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 Was ist Xamarin?

4.3 Was ist die Methode 635?

Die gesamte Beschreibung der Methode 635 wurde von kreativitästechniken.info [1] übernommen.

Die Methode 635 (auch Methode 6-3-5 geschrieben) ist eine Brainwriting-Kreativitätstechnik. Der Name leitet sich aus den drei wesentlichen Eigenschaften der Methode ab: jeweils 6 Teilnehmer erhalten ein Blatt Papier, auf dem sie je 3 Ideen notieren und die Blätter dann insgesamt 5 mal weiterreichen.

Anwendungsgebiete der Methode 635

Die Methode 635 ist eine Variante des Brainwriting. Sie eignet sich besonders für die erste Phase im kreativen Prozess. Dabei werden zunächst Ideen gesammelt ohne dass eine Bewertung stattfindet. Im Idealfall können so in kurzer Zeit 108 Ideen entstehen. Die Aufforderung, bestehende Ideen aufzugreifen und weiterzuentwickeln macht die Methode 635 zu einer konstruktiven Kreativitätstechnik. Gleichzeitig kann die Kreativität durch die strukturierte Form aber auch gebremst werden. In der Praxis entstehen daher oft etwas weniger Ideen.

Vorgehen bei der Methode 635

Zunächst erklärt der Moderator die Regeln der Methode 635, führt die Teilnehmer in das Ausgangsproblem ein und ist im Folgenden verantwortlich für die Zeitmessung. Sobald die Teilnehmer über die Ausgangsfrage oder -problem aufgeklärt sind, startet der Moderator die erste von sechs Runden. In jeder Runde werden die Teilnehmer aufgerufen, die oberste noch freie Zeile, bestehend aus 3 Kästchen, mit ihren Ideen zu füllen. Dabei sollten die Teilnehmer die Ideen der Vorgänger aufgreifen, erweitern und/oder weiterentwickeln ohne die Ideen zunächst zu bewerten. Nach einer festgelegten Zeit von beispielsweise 5 Minuten beendet der Moderator die Runde. Die Teilnehmer reichen ihr Arbeitsblatt im Uhrzeigersinn an ihren Sitznachbarn weiter und eine neue Runde beginnt. Im Idealfall sind nach 6 Runden genau 6*18 = 108 Ideen entstanden. In Realität ist die Anzahl aufgrund von doppelten oder leeren Einträgen wahrscheinlich etwas geringer. Dennoch sollten nun eine Vielzahl von Ideen vorliegen.

Nun kann eine Diskussion, Analyse und Bewertung der Ideen erfolgen.

4.4 Vorstudie

4.4.1 Erste Erfahrungen mit der Methode 635

Bevor wir uns mit den technischen Details der Umsetzung zur Cross-Plattform App auseinander gesetzt haben, spielte jeder der beiden Projektmitgliedern die Methode 635, wie in Kapitel 4.3 beschrieben, mit seiner Familie, Bekannten oder Freunden einmal durch.

Dabei wurden folgende persönliche Erfahrungen und Beobachtungen gemacht:

Diskussion gestartet Schon nach 2-3 Runden konnte beobachtet werden, dass sich spannende Diskussionen zum gestellten Problem entwickelten. Die Teilnehmer mussten sogar angehalten werden die Diskussionen auf dem Papier weiterzuführen, da sonst die Aussagen verloren gehen.

Interessante Methode Die Teilnehmer empfanden die Methode 635 als spannend und interessant. Durch die einzelnen Teilnehmer waren auch verschiedenste Blickwinkel auf das Problem vertreten, was wiederum zu unterschiedlichsten neuen Ideen führe.

Wertung einführen Was allerdings als kleiner Nachteil empfunden wurde, war der Umstand, dass die Methode 635 in der original Version keine Möglichkeit für Wertungen bietet.

Als Mensch bildet man sich während des Lesens der einzelnen Ideen automatisch eine Meinung darüber und wird dadurch stark dazu verleitet, eine wertende Bemerkung statt einer neuen oder angepassten Idee zu schreiben.

Eine mögliche Verbesserung könnte darin bestehen, ein Möglichkeit für Wertungen der Ideen einzuführen. Dies könnte so aussehen, dass man neben dem Text auch ein Label (z.B. Pro oder Contra) und die Idee, welche man bewerten will, auswählen kann.

