# MASTERARBEIT

## CROSS-PLATTFORM APPLIKATION IM BUSINESS BEREICH

ausgeführt am


Studiengang

Informationstechnologien und Wirtschaftsinformatik

Von: Dominik Nehl

Personenkennzeichen: 1610320023

Graz, am 14. Dezember 2017 .....

Unterschrift

# EHRENWÖRTLICHE ERKLÄRUNG

Ich erkläre ehrenwörtlich, dass ich die vorliegende Arbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen nicht benützt und die benutzten Quellen wörtlich zitiert sowie inhaltlich entnommene Stellen als solche kenntlich gemacht habe.

.....

Unterschrift

# DANKSAGUNG

An dieser Stelle möchte ich allen Menschen meine Anerkennung aussprechen, die mich während meiner Studienzeit und vor allem bei der Erstellung dieser Arbeit begleitet haben.

Allen voran möchte ich mich bei meinen Eltern und Freunden, die mich stets unterstützt haben bedanken. Durch sie konnte ich immer neue Kraft schöpfen, um diese Herausforderung meistern zu können.

Für die sehr gute Betreuung während der Erstellung dieser Arbeit geht mein Dank an Herrn Dipl.-Ing. Karl Blümlinger. Durch seine Ratschläge und Hilfestellungen hat er mir bei der Erstellung dieser Arbeit sehr geholfen.

# KURZFASSUNG

Die vorliegende Arbeit beschäftigt sich mit Cross-Plattform-Applikationen im Business Bereich. Es wird der Frage nachgegangen, ob die Effizienz durch die plattformunabhängige Softwareentwicklung gesteigert werden kann. Ziel dieser Arbeit ist es zu klären, in wie weit Experten und Expertinnen bereits Erfahrung mit Technologien sammeln konnten, die sich auf plattformunabhängige Softwareentwicklung spezialisiert haben. Damit herausgefunden werden kann ob die Effizienz gesteigert werden kann, beziehungsweise ob ExpertInnen in diesem Bereich herangezogen werden kann, wird zu Beginn dieser Arbeit eine Einführung in die mobilen Applikationen gemacht und gängige Begriffe näher beschrieben. Dabei wird eingegrenzt, was genau unter einem mobilen Gerät verstanden wird, damit in weiterer Folge die Zielgruppe festgelegt werden kann.

Anschließend werden dem Leser beziehungsweise der Leserin, die beliebtesten Betriebssystem die auf dem Markt existieren vorgestellt und näher beschrieben. Zum Abschluss dieses Kapitel werden die einzelnen Plattformen gegenübergestellt, damit herausgefunden werden kann, welche Vor- beziehungsweise Nachteile existieren.

Nachdem die einzelnen Plattformen vorgestellt wurden, wird auf die plattformunabhängige Softwareentwicklung näher eingegangen. Dabei werden zuerst Begriffe, die im Zusammenhang mit mobiler Entwicklung fallen definiert. Im Anschluss erfolgt eine Vorstellung verschiedener Technologien, die derzeit existieren.

Im empirischen Teil wird der Prototyp der im Zuge dieser Arbeit entwickelt wurde, näher beschrieben. Dafür werden zu Beginn die Anforderungen, die die App erfüllen muss vorgestellt. In weiterer Folge wird dem Lesendem die gewählte Technologie vorgestellt und beschrieben, warum sich der Autor dieser Arbeit für dieses Framework entschieden hat.

Zu Letzt wird im empirischen Teil berichtet werden, welche Technik bei den Experteninterviews zum Einsatz gekommen ist und wie der Aufbau aussieht. Abschließend wird vom Autor dieser Arbeit ein Erfahrungsbericht, über die Erfahrungen die beim Entwickeln des Prototyps gesammelt werden konnten, geschrieben.

Im fünften und letzten Kapitel wird über die Beantwortung der Forschungsfrage berichtet und ein kurzer Ausblick gegeben welche Schritte als nächstes durchgeführt werden können, damit die Ergebnisse gefestigt werden können.

# ABSTRACT

The aim of this thesis is to find out if an increase in efficiency is possible during a cross-platform development. At the beginning of this work, the term mobile apps will be defined. After the definition the most common operating systems will be presented. At the end of this chapter, the individual platforms are compared, so that it can be found out which advantages or disadvantages exist.

The next section introduces cross-platform software development and the author will be describe the terms of this technology. In the next step the most common framework will be presented and at the end, the platforms will be summarized.

The empirical part describes the prototype which was developed during this thesis. At the beginning the requirements, which were put on the application, will be described. Subsequently, the reader will be introduced to the chosen technology and the author will describe why this framework was chosen. Finally, it is written about how the interview process was and which technique was used for these interviews. After this description the author wrote about the difference between the quantitative and qualitative interviews and about the evaluation. At the end of this chapter the author wrote a field report about the development of the prototype.

The fifth and final chapter writes about answering the research question and the author gives a short review about this thesis. This review is about what steps could be taken next to consolidate the results.

# INHALTSVERZEICHNIS


**1 EINLEITUNG ..... 7**

1.1 Problemstellung ..... 7
1.2 Zielsetzung ..... 8
1.3 Vorgehensweise ..... 8
1.4 Aufbau ..... 9

**2 EINFÜHRUNG IN MOBILE APPS ..... 11**

2.1 Begriffsdefinition ..... 11
2.1.1 Mobile Endgeräte ..... 11
2.1.2 Mobile Applikationen ..... 13
2.1.3 Mobile-Device-Management ..... 17

2.2 Betriebssysteme ..... 17
2.2.1 Android ..... 18
2.2.2 iOS ..... 19
2.2.3 Universal Windows Plattform ..... 20
2.2.4 Vergleich der Betriebssysteme ..... 20

**3 EINFÜHRUNG IN PLATTFORMUNABHÄNGIGE SW-ENTWICKLUNG ..... 22**

3.1 Begriffsdefinitionen ..... 22
3.1.1 Cross-Plattform-Applikationen ..... 22
3.1.2 Hybrid-Applikationen ..... 23
3.1.3 Single-Page-Applikationen ..... 23

3.2 Technologien ..... 24
3.2.1 Xamarin ..... 24
3.2.2 Unity ..... 25
3.2.3 Phonegap ..... 26
3.2.4 Framework7 ..... 26
3.2.5 Mobile Angluar UI ..... 27
3.2.6 Zusammenfassung ..... 28

**4 EMPIRISCHER TEIL ..... 29**

4.1 Entwicklung des Prototyps ..... 29
4.1.1 Anforderungen ..... 29
4.1.2 Auswahl der gewählten Technologie und des Frameworks ..... 30
4.1.3 Umsetzung der Anforderungen ..... 31
4.1.4 Bewertung des Prototyps ..... 39

4.2 Experteninterview ..... 40
4.2.1 Recherche der einzelnen Interviewarten ..... 41
4.2.2 Konzept und Aufbau ..... 42
4.2.3 Quantitative Auswertung ..... 46
4.2.4 Qualitative Auswertung der Ergebnisse ..... 48

4.3 Erfahrungsbericht ..... 51

**5 ABSCHLUSS DER ARBEIT ..... 60**

5.1 Beantwortung der Forschungsfrage ..... 60
5.2 Ausblick ..... 61

**ANHANG A - INTERVIEWS ..... 63**

**ABKÜRZUNGSVERZEICHNIS ..... 74**

**ABBILDUNGSVERZEICHNIS ..... 75**

**TABELLENVERZEICHNIS ..... 77**

**LISTINGS ..... 78**

**LITERATURVERZEICHNIS ..... 79**

# 1 EINLEITUNG

Für die Feuerwehr wird es immer schwieriger die ideale Route zu einem Einsatz zu bestimmen, da sich zum einen die städtischen Siedlungsformen ausbreiten und zum anderen durch die Gemeindezusammenlegung große Gemeinden entstehen. Somit wird es für Einsatzkräfte immer schwieriger die bestmögliche Route zum Einsatzort zu finden. Wie stark die Urbanisierung auch in Bezirken, die in Stadtnähe sind, vorangeht, zeigt die Gemeinde Seiersberg-Pirka. So zählte die Gemeinde im Jahre 1961 4697 Einwohner. Im Jahr 2017 sind es bereits 11.051 Einwohner. Die folgende Statistik zeigt wie stark das Wachstum der Gemeinde ist (bevoelerkung.at, 2017).

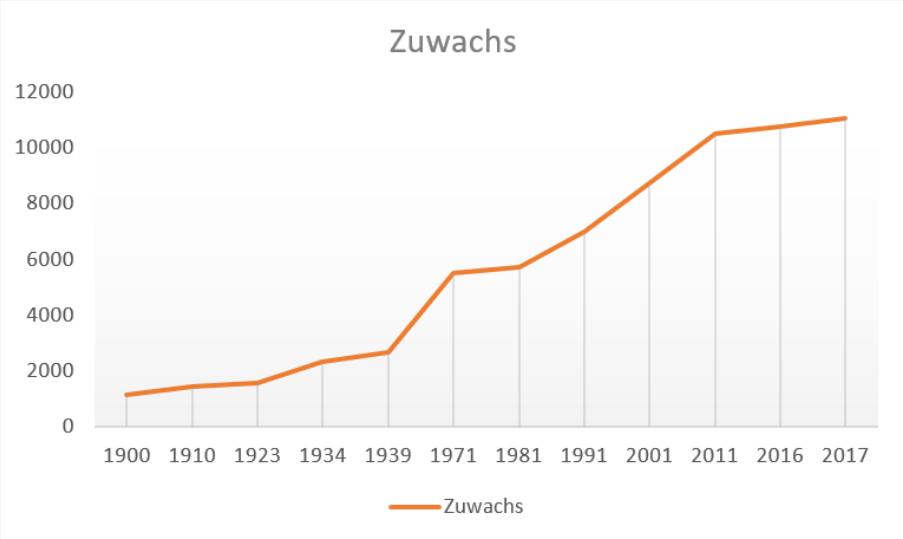



*Abbildung 1-1: Zuwachs in Seiersberg (bevoelerkung.at, 2017)*

Aus diesem Grund haben sich Kameraden der Freiwilligen Feuerwehr Seiersberg dafür entschieden diese Problematik mittels einer Softwareunterstützung zu lösen. Es soll eine Lösung geschaffen werden, die eine Unterstützung zum Finden des idealen Einsatzweges bereitstellen kann. Da solch eine Software nicht durch Abhängigkeiten wie zum Beispiel ein Betriebssystem eingeschränkt sein soll, soll die Applikation für alle gängigen Plattformen zur Verfügung gestellt werden. Diese Anforderung bringt bereits die erste große Hürde mit sich. Es soll eine plattformunabhängige Software entwickelt werden, die den Kameraden die Möglichkeit bieten kann zum Einsatz zu navigieren. Die folgende Arbeit setzt genau bei dieser Herausforderung an und es wird eine Applikation entwickelt werden, die zum einen plattformunabhängig ist und zum anderen die Anforderungen der Freiwilligen Feuerwehr erfüllt.

## 1.1 Problemstellung

Für SoftwareentwicklerInnen wird es immer schwieriger eine Applikation zu entwickeln, die auf jeglichen Plattformen zur gleichen Zeit die gleichen Anforderungen erfüllt. Aus diesem Grund ist

es meist nicht mehr möglich, dass ein Softwareentwickler bzw. eine Softwareentwicklerin zum Beispiel zur gleichen Zeit ein Programm entwickelt, dass auf einem Android- und auf einem iOS-Gerät funktioniert. Daraus hat sich der Workflow etabliert, dass zuerst eine Version für ein Betriebssystem entwickelt und in weiterer Folge dieselbe Software für andere gängige Plattformen programmiert wird. Dadurch haben oft Anwendungen auf den unterschiedlichen Betriebssystemen unterschiedliche Funktionen. Somit ergibt sich die Problemstellung, dass es immer schwieriger wird als kleines Unternehmen erfolgreiche Applikationen zu entwickeln, da eine gewisse Reichweite nur erreicht werden kann, wenn die Software für gängige Plattformen zur Verfügung gestellt wird.

Daraus hat sich die folgende Forschungsfrage für diese Arbeit ergeben:

*Inwieweit ist effiziente Entwicklung von Business Applikation mit aktuellen Cross Plattform-Technologien möglich?*

## 1.2 Zielsetzung

Damit herausgefunden werden kann, ob die Entwicklung einer Cross-Plattform-Applikation effizienter ist als die klassische native Programmierung, wird im Zuge dieser Arbeit zuerst ein Prototyp entwickelt werden, damit Erfahrungen in diesem Bereich gesammelt werden können. Diese Erfahrungen werden in einem Abschlussbericht zusammengefasst und mittels Literatur gefestigt. Neben diesem Bericht wird mit ExpertInnen eine Befragung durchgeführt, damit diese berichten können, welche Erfahrungen sie bereits gesammelt haben. Durch die Bewertung dieser Befragung soll herausgefunden werden in wie weit EntwicklerInnen bereits plattformunabhängige Technologien eingesetzt haben und welche Erfahrungen die Befragten bereits damit machen konnten.

## 1.3 Vorgehensweise

Um das definierte Ziel dieser Arbeit zu erreichen, wird folgendermaßen vorgegangen:

1. Erarbeitung theoretischer Grundlagen zu mobilen Apps
2. Einführung in die plattformunabhängige Softwareentwicklung
3. Entwicklung des Prototyps
4. Interviews und Erfahrungsbericht
5. Zusammenfassung

Zu Beginn erfolgt eine Einarbeitung in die Grundlagen von mobilen Applikationen. Hierzu werden übliche Begrifflichkeiten voneinander abgegrenzt und die gängigsten Betriebssysteme näher vorgestellt. Nachdem Plattformen näher beschrieben worden sind, werden diese miteinander verglichen damit mögliche Entscheidungshilfen herausgefiltert werden können, die bei der Entscheidung der Zielplattform eine Rolle spielen könnten.

Im nächsten Schritt wird die plattformunabhängige Softwareentwicklung vorgestellt und die am weit verbreitetsten Varianten, die derzeit am Markt vorzufinden sind, näher beschrieben. Neben der Definition werden bekannte Vor- bzw. Nachteile, die bei solch einer Entwicklung auftreten beschrieben.

Basierend auf den Grundlagen der plattformunabhängigen Softwareentwicklung und den Ergebnissen der Experteninterviews wird jene Technologie gewählt, mit der die Anforderungen der Freiwilligen Feuerwehr umgesetzt werden kann. Danach erfolgt die Umsetzung des Prototyps der mobilen Applikation. Hierbei wird versucht die zuvor identifizierten Anforderungen gezielt im Prototyp umzusetzen. Dabei wird gezielt versucht die Testapplikation ohne Abhängigkeiten zu einer bestimmten Plattform zu entwickeln.

Nach Abschluss der Entwicklung wird eine Befragung mit Experten und ExpertInnen durchgeführt werden, damit herausgefunden wird, welche Erfahrungen andere SoftwareentwicklerInnen in diesem Bereich sammeln konnten. Des Weiteren wird in diesem Abschnitt ein Erfahrungsbericht basierend auf dem im Zuge dieser Arbeit erstellten Prototyps erstellt und die darin getätigten Aussagen mit Fachliteratur gefestigt.

Am Ende wird dem Lesendem ein kurzer Überblick über die Vorgehensweise geschildert und die Ergebnisse, die durch die Befragung und des Erfahrungsberichtes gesammelt werden konnten zusammengefasst. Mittels diesen Ergebnissen wird versucht zum einen die Forschungsfrage zu beantworten und zum anderen wird auf die Hypothesen eingegangen. Nach dieser Analyse wird noch ein kurzer Ausblick gegeben, welche weiteren Studien durchgeführt werden könnten, damit die Ergebnisse bestätigt werden.

## 1.4 Aufbau

In diesem Abschnitt wird dem Lesendem der Aufbau der vorliegenden Arbeit kurz beschrieben und grafisch dargestellt. Durch die gewählte Vorgehensweise, die im Abschnitt 1.3 näher beschrieben wurde, hat sich folgende Einteilung in Kapitel ergeben: Einleitung, Einführung in mobile Apps, Einführung in plattformunabhängige SW-Entwicklung, Empirischer Teil, Abschluss der Arbeit

Das Kapitel **Einleitung** befasst sich mit der Problemstellung und der Zielsetzung und erläutert die Vorgehensweise die für die folgende Arbeit angewendet wird. Des Weiteren wird zu Beginn kurz auf die Problemstellung, für die die Software als Lösung dienen soll, näher beschrieben.

Das zweite Kapitel, welches den Namen **Einführung in mobile Apps trägt**, dient dazu dem Leser beziehungsweise der Leserin die Begriffe mobile Apps und mobile Endgeräte näher zu beschreiben und die Begrifflichkeiten die damit in Verbindung stehen zu definieren. Neben der Begriffserklärung werden Trends von mobilen Applikationen analysiert und beschrieben. Zum Abschluss werden dem Lesenden die gängigsten Betriebssysteme, die am Markt existieren erklärt und grafisch dargestellt wie die Verteilung der einzelnen Plattformen aussieht.

Im Kapitel **Einführung in die plattformunabhängige SW-Entwicklung** wird zu Beginn definiert in welchen Bezug die Unabhängigkeit des Betriebssystems gemeint ist und welche verschiedenen Ansätze auf dem Markt existieren. Am Ende dieses Kapitels werden die vorgestellten Technologien gegenübergestellt und darüber berichtet, welche Technologie für die Prototypen eingesetzt wird.

Im **empirischen Teil** der Arbeit werden zu Beginn die Anforderungen, die von der Freiwilligen Feuerwehr gestellt wurden, definiert und näher beschrieben. Im nächsten Schritt wird der Verfasser dieser Arbeit näher auf die Implementierung des Prototypen eingehen und darüber Schreiben in wie weit das Ziel der plattformunabhängigen Softwareentwicklung erreicht wurde.

Abschließend wird im Kapitel **Abschluss der Arbeit** auf die Beantwortung der Forschungsfrage eingegangen und dem Lesendem ein kurzer Überblick über die Vorgehensweise gegeben. Außerdem wird über einen Ausblick berichtet. Dabei wird kurz beschrieben, wie die Erkenntnisse die durch diese Arbeit gewonnen werden gefestigt werden können.

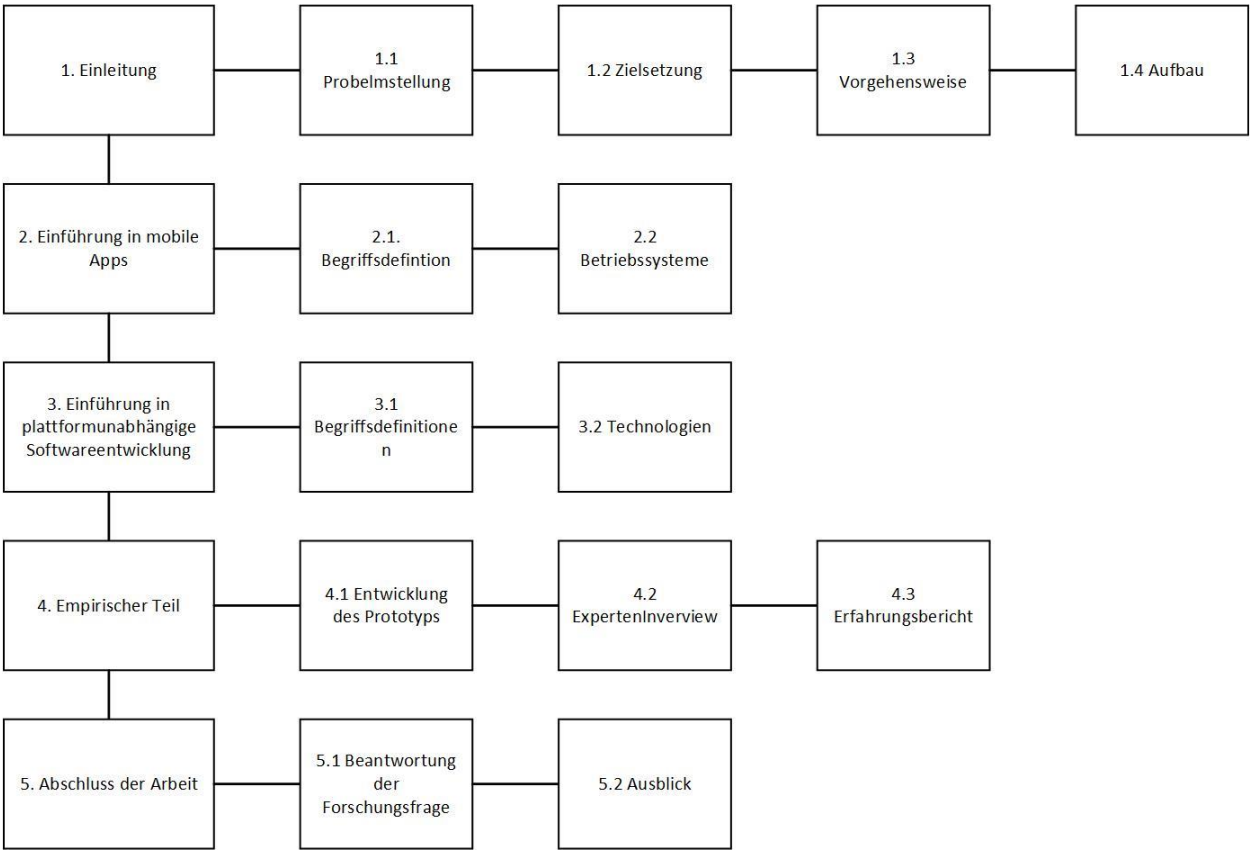



*Abbildung 1-2: Aufbau und Gliederung der Arbeit*

# 2 EINFÜHRUNG IN MOBILE APPS

Der folgende Abschnitt beschäftigt sich zuerst mit einer allgemeinen Einführung in mobile Applikationen. Im Rahmen dieser Einführung erfolgt zu Beginn eine Definition von Begriffen, die häufig im Zusammenhang mit mobilen Apps verwendet werden. Anschließend wird auf die verschiedenen Plattformen eingegangen, auf denen mobile Anwendungen eingesetzt werden. Die Betriebssysteme die am Markt angeboten werden, werden zu Beginn vorgestellt und in weiterer Folge miteinander verglichen und es wird aufgezeigt wie sich die Systeme in den letzten Jahren entwickelt haben und wie die Verteilung auf die AnwenderInnen aussieht.

## 2.1 Begriffsdefinition

In der Literatur werden immer wieder zahlreiche Begriffe verwendet. In diesem Teil des Kapitels wird dem Lesendem eine Übersicht über die gängigsten Fachtermini gegeben. Zudem wird erläutert welche Bedeutungen die einzelnen Begrifflichkeiten mit sich bringen. Die Definition der folgenden Begriffe erfolgt auf Basis facheinschlägiger Literatur:

- Mobile Endgeräte
- Mobile Applikationen
- Mobile-Device-Management

### 2.1.1 Mobile Endgeräte

Bei Mobilgeräten beziehungsweise bei mobilen Endgeräten handelt es sich um Geräte, die aufgrund ihrer Größe und ihres Gewichts ohne größerer körperlichen Anstrengung mitgeführt werden können und somit mobil einsetzbar sind. Es handelt sich dabei um elektronische Geräte die zur netzunabhängigen Daten-, Sprach- und Bildkommunikation eingesetzt werden können. Des Weiteren können mobile Endgeräte auch zur Navigation dienen (Lanzer, 2012).

Bereits im Jahre 1999 hat das Durlacher Institut eine Studie veröffentlicht, die analysiert hat welche Eigenschaften ein Gerät für mobile Kommunikation mit sich bringen muss. In dieser Studie wurden gesamt sieben Attribute identifiziert, die ein mobiles Endgerät mit sich bringen muss. Vier dieser sieben Eigenschaften (Sicherheit, Bequemlichkeit, Verfügbarkeit und Personalisierung) sollen eher eine Erweiterung der klassischen Kommunikationseigenschaften sein. Die restlichen drei Eigenschaften (Lokalisierbarkeit, Erreichbarkeit und Ortsunabhängigkeit) sind ausschließlich im mobilen Umfeld relevant und bieten dementsprechend die Grundlage zur Klassifikation von mobilen Endgeräten.