4.4.2 Xamarin.Forms oder Xamarin native?

4.4.3 Visual Studio App Center

4.4.4 Backend-Technologie

4.5 Anforderungsspezifikation

4.5.1 Funktionale Anforderungen

In den folgenden Kapiteln werden jegliche funktionalen Anforderungen an die Applikation als sogenannte Use-Cases beschrieben. Als Übersicht dient das Use-Case-Diagramm, wie in Abbildung 1 dargestellt.

4.5.1.1 Brief Use-Cases

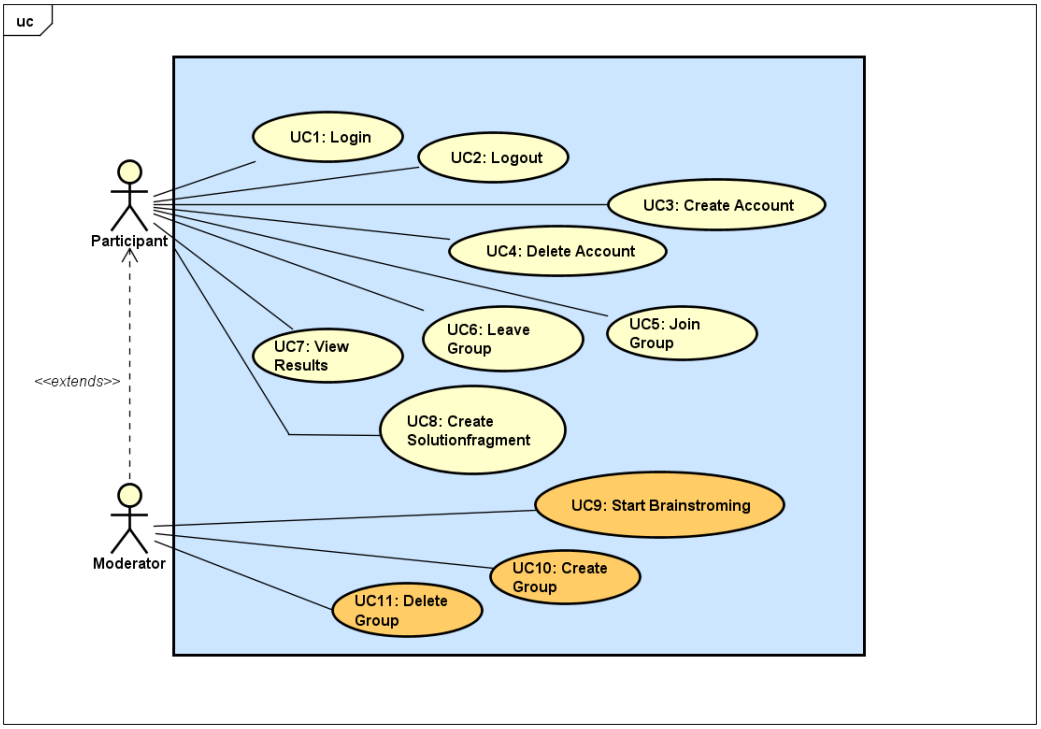

Abbildung 1: Use-Case-Diagramm

Jeder Use-Case hat den nachfolgend kurz (*briefly*) beschriebenen Funktionsumfang. Die Hauptaktivitäten UC7-9 sind am Schluss *fully-dressed* beschrieben.

***UC1:* Login**	Als Participant möchte ich mich mit Benutzernamen und Passwort in das System einloggen können.
***UC2:* Logout**	Als eingeloggter Participant will ich mich ausloggen, sodass das Startfenster wieder erscheint.
***UC3:* Create Account**	Als Participant der Applikation möchte ich mich registrieren können.

***UC4:* Delete Account**	Als Participant will ich meinen erstellten Account wieder löschen können.
***UC5:* Join Group**	Als Participant will ich einer bereits existierenden Gruppe beitreten können.
***UC6:* Leave Group**	Als Participant will eine beigetretene Gruppe verlassen können.
***UC7:* View Results**	Als Participant will ich, nachdem eine Brainstorming Session durchgeführt wurde, das Resultat meiner Gruppe einsehen können.
***UC8:* Create Solutionfragment**	Als Participant will ich während einer Brainstorming Session ein Lösungsfragment erstellen und einreichen können.
***UC9:* Start Brainstorming**	Als Moderator will ich eine Brainstorming Session starten.
***UC10:* Create Group**	Als Moderator will ich eine Gruppe erstellen können.
***UC11:* Delete Group**	Als Moderator will ich eine Gruppe löschen können.

4.5.1.2 Fully-Dressed Use-Cases

Die fully-dressed Use-Cases folgen den im Modul *Software Engineering 1* empfohlenen Punkten.

Use Case 7: View Result	
Primary Actor	Participant
Stakeholders & Interests	Ein Participant wünscht sich eine Übersicht von allen Lösungsfragmenten der Gruppenmitglieder. Er will das Gesamtresultat einsehen und daraus etwas lernen.
Preconditions	Participant existiert im System (UC3), ist eingeloggt (UC1) und einer Gruppe beigetreten (UC5). Zudem hat die Gruppe des Participants alle Runden durchgemacht.

Post Conditions/Success Guarantee	Die Notizen aller Participants werden übersichtlich dargestellt. Das heisst, jede Ausgangsidee ist mit deren Ergänzungen von den verschiedenen Participants ersichtlich.
Main Success Scenario/Basic Flow	1. Der Participant schliesst die letzte Runde als Letzter ab. 2. Das System speichert seine Notizen auf dem Server. 3. Das System zeigt dem Participant eine Meldung, dass die Resultate einsehbar sind. 4. Der Participant clickt auf "Resultate einsehen"(o.Ä.). 5. Das System zeigt dem Participant eine Übersicht mit allen Notizen der Teilnehmer.

Alternative Flows	Alternative Success Scenario 1: 1.a Der Participant schliesst die Runde nicht als Letzter ab. 1.b Das System speichert seine Notizen auf dem Server und zeigt dem Benutzer die verbleibende Zeit an. 1.c Der Participant aktualisiert nach dem Ablauf der Zeit. Alternative Success Scenario 2: 4.a Der Participant clickt nicht auf "Resultate einsehen", sondern navigiert zurück auf den Homescreen. 4.b Das System zeigt dem Participant den Homescreen an. 4.c Der Participant navigiert auf seine Gruppe. 4.d Das System zeigt dem User die Gruppe an. 4.e Der Participant will sich die Resulate dieser Gruppe anzeigen lassen und clickt entsprechenden Button. 4.f Das System zeigt dem Benutzer die Übersicht an.
Frequency of Occurrence	Oft, Kernfunktionalität

Use Case 8: Create Solutionfragment	
Primary Actor	Participant
Stakeholders & Interests	Ein Participant will, dass seine in der aktuellen Runde erfassten Notizen erfasst werden.
Preconditions	Der Participant muss existent (UC3) sowie eingeloggt (UC1) sein. Des Weiteren muss eine Gruppe existieren (UC10), zu welcher er gehört. Die Brainstorming Runde muss ebenfalls gestartet worden sein (UC9).

Post Conditions/Success Guarantee	Vom Participant erfasste Notizen werden erfolgreich auf dem System gespeichert.
Main Success Scenario/Basic Flow	1. Der Participant erfasst während der aktuellen Rundenzeit Notizen. 2. Das System zeigt diese Notizen an. 3. Der Participant beendet frühzeitig die Runde und clickt auf "Notizen abgeben"(o.Ä). 4. Das System persistiert die Notizen und zeigt dem Participant eine Bestätigung an. 5. Der Benutzer kann aktualisieren, um den nächsten Rundenstart nicht zu verpassen.
Alternative Flows	Alternative Success Scenario 1: 3.a Der Participant beendet die Runde nicht manuell. 3.b Das System zeigt dem Participant eine Meldung an, dass die Zeit der Runde abgelaufen ist. Es persistiert die bis zu dem Zeitpunkt erfassten Notizen.
Frequency of Occurrence	Sehr oft, passiert pro Brainstorming-Session mehrmals.

Use Case 9: Start Brainstorming	
Primary Actor	Moderator
Stakeholders & Interests	Ein Moderator will eine neue Brainstorming-Session starten können.

Preconditions	Der Moderator muss existent (UC3) sowie eingeloggt (UC1) sein. Des Weiteren muss eine Gruppe existieren (UC10), welche er erstellt hat. Zudem muss die Gruppe komplett sein (konfigurierte Anzahl an Participants sind in der Gruppe).
Post Conditions/Success Guarantee	Alle Participants der Gruppe können aktualisieren und sehen die gestartete Runde.
Main Success Scenario/Basic Flow	1. Der Moderator clickt auf "Brainstorming starten"(o.Ä) auf der Gruppe, die er erstellt hat. 2. Das System überprüft, dass alle Einstellungen korrekt sind und die korrekte Anzahl an Participants in der Gruppe sind. 3. Das System zeigt dem Moderator an, dass die Session gestartet ist. 4. Der Moderator kann wie die normalen Participants Solutionfragments erfassen.
Alternative Flows	-
Frequency of Occurrence	Oft, zählt zur Kernfunktionalität der Applikation.