Wenn, wie in der folgenden Grafik gezeigt, die drei relevanten Attribute in eine Matrix überführt werden, dann einstehen gesamt acht Quadrate. In diesen acht Quadraten wurden nun die üblichen Gerätetypen, entsprechend ihrer Ausprägung eingeteilt. Die Gerätearten, bei denen alle

drei Dimensionen hoch ausgeprägt sind, erfüllen nach Aussage der Studie die Kriterien um als mobiles Endgerät bezeichnet werden zu können (Aichele & Schönberger, 2016).

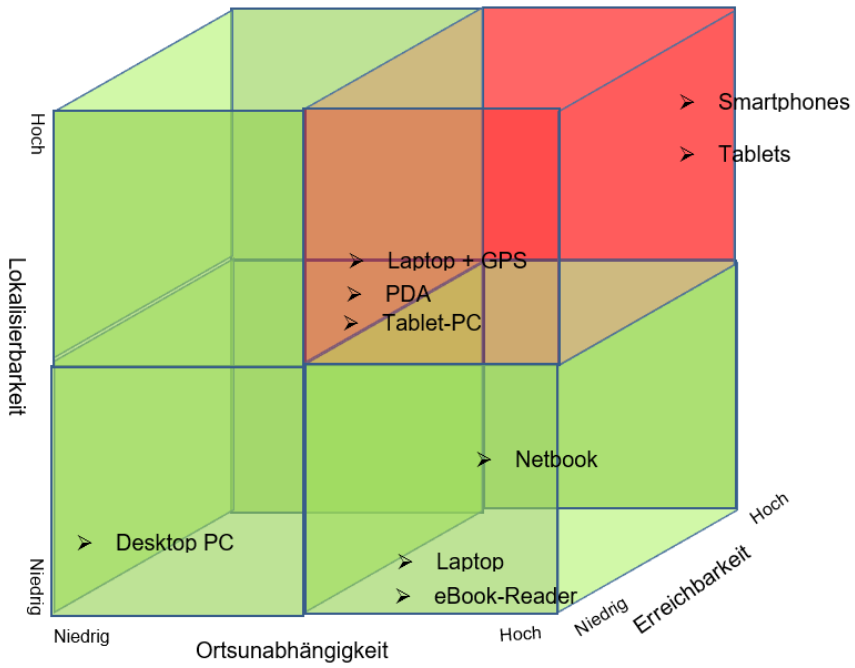



*Abbildung 2-1 Dimensionsmodell (Durlacher, 1999)*

Die zuvor gezeigte Grafik definiert zwar die Aussage was ein mobiles Endgerät ist, aber allein durch diese Attribute würde heutzutage ein Notebook beziehungsweise ein Netbook ebenfalls zu mobilen Endgeräten zählen, da diese Geräte ebenfalls die Eigenschaften Erreichbarkeit, Ortsunabhängig und Lokalisierbarkeit mit sich bringen. Aus diesem Grund wird die Unterscheidung zunehmend schwieriger, da Geräte wie Notebooks immer handlicher, also leichter und kompakter werden, und daher auch für den mobilen Einsatz sehr gut geeignet sind. In der Literatur werden Notebooks und Netbooks meist nur mehr in Hinsicht auf Mobilitätsphasen von mobilen Geräten unterschieden. Ein Beispiel von einer Mobilitätsphase wäre Radfahren. Laptops können zwar leicht und komfortabel transportiert werden aber können in solchen Phasen nicht genutzt werden. Smartphones hingegen können auch in Mobilitätsphasen sehr einfach eingesetzt werden (Maske, 2012).

Für die vorliegende Arbeit reicht die Unterscheidung zwischen mobilen und nicht mobilen Endgeräten durch die Größe und dem Gewicht nicht aus. Wenn nur diese Parameter betrachtet werden, könnten auch Offline-Notebooks, Netbooks oder mobile Geräte ohne GPS Funktion als mobile Endgeräte klassifiziert werden. Für diese Arbeit wird als entscheide Kriterium ob mobiles Endgerät oder kein mobiles Endgerät die zu Beginn erwähnte Mobilitätsphase gewählt. Laut Aiche und Schönberger werden unter Berücksichtigung dieses Kriteriums auch Tablets als mobile Endgeräte bezeichnet (Aichele & Schönberger, 2016). Dementsprechend werden in dieser Arbeit Tablets, Smartlets und Smartphones als mobile Endgeräte identifiziert.

### 2.1.2 Mobile Applikationen

Als mobile Applikation wird eine Anwendungssoftware für mobile Geräte bezeichnet. Der Begriff App bezieht sich zwar auf jegliche Art von Anwendungssoftware, so wird er im deutschen Sprachraum meist nur für Anwendungen herangezogen, die für Smartphones oder Tablets entwickelt wurden. Bei mobilen Applikationen kann zwischen nativen Apps, plattformunabhängigen Web- und Cross-Plattform-Apps unterschieden (Verclas & Linnhoff-Popien, 2011).

**Native Apps**

Bei Nativen Apps handelt es sich um Anwendungen die gezielt für eine Plattform entwickelt wurden. Somit können alle gerätespezifischen Features und Funktionen genutzt werden. So kann beispielsweise uneingeschränkt auf die Kamera, Datei-Explorer oder Sensoren zugegriffen werden. Ein wesentlicher Pluspunkt von nativen Applikationen ist die Offlinefähigkeit, die damit umgesetzt werden kann. Neben diesen Punkten sind native Apps auch performancetechnisch meist einen Schritt gegenüber Hybrid Apps voraus (Aichele & Schönberger, 2016).

Buettner und Simmons (2011) haben in ihrem Buch gezielt Web-Apps mit nativen Applikationen verglichen und haben neben der Offlinefähigkeit und der Performance noch einen weiteren entscheiden Vorteil von nativen Apps feststellen können. Der Vorteil von nativen Apps gegenüber Web-Apps ist jener, dass eine Verteilung über einen Marketplace stattfindet. So sind diese Apps an den jeweiligen Store gebunden und dieser nimmt den EnwicklerInnen das Bezahlsystem ebenfalls ab. Buettner und Simmons beschreiben in ihrem Buch weitere Vorteile eines gezielten Stores für mögliche Zielgruppen. So kann beispielsweise ein Endbenutzer, eine Endbenutzerin den Marketplace nach Kategorien oder ähnlichen Anwendungen durchsuchen und es können direkt Bewertungen von anderen BenutzerInnen herangezogen werden.

Native Applikationen bringen für die AnwenderInnen eine Vielzahl an Vorteilen mit sich. Trotzdem werden nicht nur native Apps am Markt angeboten und das liegt daran, dass der Aufwand für die EntwicklerInnen bei solchen Anwendungen höher ist, als zum Beispiel bei Cross-Plattform-Apps. So muss eine App die für Android, iOS und Windows Phone angeboten wird, für alle Plattformen separat entwickelt werden. Neben der mehrfachen Entwicklung müssen auch gewisse Abläufe wie zum Beispiel der Lebenszyklus einer App auf dem jeweiligen Betriebssystem beachtet werden. Zu guter Letzt haben die verschiedenen Plattformen spezifische Voraussetzungen, die eingehalten werden müssen (Heitkötter, Hanschke, & Majchrzak, 2012). In der folgenden Tabelle werden dem Lesenden die einzelnen Unterschiede nähergebracht.

| Voraussetzung | Android | iOS | UWP |
|---|---|---|---|
| Programmiersprache | Java | Objective-C/ Swift | .Net |
| Entwicklungsumgebung | Android Studio | XCode | Visual Studio |
| Bestimmtes Betriebssystem vorausgesetzt | Nein | Ja | Ja |

| | | | |
|---|---|---|---|
| Entwicklerkonto erforderlich | Nein | Ja | Ja |

*Tabelle 2-1: Gegenüberstellung Android, iOS und UWP*

Die Architektur von nativen Applikationen ähnelt der Architektur einer klassischen Windows-Desktop Anwendung wie zum Beispiel Word. Die Anwendung selbst benötigt keine weiteren Unterkomponenten. Es wird auch keine zusätzliche App benötigt. Sie wird lediglich auf dem jeweiligen Betriebssystem, welches auf dem mobilen Endgerät installiert ist ausgeführt.

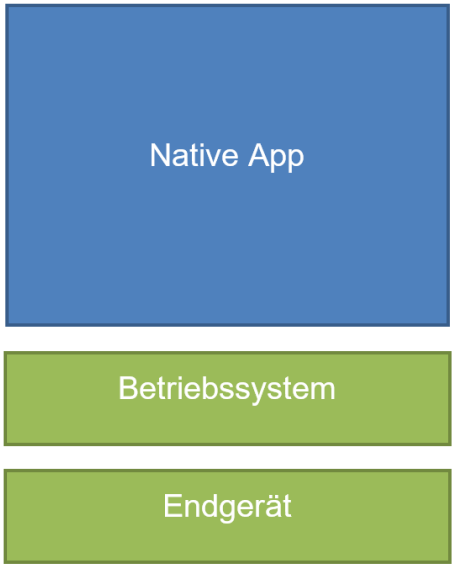



*Abbildung 2-2: Architektur einer nativen App (Schönberger, 2014)*

**Plattformunabhängige Web-Applikationen**

Unter Web-Applikationen werden Anwendungen verstanden, die nach einem sogenannten Client-Server-Model funktionieren. Anders als native Apps werden Webanwendungen also nicht auf einem mobilen Endgerät installiert und dort ausgeführt, sondern über den Webbrowser des Mobilgeräts. Hierbei kommen Technologien wie HTML5 oder JavaScript zum Einsatz, damit sie auf beliebigen Endgeräten betrieben werden können. Das Ziel einer Web-App ist jenes, dass sich diese genauso wie eine native App verhält. Die BenutzerInnen sollen im Idealfall die Anwendung nicht als Webseite wahrnehmen, sondern ähnlich wie eine App die auf dem Gerät installiert ist. Bei der Nutzung von mobilen Web-Applikationen müssen meist mehr Daten übertragen werden, als bei herkömmlichen Anwendungen. Dadurch können höhere Übertragungs-Gebühren für den entstehenden Datenverkehr ein Hindernis für viele AnwenderInnen darstellen. Durch ein sogenanntes Caching-Verfahren sind viele Web-Applikationen mittlerweile offlinefähig. In solch einen Fall kann eine mobile Web-Anwendung geladen und für einen späteren Zeitpunkt zwischengespeichert werden. Diese Funktion bietet einen praktischen Ausweg dar, um die App auch ohne aktiver Datenverbindung verwenden zu können (Franke, 2012).

Der große Vorteil an mobilen Web-Applikationen ist jener, dass sie eine sehr große Reichweite haben. Grundsätzlich kann jedes Gerät, welches einen Web-Browser und Zugang zum Internet hat eine Web-App aufrufen. Des Weiteren ist der Rollout neuer Versionen ebenfalls sehr einfach gehalten. Die aktualisierte App muss nicht im jeweiligen Store veröffentlicht und in weiterer Folge

von den einzelnen Benutzerinnen installiert werden, sondern diese wird einfach am Server aktualisiert und sobald die Seite erneut aufgerufen wird, wird bereits die aktualisierte Version angezeigt. Des Weiteren müssen sich die EntwicklerInnen nicht mit Richtlinien unterschiedlicher Marketplaces auseinandersetzen und Genehmigungen einholen. Nachteile von mobilen Web-Apps sind einerseits die eingeschränkten und zum Teil nicht vorhandenen Möglichkeiten gerätespezifische Features zu verwenden. Andererseits ist es auch eine Herausforderung, Richtlinien in Bezug auf Design- oder User-Experience für die jeweilige Zielplattform einzuhalten, damit bestimmte Erwartungen vom Endbenutzer, von der Endbenutzerin erfüllt werden (Buettner & Simmons, 2011).

Viswanathan (2017) bestätigt die Aussagen in seinem Artikel. Er nennt zudem auch den Nachteil der Performance von einer Web-Applikation und dass EntwicklerInnen sehr schnell an Limitierungen stoßen, da beispielsweise mobile Betriebssysteme so gebaut werden, dass der Web-Browser keinen Zugriff auf den Speicher hat. In seinem Artikel beschreibt er, dass Web-Applikationen dann von Vorteil sind, wenn es sich um leistungsarme Apps handelt und diese keinen Hardwarezugriff benötigen.