4.5.1.3 Abuse-Cases

Um einem Missbrauch der Applikation entgegenzuwirken, sind neben den Use-Cases auch Abuse-Cases definiert. Diese helfen, mit unangebrachtem Inhalt und unangebrachter Verwendung umzugehen.

***AC1:* Unangebrachte Lösungsfragmente** Ein Participant könnte unangebrachte Inhalte in einer Gruppe hinzufügen. Um dies zu verhindern, könnte eine die Applikation um eine Funktion erweitert werden, die es dem Moderator erlaubt, Benutzer auszuschliessen.

4.5.1.4 Sequenzdiagramm

Der Ablauf der Kernlogik ist der Abbildung 2 zu entnehmen. Darin ist der Prozess vom Erstellen des BrainstormingTeams (UC10) bis zum Abschliessen des BrainstormingFinding modelliert.

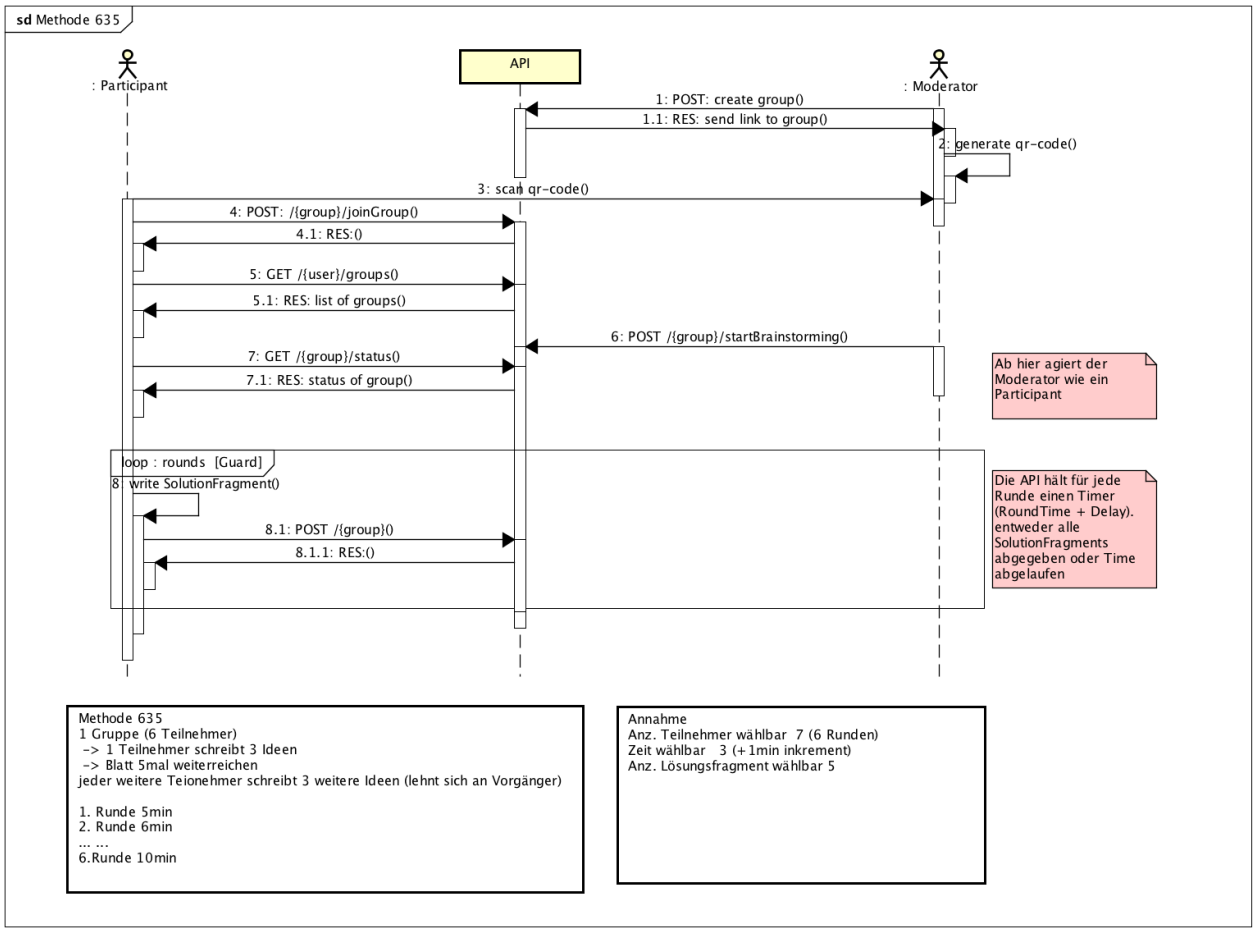

Abbildung 2: Ablauf der Kernlogik

4.5.2 Nicht-Funktionale Anforderungen

Beim Thema Nicht-Funktionale Anforderungen halten wir uns an die Standards ISO 9126[2] bzw. dessen Nachfolger ISO 25010[3]. Beide ISO-Normen sind sich sehr ähnlich und liefern eine gute Checkliste für jegliche Art von Systemanforderungen.

Diese Normen sind sehr umfangreich gestaltet. Wir werden uns daher auf die, für uns, wichtigsten Anforderungen konzentrieren. Um genaue und erfüllbare nicht-funktionale Anforderungen zu definieren, müssen die SMART-Kriterien [5] erfüllt sein.

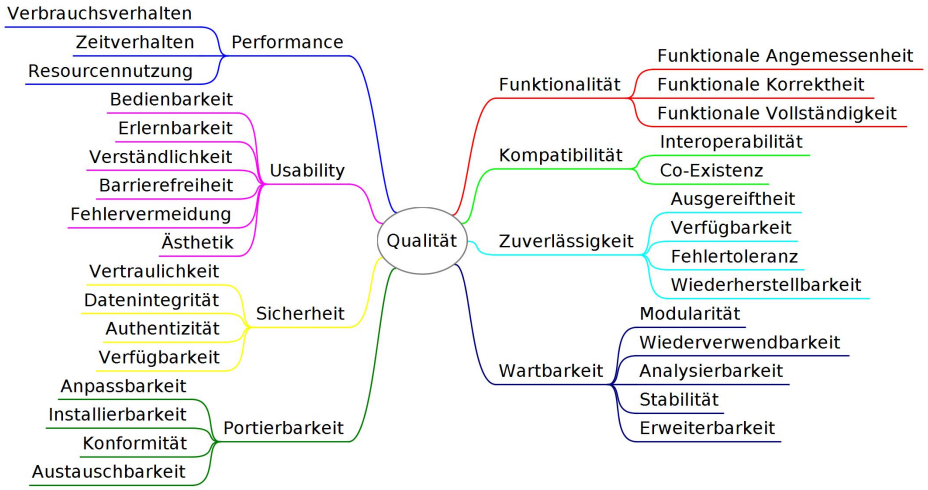

Abbildung 3: Anforderungskategorien nach ISO 25010 [4]

Ressourcennutzung Die internen Ressourcen Kamera, Dateisystem dürfen nur bei effektivem Bedarf benützt werden. Die CPU-Ressourcennutzung darf pro Minute im Bereich von bis zu 40% in Anspruch genommen werden.[1][2]

Bedienbarkeit Wenn eine Aktion länger als 1-2s geht, soll dem User ein Wartesymbol angezeigt werden.

Ästhetik Die Benutzeroberflächen der Applikation sind so gestaltet, dass die Elemente wiedererkennbar sind (Buttons haben gleichen Stil, leere Textfelder haben Platzhalter).

Vertraulichkeit Die Daten einer Brainstorming Session können nur von der zugehörigen Gruppe eingesehen werden.

Anpassbarkeit Die Anpassung bestehender oder Integration neuer Brainstorming-Methoden muss gewährleistet sein.

[1] Referenzsystem Android: Huawei P10 mit Android Version 8.0.0 mit Hisilicon Kirin 960 CPU und 4GB RAM

[2] Referenzsystem iOS: iPhone 6 mit iOS Version 12 mit Dual-core 1.4 GHz Typhoon CPU und 1GB RAM

Installierbarkeit Die Installation der Applikation auf einem Endgerät erfolgt durch das Ausführen eines *.apk oder *.ipa.[3] Dieser Prozess soll unter 1 Minute geschehen.

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 letzten funktionierenden Stand zurück zu holen und erneut zu deployen.

Wiederverwendbarkeit Die Auswertung von 'Duplicated Code' in SonarQube soll im Bereich zwischen 5-10% [6] liegen. Dies deutet auf eine hohe Wiederverwendbarkeit hin, denn ansonsten müsste der Code kopiert werden.

Analysierbarkeit Das Ausführen eines Use-Cases muss durch Analyse von Logfiles erkennbar sein.