Abbildung 2-3 veranschaulicht die Architektur einer mobilen Web-App. Ähnlich wie bei den bereits erwähnten nativen Apps liegt auch hier ein installiertes Betriebssystem am Endgerät zu Grunde. Auf dem Betriebssystem setzt ein Browser auf, in dem in weiterer Folge die Web-App ausgeführt wird. Diese Applikation selbst ist mit HTML, CSS und JavaScript entwickelt (Peters, 2017).

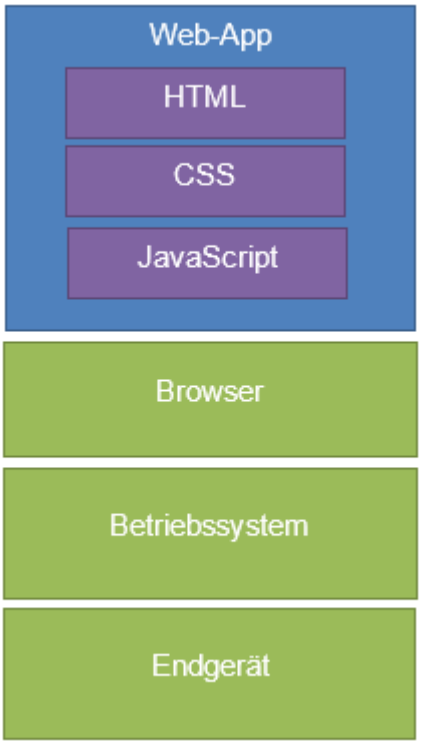



*Abbildung 2-3: Architektur einer Web-App (Peters, 2017)*

### Cross-Plattform-Apps

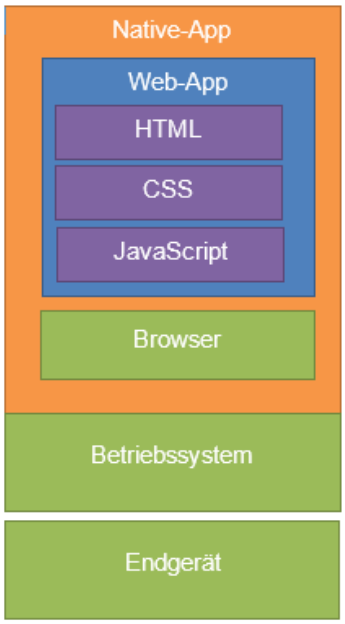
Der Begriff "Cross-Platform" ist ein Begriff in der Informatik für Programme, die die Eigenschaft haben auf verschieden Plattformen ausgeführt werden zu können. Im Deutschen wird diese

Funktionalität auch als plattformunabhängig, genauer als plattformübergreifend bezeichnet (Schnauder & Jarosch, 2001).

Petzold (2017) schreibt beispielsweise, dass sich das Verhalten der NutzerInnen in den letzten Jahren massiv verschoben hat. Desktopcomputer existieren zwar noch und es bleiben weiterhin Aufgaben die eine Tastatur und große Bildschirme benötigt wie zum Beispiel die Softwareentwicklung oder das Schreiben von Artikel, aber einen Großteil der Zeit verbringen AnwenderInnen auf mobilen Endgeräten, insbesondere um an Informationen zu kommen und für soziale Netzwerke. Der große Unterschied hierbei ist jener, dass die Userinterfaces dahingehend ein Wandel durchgemacht haben und hauptsächlich nur mehr mit Touch bedient werden können.

Bei Cross-Plattform-Apps gibt es zwei wesentliche Technologien. Einerseits können sogenannte Hybrid-Apps erstellt werden die eine Kombination aus mobilen Web-Apps und nativen Apps darstellen (Benisch & Müller, 2016). Andererseits können Anwendungen in einer Programmiersprache erstellt werden und das Framework selbst rendert den geschriebenen Source-Code, dass das jeweilige Betriebssystem in der Lage ist, die Anwendung auf dem Gerät zu installieren. Durch diese Technologie ist es mittlerweile möglich, eine geteilte Code-Basis für mobile Endgeräte und klassische Desktop-Computer zu verwenden (Petzhold, 2017).

Die Architektur einer Cross-Plattform kann somit, je nach eingesetzter Technologie, unterschiedlich aufgebaut sein. Bei hybriden Applikationen, wird in einen Container eine Web-App gelegt und dieser Container verhaltet sich in weiterer Folge wie eine native Applikation. Der Aufbau bei Applikationen, die eine geteilte Codebasis haben, ist dieselbe Architektur wie bei nativen Anwendungen.



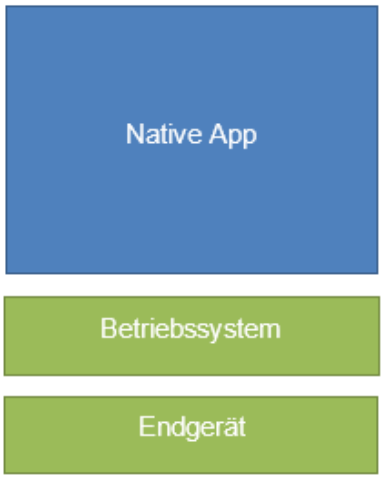




*Abbildung 2-4: Architektur von einer Xamarin-Applikation (Petzhold, 2017)*

*Abbildung 2-5: Architektur von einer Web-App (Schönberger, 2014)*

Zusammenfassend kann gesagt werden, dass mittlerweile sehr viele verschiedene Technologien auf den Markt existieren und dass alle verschiedene Vor- und Nachteile bieten und sich für bestimmte Einsatzzwecke eignen. Wie aus diesem Abschnitt auch hervor geht, kann keine

pauschale Aussage getroffen werden, welche Technologie grundsätzlich den anderen vorzuziehen ist. Tabelle 2-2 fasst die Eigenschaften der einzelnen Technologien im Groben zusammen.

| Eigenschaft | Native App | Web-App | Cross-Plattform App |
|---|---|---|---|
| Plattformunabhängigkeit | Nicht vorhanden | Vorhanden | Vorhanden |
| Hardwarezugriff | Vorhanden | Nicht vorhanden | Vorhanden |
| Hohe Performance | Vorhanden | Nicht vorhanden | Teilweise vorhanden |
| Einfache Versionierung | Nicht vorhanden | Vorhanden | Teilweise vorhanden |
| Einfacher Zugang für Entwickler | Nicht vorhanden | Vorhanden | Teilweise vorhanden |

*Tabelle 2-2: Vergleich der Technologien*

### 2.1.3 Mobile-Device-Management

Unter Mobile-Device-Management (MDM) – im Deutschen oft als Mobilgeräteverwaltung bekannt – wird die zentralisierte Verwaltung von Mobilgeräten wie Smartphones, Sub-Notebooks, PDAs oder Tablets verstanden. Diese Administration kann entweder mit Hilfe von Software oder Hardware stattfinden. Hierbei wird vor allem die Inventarisierung von mobilen Geräten in Organisationen, die Software-, Daten- und Richtlinienverwaltung ins Visier genommen. Ein weiteres Ziel der Mobilgeräteverwaltung ist der Schutz der Daten, die sich auf den mobilen Geräten befinden (Hommes, 2016).

Kersten (2014) beschreibt die Problematik von "Bring Your Own Device" und die Hürden die dadurch für die IT-Abteilung von Unternehmen entstehen. Wenn MitarbeiterInnen ihre eigenen Mobilgeräte verwenden dürfen um Unternehmensdaten abzurufen, ist es oft schwierig eine gewisse Transparenz und Kontrolle zu schaffen, damit eine erfolgreiche Datensicherheit gewährleistet ist. Die britische Studie Vanson Bourne zeigt beispielsweise, dass 59 Prozent der befragten IT-Leiter glauben, dass ein Verbot von BYOD einen Wettbewerbsnachteil zur Folge hätte. Aus diesem Grund versuchen immer mehr Unternehmen diesen Trend zu akzeptieren und die dadurch entstandenen Sicherheitslücken mittels Mobile-Device-Management zu beseitigen. (Daubney, 2014).

## 2.2 Betriebssysteme

Im folgenden Abschnitt wird dem Lesenden ein Überblick über die zur Verfügung stehenden Betriebssysteme und Plattformen für mobile Anwendungen gegeben. Zu Beginn wird kurz die Geschichte der einzelnen Betriebssysteme vorgestellt und in weitere Folge die Architektur näher beschrieben. Abschließend erfolgt eine Gegenüberstellung der vorgestellten Plattformen.

In dieser Arbeit werden nicht nur die zwei am weit verbreiteten Betriebssysteme vorgestellt. Es wird auch eine weitere Plattform vorgestellt.

- Android
- iOS
- Universal Windows Plattform

Eine Statistik aus dem Jahr 2017, über mobile Betriebssysteme zeigt zwar, dass in Österreich Android mit 63.8 Prozent und iOS mit 34.7 Prozent den österreichischen Markt klar dominiert und somit weitere Systeme nicht betrachtet werden müssten, trotzdem wird im folgenden Abschnitt eine weitere Plattform vorgestellt. Dabei handelt es sich um die Windows-Phone-Runtime, die nur mehr einen Marktanteil von 1.3 Prozent haben (ÖWA, 2017).

Auch im internationalen Vergleich ändert sich an der Verteilung von den Betriebssystemen nicht viel. Eine Auswertung vom Statistik-Portal zeigt einen weltweiten Vergleich von verschiedensten Betriebssystemen. Hier ist sehr gut zu erkennen, dass der Markt Großteils mit Android und iOS besetzt ist (statista, 2017).

### 2.2.1 Android

Android wurde im Jahre 2003 von Andy Rubin gegründet und von Google im Sommer 2005 gekauft. Zu Beginn konzentrierten sich die EntwicklerInnen von Android ausschließlich auf die Steuerung von Digitalkameras (Elgin, 2005). Die ersten Versionen wurden im Jahr 2008 veröffentlicht. Der große Vorteil von Android ist jener, dass es sich vor allem durch die Flexibilität und Offenheit auszeichnet, sodass viele verschiedene Gerätehersteller die Möglichkeit haben, mobile Geräte zu produzieren und darauf Android als Betriebssysteme auszuliefern. Die Offenheit führt dazu, dass viele Gerätehersteller eigene Anpassung an der Oberfläche durchführen, sodass nur die wenigsten Endgeräte mit einem unveränderten Android-Betriebssystem ausgeliefert werden (Künneth, 2016).

Android selbst basiert auf dem Betriebssystem Linux. Dieses beinhaltet alle notwendigen Treiber um die darunterliegende Hardware anzusprechen. Somit bildet Linux die Basis für die Android-Runtime, die aus speziellen Core Java Libraries für Android und einer virtuellen Maschine namens Davlik besteht. Apps, die auf einem Android-Gerät installiert werden, laufen in dieser virtuellen Maschine (Krajci & Cummings, 2016).

Im Laufe der Zeit versucht Google die Architektur von Android zu verbessern und hat mit der Android Version 5.0 – auch Lollipop genannt – eine neue virtuelle Maschine namens Android Runtime vorgestellt. Ab dieser Version laufen die Android Apps in dieser neu entwickelten virtuellen Maschine. Erstmals wurde die Version als sogenannte Vorschauversion unter Android 4.4 ausgeliefert. Ab der Version 5.0 wurde die virtuelle Maschine Davlik vollständig durch die Android Runtime ersetzt (Morgillo & Viola, 2016).

In Android kann die Verteilung von Anwendungen auf mehrere Arten durchgeführt werden. Der bekannteste und am häufigste verwendete Weg ist jener des Google Play Stores. Dieser ist wiederrum eine App die bereits in Android integriert ist. Neben dem Google Play Store, sind auch

weitere Applikationen wie beispielsweise Google-Maps direkt in Android integriert. Über diesen Store kann ein Großteil der potenziellen Kunden ohne großen Aufwand erreicht werden. Gerätehersteller nutzen die Flexibilität von Android meist aus und installieren bestimme Applikationen, die nur für die hergestellte Hardware zur Verfügung steht (Joseph, 2015).

Neben dieser Art der Verteilung, ist es unter Android auch möglich, die Apps außerhalb des Google Play Stores zu installieren. Dazu muss jedoch eine bestimmte Systemeinstellung aktiviert werden. Diese Einstellung ermöglicht den BenutzerInnen das Installieren von unbekannten Quellen. Sobald diese Option aktiviert wird, kann das gewünschte Softwarte-Development-Kit installiert werden (Haseman & Grant, 2016).

### 2.2.2 iOS

Neben Android gibt es eine weitere weit verbreitete Plattform und zwar iOS. Dieses Betriebssystem ist ausschließlich für die mobilen Geräte iPhone, iPad und iPod von Apple. Erstmalig wurde diese Plattform im Jahr 2007 zusammen mit dem ersten iPhone präsentiert. Der Erfolg von iOS lag zu Beginn vor allem an der komfortablen Touch-Bedienung und die leichte Bedienbarkeit gegenüber der Konkurrenz (Negm-Awad, 2012).

Das Grundgerüst von iOS passiert auf einem Kernbetriebssystem, dass von Mac OS X abgeleitet wurde. Die weiteren Eben von diesem Betriebssystem bestehen aus sogenannten Core Services und einer Media-Ebene, die den Zugriff auf den persistenten Speicher, Netzwerk und Grafik sowie Audiofunktionen abstrahieren. Die Oberste Ebene der Architektur ist die sogenannte Cocoa Touch-Ebene, auf der die zu entwickelnden Anwendungen basieren. Cocoa-Touch wurde von Cocoa aus MAC OS x abgeleitet und bietet SoftwareentwicklerInnen eine einfach Möglichkeit, touch-optimierte Applikationen zu entwickeln (Mayer & Wichers, 2016).

Ähnlich wie bei Android gibt es auch unter iOS eine Applikation, den App Store, zur Auslieferung der entwickelten Apps. Eine Installation über Drittanbietern kann unter iOS nicht so einfach durchgeführt werden, wie unter Android. Die Verteilung findet ausschließlich über den App Store statt und jede Veröffentlichung im Store muss gewisse Richtlinien und Vorgaben erfüllen. Erst wenn diese Prüfung erfolgreich abgeschlossen wurde, wird das App im App Store veröffentlicht (Wurm, 2017).

Unternehmen haben die Möglichkeit ihre Apps als sogenanntes Enterprise App auszuliefern. Durch diese spezielle Form ist es möglich, dass spezifische Anwendungen für Unternehmen nicht im öffentlichen Store zugänglich sind. Diese Applikationen unterliegen auch nicht den strengen Vorgaben von Apple und werden auch nicht geprüft. Unternehmen können somit für ihre MitarbeiterInnen einen eigenen Unternehmensstore einrichten und dort gewisse Anwendungen zur Verfügung stellen (Turner, 2011).

### 2.2.3 Universal Windows Plattform

Die dritte und letzte Plattform, die im Zuge dieser Arbeit vorgestellt wird, ist die universal Windows Plattform. Diese Laufzeitumgebung wurde mit der in Windows 8 eingeführte Windows Runtime eingeführt. Das Ziel von Microsoft ist jenes, dass ein zentrales Betriebssystem für verschiedenste Geräte eingesetzt wird. So können Anwendungen die mit der Universal Windows Plattform – kurz UWP – entwickelt werden, auf allen Geräten installiert werden, die die Windows Runtime unterstützen. Darunter fallen Computer auf denen Windows 8, Windows 8.1 oder Windows 10 installiert ist, aber auch mobile Geräte, wie beispielsweise Smartphones, auf denen Windows 10 mobile installiert ist. Neben Computer und mobilen Geräten, können die Anwendungen auch auf Spielkonsolen wie zum Beispiel auf der Xbox One installiert werden (Baidachnyi, 2016).

UWP basiert auf der Laufzeitumgebung Windows Runtime. Diese ermöglicht es Applikationen zu entwickeln, die auf einem PC, Tablet und Smartphone funktionieren. Das zugrundeliegende Model ist eine verbesserte Version des Component Object Models und ermöglicht es EntwicklerInnen, auf der Kombination aus HTML, CSS und JavaScript oder XAML und einer der Programmiersprachen Visual Basic Dot-Net, C++ oder C#, Programme zu erstellen (Richter & Bospoort, 2013).

Die Verteilung der Applikationen funktioniert ähnlich wie unter Android und iOS. Auf jedem Endgerät, welches die Windows Runtime – sprich Windows 10 – installiert hat, ist eine Applikation installiert, die sich Store nennt. Diese Anwendung dient dazu, dass die einzelnen Apps installiert werden können. Ähnlich wie bei iOS werden auch Applikationen, die im Microsoft Store veröffentlicht werden, geprüft und müssen gewissen Richtlinien entsprechen (Cordts, 2016).

Auch bei Microsoft ist es möglich, dass ein eigener Unternehmensbereich eingerichtet wird, in dem gewisse Applikationen zur Verfügung gestellt werden. So kann beispielsweise das Unternehmen entscheiden, welche Anwendungen im Store zur Verfügung gestellt werden und welche ausgeblendet werden. Des Weiteren ist es möglich, Apps auch ohne Store zu installieren. Hierfür muss in den Einstellungen die Option "Apps querladen" aktiviert werden. Zu beachten ist hierbei, dass Microsoft Installationen nur dann akzeptiert, wenn die App signiert ist (Borycki, 2016).

### 2.2.4 Vergleich der Betriebssysteme

Wie bereits eingangs erwähnt sind die Betriebssystem Android und iOS am weitersten verbreitet. Zirka 98 Prozent des mobilen Internetverkehrs mit Smartphones in Österreich erfolgt mit einem Gerät, dessen Betriebssystem entweder Android oder iOS ist (ÖWA, 2017). Im folgenden Abschnitt wird erläutert, wie sich dieser Prozentsatz zwischen diesen beiden führenden mobilen Plattformen aufteilt und in wie weit Microsoft mit ihrer mobilen Plattform noch eine Rolle spielt.

Eine Studie von Statista zeigt den weltweiten Trend seit 2009 von mobilen Endgeräten. Die Statistik zeigt, dass beispielsweise Android im Jahr 2009 nur einen sehr geringen Marktanteil hatte und iOS sich den Markt mit Nokia geteilt hat. Im Jahre 2011 hatte Microsoft ihr Produkt auf

den Markt gebracht und konnten sich zu Beginn einige Prozent vom Markt sichern. Vor allem Unternehmen fanden die Idee hinter dem Betriebssystem und der Vernetzung mit den Office-Produkten von Microsoft gut. Seit 2009 konnte iOS immer einen Schnitt von 30 bis 40 Prozent vom Markt sichern und schafft es auch diesen sehr gut zu halten. Android selbst hat sich vor allem in den letzten sieben Jahren sehr viele Marktanteile sichern können. Dies liegt vor allem an der Flexibilität des Betriebssystems und daran, dass alle Gerätehersteller, außer Apple, auf diese mobile Plattform zurückgreifen.

Wenn der Tablet-Markt näher betrachtet wird, ist sehr gut zu erkennen, dass Windows in den letzten Jahren immer mehr Anteile bekommt. So zeigt eine Statistik von Strategy Analytics, dass zwar das Wachstum an Tablet sinkt, aber nur iOS und Android weniger Zuwachs haben. Windows selbst schafft es, trotz sinkender NutzerInnenzahlen mehr Marktanteile zu sichern.

| Betriebssystem | Marktanteil in % 2015 | Marktanteil in % 2016 |
|---|---|---|
| Android | 65 | 63 |
| iOS | 23.2 | 20.6 |
| Microsoft | 11.8 | 16 |

*Tabelle 2-3: Vergleich der Betriebssysteme (Gschweidl, 2017)*

Sehr gut zu erkennen ist jedoch, dass Android sowohl im Smartphone Bereich als auch im Tablet Bereich nach wie vor Marktführer sind. Vor allem wenn die letzten sieben Jahre betrachtet werden, ist Android die einzige Plattform, die ein stätiges Wachstum verzeichnen kann. iOS verliert nicht viel, aber die Kurve zeigt vor allem in Europa nach unten (Gschweidl, 2017). Microsoft und die von ihnen entwickelte Windows 10 Mobile Plattform, wird auch seitens Microsoft nicht mehr weiterentwickelt und es wird vermutet, dass dieses Betriebssystem in naher Zukunft nicht mehr existiert (heise.de, 2017).

# 3 EINFÜHRUNG IN PLATTFORMUNABHÄNGIGE SW-ENTWICKLUNG

Das folgende Kapitel beschäftigt sich genauer mit der Thematik der plattformunabhängigen Softwareentwicklung. Zu Beginn erfolgt die Definition von gängigen Begriffen in diesem Umfeld. Da viele Begriffe oftmals falsch Verstanden werden, werden dem Lesenden diese durch eine genauere Definition nähergebracht. Im Anschluss daran werden einige Technologien, die derzeit auf dem Markt existieren kurz vorgestellt und die Funktionsweise erklärt. Nach der Vorstellung der gängigen Technologien werden bekannte Umsetzungen vorgestellt und Erfahrungen von anderen EntwicklerInnen präsentiert. Abschließend wird die Thematik, die in diesem Kapitel genauer erläutert wurde kurz zusammengefasst.

## 3.1 Begriffsdefinitionen

Im folgenden Abschnitt werden die gängigsten Begrifflichkeiten, die in der Literatur im Zusammenhang mit plattformunabhängiger Softwareentwicklung fallen, beschrieben. Dem Lesendem wird eine Übersicht über die gängigsten Fachtermini geben. Zudem wird erläutert welche Bedeutung die Begriffe mit sich bringen. Die Definition der folgenden Begriffe erfolgt auf facheinschlägiger Literatur:

- Cross-Plattform-Applikation
- Hybrid Applikation
- Multi-Channel-Apps
- Single-Page-Applikation
- Fat Binaries und Univerals Binaries

### 3.1.1 Cross-Plattform-Applikationen

Da Cross-Plattformen Applikationen bereits im Abschnitt 2.1.2 näher beschrieben worden sind, werden im folgenden Abschnitt nur die wichtigsten Funktionalitäten aufgeführt.

Apps werden dann als Cross-Plattform-Applikation bezeichnet, wenn das zu Grunde liegende Betriebssystem keinen Einfluss auf die Verwendung der App hat. Moderne Frameworks erlauben es, dass der Softwareentwickler, die Softwareentwicklerin beim Entwickeln keine großen Unterschiede zur klassischen App-Entwicklung hat. Es wird eine gemeinsame Code-basis generiert und diese wird in weiterer Folge für das jeweilige System gerendert (Rauber, 2017).

Krämer (2018) schreibt darüber, dass heutzutage die plattformunabhängige App-Entwicklung sehr viele Einsatzmöglichkeiten bietet, da immer mehr hardwarespezifische Funktionalitäten verwendet werden können. So können App-EntwicklerInnen ohne spezielle Zusatzplugins auf

Sensoren oder auch den Speicher zugreifen. Dadurch nimmt der Trend von Cross-Plattform-Applikationen immer mehr zu.

### 3.1.2 Hybrid-Applikationen

Hybrid-Applikationen sind Anwendungen die eine Kombination aus klassischen Web-Anwendungen und einer nativen App. Die Funktionalitäten einer App werden mit Webtechnologie wie HTML, CSS und JavaScript umgesetzt und in der jeweiligen Browser-Umgebung ausgeführt. Die Browser-Umgebung fungiert dabei als Container und wird somit lokal auf dem Gerät installiert. Diese Installation verläuft meist mittels Verknüpfung.

Bei hybriden Anwendungen wird vom Framework eine Library geladen, die die Kommunikation zwischen JavaScript und der jeweiligen betriebssystemspezifischen Sprache herstellt. Dies bringt den großen Vorteil mit sich, dass Hybrid-Apps auf diverse Hard- und Software-Komponenten des mobilen Endgerätes zugreifen können. Dies ermöglicht einen Zugriff auf Kamera, Kontakte, GPS oder aber auch auf den lokalen Speicher. Moderne Render-Engines verarbeiten HTML5-, CSS3 und JavaScript-Codes um einiges schneller und dadurch wird die Performance von Hybrid Applikationen um einiges gesteigert (Eschenbach, 2016).