[3] Eine Applikation auf einem Android Smartphone hat üblicherweise die Endung *.apk. Beim iPhone bzw. beim iOS haben die einzelnen Applikationen die Endung *.ipa.

4.6 Domainanalyse

4.7 Architekturentscheide

4.7.1 Erste Erfahrungen mit der Methode 635

Wie in Kapitel 4.4.1 beschrieben, tendiert der Mensch dazu, die Ideen der anderen Teilnehmer automatisch zu bewerten. Als mögliche Lösung wurde die Integration einer Bewertungsmöglichkeit beschrieben.

Wir haben uns allerdings darauf geeinigt, dass wir die originale Version, also ohne die Möglichkeit für eine Wertung, als Vorlage nehmen und diese auch so in unserer Cross-Plattform Applikation umsetzen.

Die Integration einer Bewertungsmöglichkeit wird als optionales Feature angesehen und lediglich bei genügend Restzeit im Projekt umgesetzt.

4.7.2 Xamarin.Forms oder Xamarin native

4.7.3 Backend-Technologie

4.7.4 Methode 635 als verteiltes System

Wir haben uns auch überlegt, die Cross-Plattform Applikation d.h. vor allem die Kommunikation zwischen den einzelnen Teilnehmer, als verteiltes System zu konzipieren.

Prof. Thomas Bocek, Professor für verteilte System an der Hochschule Rapperswil, hat uns allerdings davon abgeraten. Ein verteiltes System sei immer komplexer und komplizierter als ein Server/Client System. Für diese geringe Anzahl von Teilnehmern, welche prinzipiell nur Messages austauschen, lohnt es sich nicht ein verteiltes System zu bauen.

Daher haben wir uns für eine klassische Server/Client Architektur entschieden.

4.7.5 Kommunikation zwischen Server und App

Die Kommunikation zwischen dem zentralen Server und den Clients, also den Cross-Plattform Applikationen in unserem Fall, ist von grosser Bedeutung. Da die Clients meist hinter einer NAT [7] agieren, müssen jegliche Anfragen an den Server von den Clients kommen.

Dazu ein kleines Beispiel: Wenn ein Teilnehmer seine Ideen aufschreibt, fragt die Applikation im Hintergrund in regelmässigen Abständen den Server nach der verbleibenden Zeit für diese Runde ab.

Auch nach der Abgabe der aufgeschriebenen Ideen an den Server, muss der Teilnehmer warten, bis er die Ideen bzw. das Blatt seines Nachbarn bekommt. Dafür muss die Applikation immer wieder den Server fragen, ob der Nachbar überhaupt sein Blatt bzw. seine Ideen abgegeben hat. Denn vorher kann der Teilnehmer auch nicht weitermachen.

Wir haben uns daher für das stetige Abfragen von Informationen (Pulling) entschieden, da es eine einfache Variante darstellt. Zwar werden dadurch vermeidbare Requests an den Server gesendet, da aber die Anzahl an Teilnehmer bzw. die Ressourcennutzung des Backends in einem vertretbaren Rahmen liegt, ist diese Pulling-Variante völlig in Ordnung.

Andere Varianten der Kommunikation, wie Webhooks oder Websockets werden daher nicht weiter verfolgt.

4.8 Ergebnisse

4.9 Schlussfolgerungen

4.9.1 Ergebnisbewertung

4.9.2 Ausblick

Literatur

[1] Kreativitätstechniken, "635-methode." https://kreativitätstechniken.info/6-3-5-methode/, Oktober 2011. Accessed on 2018-09-18.

[2] Wikipedia, "Iso 9126." https://de.wikipedia.org/wiki/ISO/IEC9126, Mai 2018. Accessed on 2018-09-25.

[3] Johner-Institut, "Iso 25010 und iso 9126." https://www.johner-institut.de/blog/iec-62304-medizinische-software/iso-9126-und-iso-25010/, August 2015. Accessed on 2018-09-25.

[4] M. Kops, "Qualität, funktionale und nichtfunktionale anforderungen in der software-entwicklung." https://blog.seibert-media.net/blog/2018/05/14/qualitaet-funktionale-und-nichtfunktionale-anforderungen-in-der-software-entwicklung/, Mai 2018. Accessed on 2018-09-25.

[5] Wikipedia, "Smart criteria." https://en.wikipedia.org/wiki/SMART_criteria, August 2018. Accessed on 2018-09-27.