### 3.1.3 Single-Page-Applikationen

Ein weit verbreiteter Fachterminus im Bereich Web-Applikationen aber auch plattformunabhängiger Softwareentwicklungen sind sogenannte Single-Page-Applikationen. Diese Technologie ermöglicht es, dass aus einem einzigen HTML-Dokument die Inhalte dynamisch nachgeladen werden können. Durch diesen Aufbau unterscheiden sich solche Web-Architekturen von klassischen Webanwendungen, die aus mehreren, untereinander verlinkten HTML-Dokumenten bestehen. Dieser Aufbau setzt jedoch voraus, dass eine Webanwendung in Form einer Rich-Client- bzw. Fat-Client-Verteilung zu entwickeln ist. Durch die verstärkte clientseitige Ausführung der Webanwendung wird die Serverlast reduziert und die Möglichkeit geschaffen, selbstständige Webclients zu entwickeln, die zum Beispiel eine Offline-Unterstützung anbieten (Steyer & Softic, 2015).

Woiwode, Malcher, Koppenhagen und Hoppe (2017) beschreiben das der Unterschied zwischen klassischen, serverseitigen Webanwendungen und Single-Page-Applikationen hauptsächlich in der geänderten Rolle des Servers liegt. Die ursprünglichen Aufgaben eines Webservers waren das Ausliefern der Ressourcen wie HTML, CSS, JS und das komplette Rendering. Bei modernen Single-Page-Applicationen liefert der Server nur mehr die Daten, wie beispielsweise ein JSON-Objekt. Navigiert ein User, eine Userin auf eine moderne Webseite, fordert die Anwendung Daten vom Server an und rendert diese anschließend dynamisch auf dem Client. So fungieren heutzutage Server großteiles als Datenlieferant. Eine Aufgabe, die weiterhin vom Server übernommen wird, ist die Rolle während der Authentifizierung.

## 3.2 Technologien

In diesem Abschnitt werden die möglichen Technologien erläutert, mit denen eine mobile plattformunabhängige App auf den in Abschnitt 2-2 beschriebenen Plattformen umgesetzt werden kann. Neben diesen drei erwähnten Betriebssystemen stehen in der Praxis weitere Systeme zur Verfügung. Da die Anforderungen an die, in weiterer Folge entwickelten App, an diese drei Plattformen gestellt wurde, werden nur Technologien aufgezählt, welche mindestens diese drei Betriebssysteme unterstützen. Dazu wird kurz auf die Geschichte der einzelnen Technologien eingegangen gefolgt von einer Architekturbeschreibung. Abschließend werden die vorgestellten Frameworks gegenübergestellt und über bekannte Umsetzungen berichtet.

Zu den bekanntesten und am häufigsten eingesetzten Frameworks zählen:

- Xamarin
- Unity
- Phonegap
- Framework7
- Mobile Angular UI

### 3.2.1 Xamarin

Miguel de Icaza, ein Mitarbeiter von Ximian begann 2001 eine Linux-Version die auf Dot-Net basiert zu entwickeln. Aus diesem Projekt wurde das Unternehmen Mono, welches im Jahr 2011 Xamarin gründete. Im Jahr 2016 wurde Xamarin von Microsoft aufgekauft und als Open-Source-Produkt neu vorgestellt. Dieses Framework bietet App-EntwicklerInnen in der Sprache C-Sharp Anwendungen für iOS, Android und UWP, auf gemeinsamer Code-Basis, entwickeln (Krämer, 2018).

Das User-Interface kann in Xamarin auf zwei verschiedene Arten entwickelt werden. Zum einen können native Apps erstellt werden, wo die Oberfläche für jede Plattform separat entwickelt werden muss. Zum anderen können die sogenannten Xamarin-Forms zur Beschreibung der Oberfläche verwendet werden. Der große Vorteil der XAML-beschreibenden Sprache Xamarin-Forms ist jener, dass für die drei Plattformen Android, iOS und Universal Windows Application, einmal die Benutzeroberfläche designt werden muss und das Framework diese in die jeweilige Sprache übersetzt. Ziel von Xamarin ist jenes, dass jedes Control auf dem jeweiligen mobilen Endgerät das gleiche Verhalten und Aussehen hat, wie wenn es sich um eine native Applikation handelt. Die BenutzerInnen sollen keine Unterschiede zu einer nativen Anwendung feststellen können.

Wie in Abbildung 3-1 zu erkennen ist, werden die Controls auf jedem Endgerät unterschiedlich dargestellt (Petzhold, 2017).

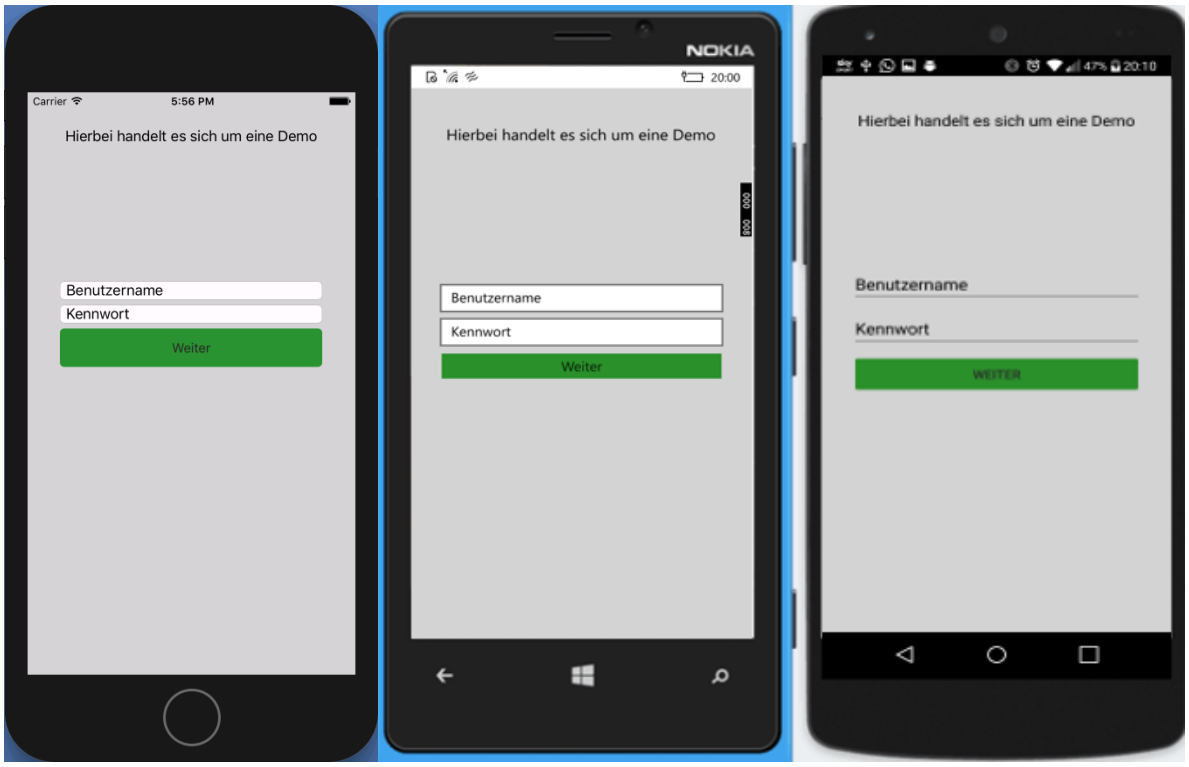



*Abbildung 3-1: Xamarin iOS, UWP und Android*

Auf der Build 2017 verkündete Microsoft, dass das Ziel von Xamarin jenes ist, dass eine Applikation entwickelt wird und diese in weiter Folge auf jedem mobilen Endgerät auf dem Android, iOS oder Windows 10 Mobile installiert ist, installiert werden kann und zusätzlich auf jedem normalen Desktop-Rechner. Das heißt, dass sich Microsoft das Ziel gesetzt hat, alle Betriebssystem zu unterstützen. Somit sollen Linux-, Apple- und Microsoft-AnwenderInnen dieselbe Software zur Verfügung gestellt bekommen, ohne dass diese für jede Plattform separat entwickelt werden muss (Ortinau, 2017). So veröffentlichte Xamarin am 22. Februar 2017 einen Artikel, in dem beschrieben wird, wie EntwicklerInnen eine App für Mac OS entwickeln können (Hartley, 2017).

### 3.2.2 Unity

Unity wurde 2004 in Kopenhagen gegründet und ist eine Laufzeit- und Entwicklungsumgebung für Spiele. Zielplattformen sind Computer, Spielekonsolen, mobile Geräte und Webbrowser. Da bei Unity interaktive 3D-Grafik-Anwendungen entwickelt werden, können diese Applikationen auf Windows, Linux und macOS ausgeführt werden (Theis, 2017).

Die Mechanismen, die Unity bietet, können mit selbst geschriebenen Programmen, sogenannten Skripten, erweitert werden. Diese sind notwendig, um den Spielablauf beziehungsweise die Logik

zu beschreiben. Das Scripting selbst passiert auf Mono und bietet die Scriptsprache UnityScript, C-Sharp und Boo. Damit eine plattformunabhängige Entwicklung möglich ist, kann Unity in der Pro-Version um C++-Programme erweitert werden (Seifert, 2014).

### 3.2.3 Phonegap

Bei Phonegap handelt es sich um ein Framework welches zur Entwicklung von hybriden Applikationen eingesetzt wird. Im Jahr 2011 wurde die Firma Nitobi, welche das Framework entwickelt hat, von Adobe aufgekauft. Phonegap ermöglicht es, Apps für mobile Endgeräte mit JavaScript, HTML5 und CSS 3 zu schreiben, anstelle von gerätespezifischen Programmiersprachen wie Swift oder Java (Hann, 2013).

Ross (2013) beschreibt, dass Phonegap durch die eingesetzte Technologie den Vorteil hat, mittels PlugIns auf native, gerätespezifischen Funktionalitäten aufrufen. So können Applikation auf Kontakte, GPS oder auch auf die Kamera zugreifen. Da es sich um eine mobile Web-Applikation handelt, die in einem Container läuft, ist die Performance gegenüber einer nativen Anwendung schlechter. Diese Problematik führt oft dazu, dass die Anwendungen von BenutzerInnen nicht akzeptiert wird und auf eine native App zurückgegriffen wird.

### 3.2.4 Framework7

Framework7 ist ein kostenloses, mobiles HTML-Framework für die Entwicklung von mobilen Hybrid-Anwendungen oder Web-Apps mit iOS- und Android-Look. Der Hauptansatz des Frameworks besteht darin, dass EntwicklerInnen Apps entwickeln können, die auf allen gängigen mobilen Geräten funktionieren, die iOS oder Android als Betriebssystem haben. Dies ermöglicht es eine große Anzahl an BenutzerInnen anzusprechen, dass die Anwendungen für jede Plattform separat entwickelt werden müssen. Das Grundprinzip hinter dem Framework jenes das eine Applikation entwickelt werden kann, die nicht von gewissen Plattformen abhängig ist. Derzeit konzentrieren sich die EntwicklerInnen hinter Framework7 aber nur auf die zwei größten Vertreter, Android und iOS. Das Look der Applikation passt sich ähnlich wie bei Xamarin an die jeweilige Plattform an. So haben die Apps unter Android ein anderes Aussehen als unter iOS.

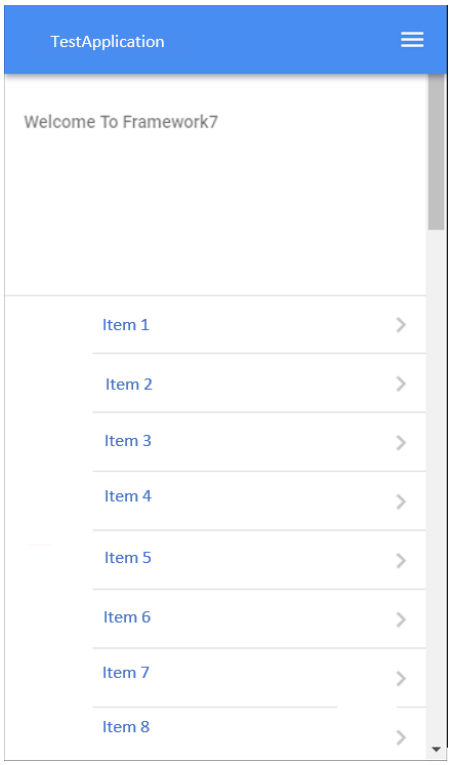



*Abbildung 3-3: Android*

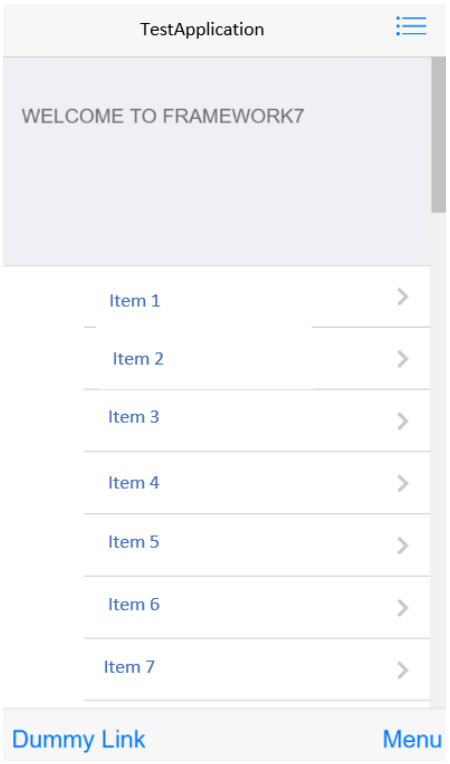



*Abbildung 3-2: iOS*

### 3.2.5 Mobile Angluar UI

Angular, früher Angular JS, ist ein Typescript-basiertes Front-End-Framework. Es ist eine Open-Source-Software, bei der die Firma Google mitwirkt. Mithilfe dieser Technologie werden Single-Page-Applikationen entwickelt, die durch ihre Funktionsweise eine hohe Performance als klassische Web-Anwendungen haben und zusätzlich eine Offlinefunktionalität mit sich bringen. Darüber hinaus sind Applikationen, die mit Angluar entwickelt werden responsive. Das heißt, dass sich die Anwendung an die Größe des Bildschirmes anpasst und somit eine bessere Usability habenals klassische Webseiten (Steyer & Softic, 2015).

Bei der mobile Angular UI handelt es sich um eine Erweiterung, die speziell für mobile Endgeräte geeignet ist. Die Funktionalität ist ähnlich wie bei Sencha Touch oder jQerry Mobile. Anders als bei Technologien wie Xamarin oder Framework7, passt sich das Aussehen einer Anwendung nicht an die jeweilige Plattform an. Die folgenden Abbildungen zeigen kurz, dass zwischen iOS Android und Windows-Phone kein Unterschied zu erkenn ist (Asim, 2017).

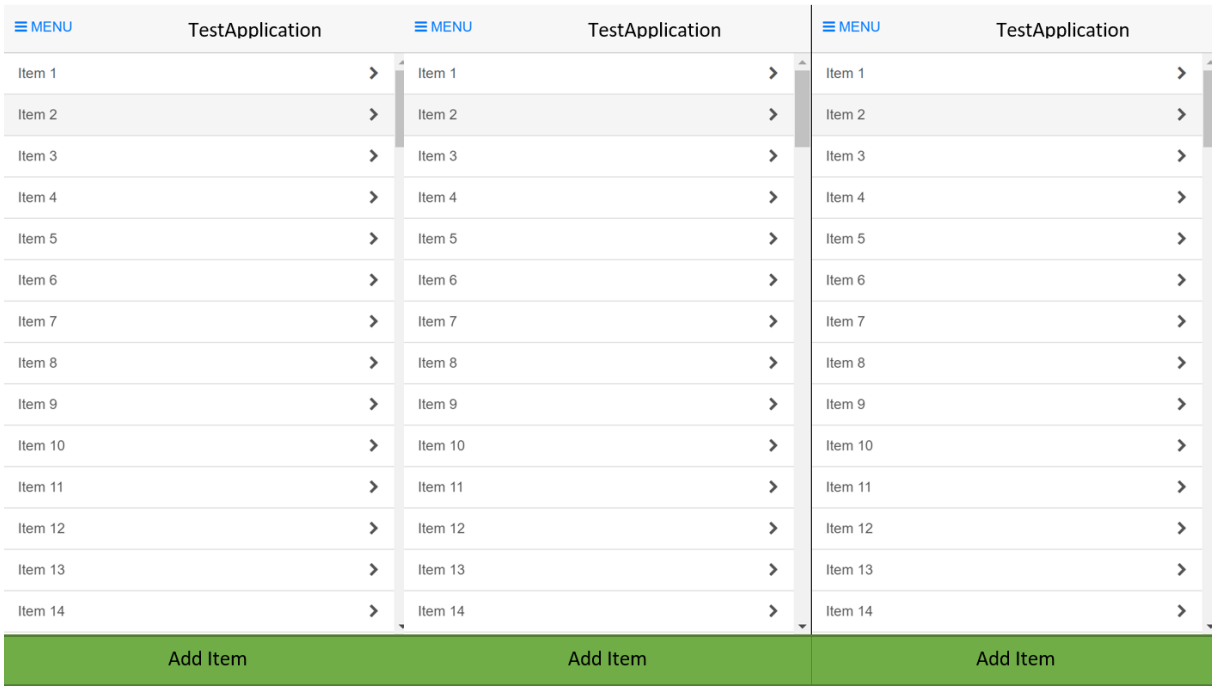



*Abbildung 3-4:Android, iOS und Windows Phone*

### 3.2.6 Zusammenfassung

Zusammenfassend kann gesagt werden, dass sich bereits einige Technologien auf dem Markt etabliert haben, aber keine generelle Entscheidung getroffen werden kann, welche der oben erwähnten Technologien die Beste ist. Die Entscheidung, welche dieser Frameworks verwendet werden soll, hängt sehr stark von den Anforderungen ab. So haben Apps, bei denen zuvor definiert wird, dass diese aus dem Store heruntergeladen werden sollen, andere Grundanforderungen als Applikationen bei denen die Verteilung keine Rolle spielt. Ein weiterer Entscheidungsfaktor welche Technologie beziehungsweise ob eine Web-Anwendung ausreicht ist jener welche gerätespezifischen Eigenschaften benötigt werden. Da der Prototyp, der im Zuge dieser Arbeit entwickelt wird, als native Applikation zur Verfügung gestellt werden soll und viele gerätespezifischen Funktionen benötigt, hat sich der Autor dieser Arbeit dazu entschieden, die Anwendung mittels Xamarin zu entwickeln. Der Grund für diese Technologie ist einerseits jener, dass die Apps als native Applikation zur Verfügung gestellt werden kann und andererseits kein weiteres Knowhow aneignet werden muss, da der Autor dieser Arbeit bereits viel Erfahrung im Dot-Net-Bereich hat und bereits einige Projekte mit Xamarin umgesetzt hat.

# 4 EMPIRISCHER TEIL

Im folgenden Kapitel wird der empirische Teil dieser Arbeit erläutert. Der Aufbau des empirischen Teiles setzt sich aus einer Befragung von fünf Expertinnen und Experten und eines Erfahrungsberichtes des Autors zusammen. Neben dieser Untersuchung wird die Umsetzung eines Prototyps einer mobilen Applikation näher beschrieben. Der Aufbau und der Ablauf dieses Kapitels sehen wie folgt aus:

- Entwicklung des Prototyps
- Experteninterview
- Erfahrungsbericht
- Bewertung der Interviews und des Erfahrungsberichtes

## 4.1 Entwicklung des Prototyps

In diesem Abschnitt wird die Umsetzung des Prototyps näher beschrieben. Zu Beginn werden die ausgewählten Plattformen, die gewählte Technologie und das zugrungeliegende Framework erläutert. Nach dieser Einführung erfolgt die Dokumentation der Umsetzung der Anforderungen.

Der Prototyp der in dieser Arbeit vorgestellt wird, wird als experimenteller Prototyp entwickelt. Diese Art des Prototyping ermöglicht es, Erfahrungen sammeln zu können und somit wird herausgefunden, ob die Realisierung des entwickelten Prototyps möglich wäre. Nach Erstellung des ersten Entwurfes wird in weiterer Folge eine umfangreiche Problemanalyse und Systemspezifikation durchgeführt. Die gewonnenen Erkenntnisse können anschließend für die Entwicklung des Produktes verwendet werden. Da das Ziel dieser Arbeit nicht die Entwicklung einer fertigen Software ist, werden die Problemanalyse und Systemspezifikation nur in Bezug auf die Machbarkeit einer Cross-Plattform-Applikation näher betrachtet (Budde, Kautz, Kuhlenkamp, & Züllighoven, 1992).

### 4.1.1 Anforderungen

Im Kapitel 4.1.1. werden dem Lesendem kurz die Anforderungen an den Prototypen vorgestellt und näher beschrieben. Die Anforderungen, die in der folgenden Tabelle vorgestellt werden, wurden zusammen mit dem Auftraggeber erarbeitetet.

Tabelle 4-1 fasst die Anforderungen zusammen, die aus dem Gespräch mit den Stakeholdern entstanden sind.

| # | Anforderung |
|---|---|
| A01 | Einsatz empfangen |
| A02 | Detaillierte Einsatzübersicht |

| | |
|---|---|
| A03 | Aktuellen Einsatz beenden |
| A04 | Navigation zu aktivem Einsatz zeigen |
| A05 | Liste der Einsätze anzeigen |
| A06 | Einsatz auf Karte anzeigen |
| A07 | Horizontal und vertikal bedienbar |

*Tabelle 4-1: definierte Anforderungen*

Die Anforderungen, die vom Auftraggeber gestellt wurden, wurden hier in sieben Use Cases unterteilt und müssen mittels dem Prototyps, der erstellt wird, umgesetzt werden.

### 4.1.2 Auswahl der gewählten Technologie und des Frameworks

Dieser Abschnitt beschäftigt sich mit der gewählten Technologie und dem eingesetzten Frameworks. Auf die Zielplattformen selbst wird nicht mehr näher eingegangen, da diese zum einen im Abschnitt 2.2 näher erläutert wurden und zum anderen nicht näher berücksichtigt werden müssen, da der Prototyp so entwickelt wird, dass dieser auf allen gängigen Plattformen ausgeführt werden kann.

Bei der Technologieauswahl muss somit berücksichtigt werden, dass der Prototyp auf allen Plattformen funktioniert. Wie bereits im Abschnitt 2.2. erläutert wurde, kann die Applikation entweder nativ oder für alle drei Plattformen, hybrid, oder als mobile Web-App umgesetzt werden. Bei der Auswahl muss ebenfalls berücksichtigt werden, welche nativen Funktionalitäten für die Entwicklung benötigt werden. Da die Applikation plattformunabhängig entwickelt werden soll und die Anwendung nicht als Web-App fungieren soll, bleiben nur mehr wenige Technologien im Rennen. Da der Prototyp einige gerätespezifischen Features wie einen GPS Sensors, einer Kartenansicht und ein Notificationsystem benötigen, wird Xamarin als Technologie eingesetzt. Diese Technologie bietet den sehr großen Vorteil, dass eine gemeinsame Codebasis für alle drei Plattformen geschaffen wird, und mittels Plugins gerätespezifische Features angesprochen werden können. Diese sogenannten Erweiterungen, bietet Xamarin teilweise selber an, oder werden von EntwicklerInnen entwickelt und der Öffentlichkeit zur Verfügung gestellt.