[6] SolidSourceIt, "Does source code duplication matter?." https://solidsourceit.wordpress.com/2012/08/03/does-source-code-duplication-matter/, August 2012. Accessed on 2018-10-06.

[7] A. Donner, "Was ist nat (network address translation)?." https://www.ip-insider.de/was-ist-nat-network-address-translation-a-663954/, August 2017. Accessed on 2018-10-06.

[8] Microsoft, "Framework design guidelines." https://docs.microsoft.com/en-us/dotnet/standard/design-guidelines/, Dezember 2017. Accessed on 2018-09-18.

A Projektplan

Zweck dieses Dokuments

Dieses Dokument beschreibt den Projektplan des Projekts ≪Methode 635 als Cross Plattform App mit Xamarin≫. Es beinhaltet die Planung, die Organisation sowie weitere Aspekte und liefert damit eine Übersicht über das Projekt. Es dient daher als Grundlage für den Verlauf des Projekts.

Gültigkeitsbereich

Der Gültigkeitsbereich erstreckt sich über die gesamte Dauer des Projekts. Der Zeitraum geht vom 17. September 2018 bis zum 21. Dezember 2018. Das Projekt findet im Rahmen des Moduls ≪Studienarbeit≫ im Herbstsemester 2018 statt.

Verweise

An dieser Stelle ist noch zu anzumerken, dass einzelne Textstellen von eigenen, älteren Projektplänen verwendet wurden. Dabei handelt es sich um Projektpläne von Engineering-Projekten oder Studienarbeiten.

A.1 Projektziel

Die Motivation dieser Studienarbeit besteht darin, eine Cross-Platform App zu programmieren, welche die Methode 635 [1] als mobile App für Android und iOS umsetzt. Dabei sollen moderne Technologien zum Einsatz kommen, welche es den Anwendern ermöglichen schneller und einfacher eine Lösung für ein Problem zu erarbeiten.

Mehr zum Thema Projektziel ist dem Kapitel 1 zu entnehmen.

Einschränkungen

Das Projekt ist auf die Dauer des Herbstsemester 2018 begrenzt (bis 21. Dezember 2018). Zudem sollte das Projekt mit ungefähr 240 Arbeitsstunden (gesamthaft 480 Stunden) realisiert werden können. Bleibt am Ende Zeit übrig, werden optionale Features implementiert.

A.2 Projektorganisation

In unserem Projekt arbeiten wir in einer flachen Organisationsstruktur, wobei die wesentlichen Entscheide im ganzen Projektteam und/oder mit dem Dozenten an den wöchentlichen Besprechungen getroffen werden. An den Besprechungen getroffene Entscheidungen werden in Protokollen dokumentiert. Die Projektmitglieder sind innerhalb des Teams gleichgestellt.

Organisationsstruktur

Die Projektmitglieder sowie deren Verantwortung sind der Abbildung 4 zu entnehmen.

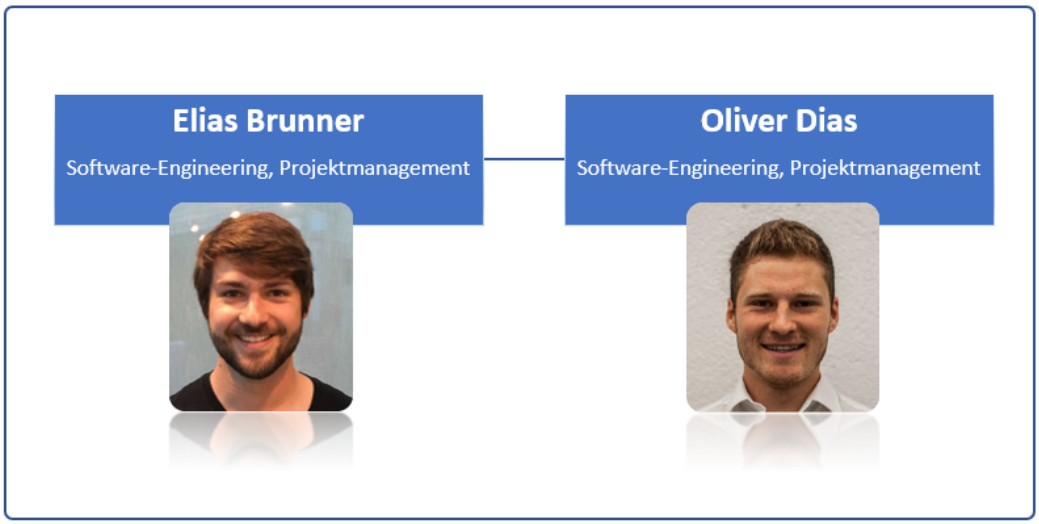

Abbildung 4: Methode 635 Organisation

Ansprechspartner

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

Betreuer 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 des Projekts laufend aktualisiert und den aktuellen Gegebenheiten angepasst.

Ein grober Projektablauf ist in Abbildung 5 ersichtlich.

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 beide Teilnehmer bereit erklärt ihr Engagement punktuell zu erhöhen, falls die Situation dies erfordert. Diese Erhöhung sollte jedoch nur Phasenweise erfolgen und in einer folgenden Phase kompensiert werden.

Die häufigsten Risiken wurden mit einer Risikotabelle (Unterkapitel A.6) 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 4 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 4: 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 5: 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.

Design Hierzu zählen Arbeitspakete wie das Ausarbeiten von Benutzeroberflächen.

Analyse Typische Analyseaufgaben ist zum Beispiel das Recherchieren für bestimmte Frameworks, Bibliotheken, etc.

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.

> "Commit early and commit often"

Dies wurde uns so in den Mudulen SE1 und SE2 vermittelt.

Code Guidelines

Da Xamarin auf .Net bzw. C# aufbaut, werden die Code Guidelines von .Net verwendet. [8]

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

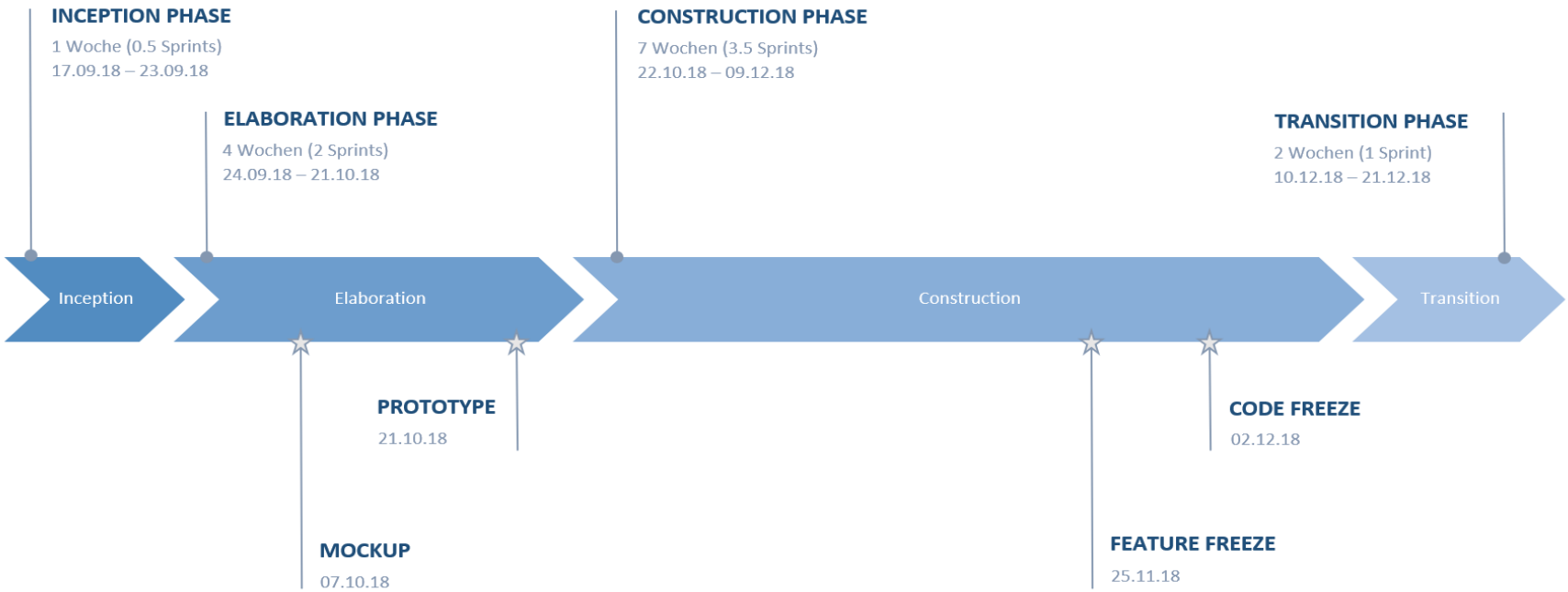

Abbildung 5: Projektplan

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		

B Eigenständigkeitserklärung