Damit herausgefunden werden kann, ob die Anforderungen, die im Abschnitt 4.1.1 umgesetzt wurden, mit der Technologie „Xamarin“ umgesetzt werden können wird dieser Entscheidungsprozess grafisch dargestellt

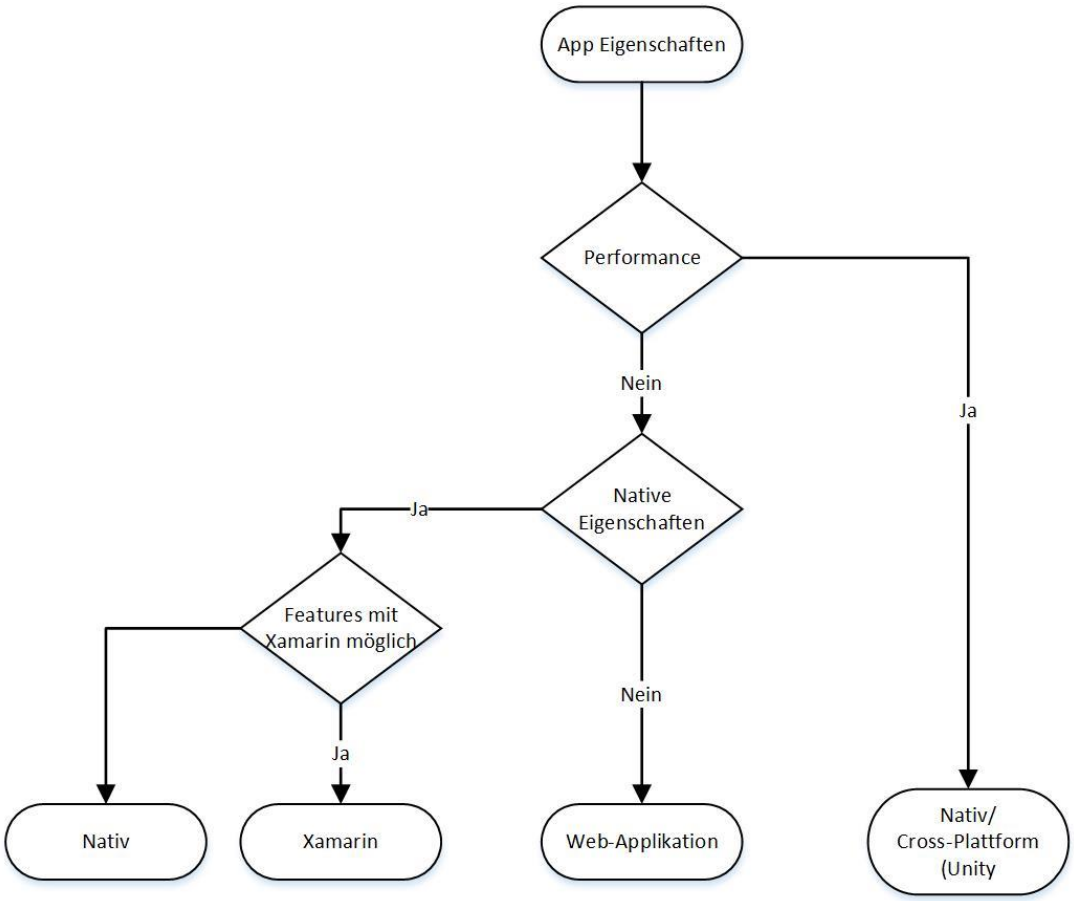



*Abbildung 4-1: Entscheidungsprozess*

Da die gerätespezifischen Anforderungen wie beispielsweise GPS oder ein Notificationservice, mittels Xamarin umgesetzt werden können, wird der Prototyp, mittels dieser Technologie umgesetzt werden und im folgenden Abschnitt näher beschrieben. Aufbauend auf Xamarin wird zusätzlich der Dienst von „HockeyApp“ verwendet. Dies ermöglicht es, den Prototyp über diese Plattform an die TesterInnen zu verteilen, ohne dass ein eigenes Entwicklerkonto bei Google, Apple oder Microsoft angelegt wird. Somit muss von den Testpersonen lediglich die App „HockeyApp“ installiert werden um den Prototyp auf den Geräten ausrollen zu können. Die Applikation übernimmt auch die Aktualisierung des Prototyps.

### 4.1.3 Umsetzung der Anforderungen

In diesem Abschnitt erfolgt die Dokumentation der Umsetzung der Anforderungen A01 bis A07, die im Abschnitt 4.1.1 näher beschrieben wurden. Die Screenshots die dem Lesendem nun vorgestellt werden, zeigen wie der Prototyp auf einem LG G6, einem iPhone 6s und einem Windows Phone Lumia 820 aussehen. Die definierten Anforderungen werden zum Teil gemeinsam und nicht in aufsteigender Reihenfolge erläutert, da gewisse Anforderungen thematisch zusammengehören oder aufgrund der Umsetzung in einem Screenshot besser erläutert werden können.

Bei diesem Prototyp handelt es sich zwar in Hinsicht der plattformunabhängigen Softwareentwicklung um einen experimentellen Prototypen, aber es wurden bereits erste

Livedaten eingebunden und beim Testen verwendet. Da es sich hier um eine Applikation für die freiwillige Feuerwehr Seiersberg handelt, muss die Software auch nicht in andere Sprachen übersetzt werden. Da es sich bei den im Folgendem gezeigten Screenshots bereits um echte Daten handelt, müssen diese teilweise zensiert werden. Somit entsprechen diese Daten und deren Informationsgehalt realen Gegebenheiten.

Zusätzlich zu den gezeigten Screenshots, werden dem Lesendem Code-Snippet gezeigt, damit besser gezeigt werden kann, welche Vorteile eine plattformunabhängige Softwareentwicklung mit sich bringen kann.

**A01 – Einsatz empfangen**

Die Applikation bietet die Möglichkeit, Einsätze über den bekannten Firebaseservice von Google, zu empfangen. Über diese verschlüsselte Technologie, werden die Informationen des neuen Einsatzes an das Smartphone gepusht und auf nativer Ebene weiterverarbeitet. Damit diese Anforderung umgesetzt werden kann, ist entweder Entwicklung auf nativer Ebene notwendig oder andererseits muss auf ein Plugin zurückgegriffen werden. Wenn vom mobilen Endgerät ein Einsatz empfangen wurde, und das App geöffnet wird, soll den BenutzerInnen eine Nachricht angezeigt werden, dass ein neuer Einsatz zur Verfügung steht und die Möglichkeit geboten werden, zu diesem Einsatz zu navigieren.

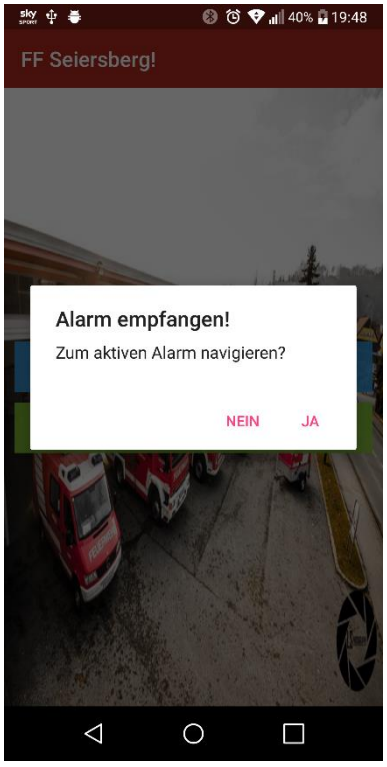



*Abbildung 4-4: Android neuer Alarm*


*Abbildung 4-3: iOS neuer Alarm*


*Abbildung 4-2: UWP neuer Alarm*

Wenn die AnwenderInnen diese Nachricht mit „Ja“ bestätigen, wird direkt die Detailansicht des betroffenen Einsatzes angezeigt.

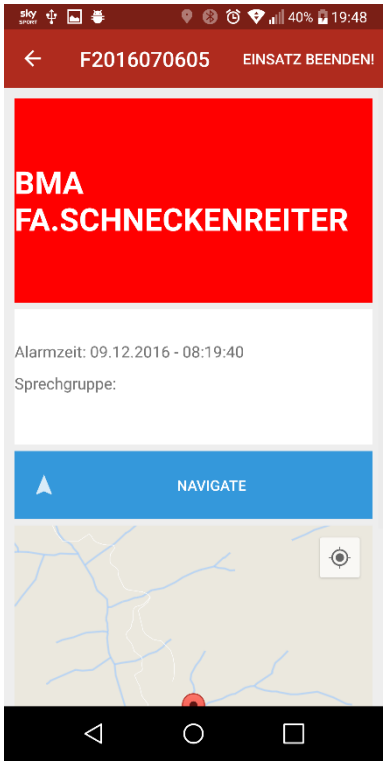



*Abbildung 4-7: Android neuer Alarm 2*

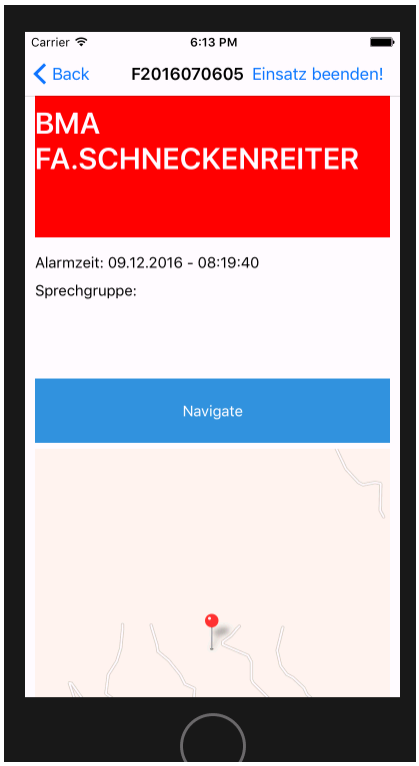



*Abbildung 4-6: iOS neuer Alarm 2*


*Abbildung 4-5: UWP neuer Alarm 2*

Diese Meldung wird den BenutzerInnen solange angezeigt, bis der Einsatz in der Detailansicht mit „Einsatz beenden“ beendet wurde.

**A02 – Detaillierte Einsatzübersicht, Einsatz, A06 – Einsatz auf Karte anzeigen, A07 – horizontal und vertikal bedienbar**

In der Detailansicht wird der aktuelle Einsatz, mit den wichtigsten Informationen wie zum Beispiel den Einsatzgruppen, angezeigt. Neben diesen Informationen wird dem Benutzer, der Benutzerin eine kleine Karte, mit dem Standort des aktuellen Einsatzes gezeigt. Zusätzlich haben die AnwenderInnen die Möglichkeit den aktuellen Einsatz zu beenden. Neben diesen Eigenschaften wurde auch berücksichtigt, dass die mobilen Endgeräte in horizontaler und in vertikaler Position gehalten werden können, und somit für beide Ansichten ein eigens Interface benötigen. Da diese Anforderung auf allen drei Plattformen verfügbar sein sollte.

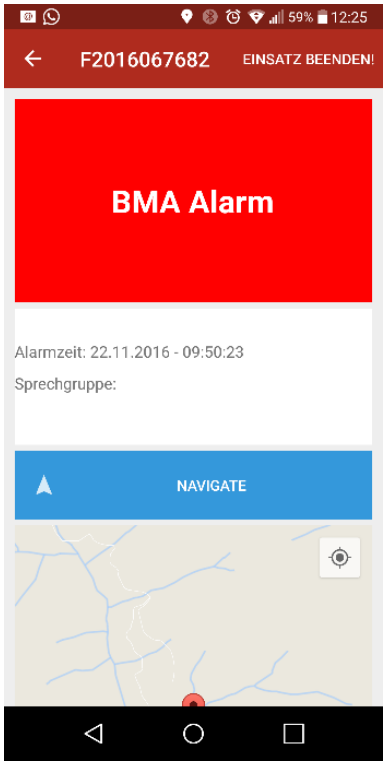



*Abbildung 4-10: Android Detailansicht*

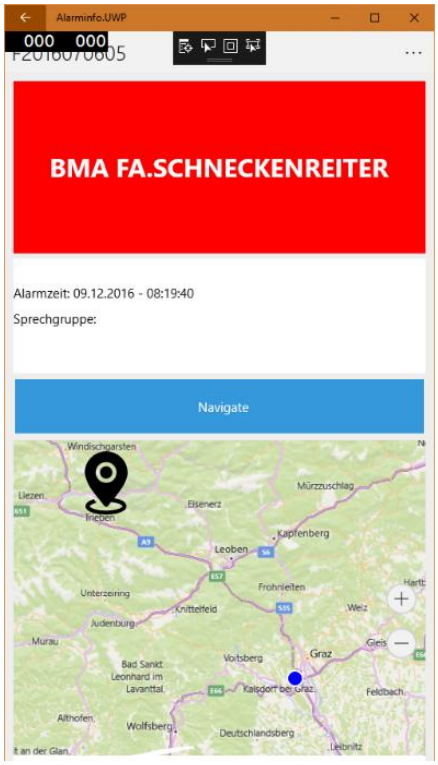



*Abbildung 4-9: UWP Detailansicht*

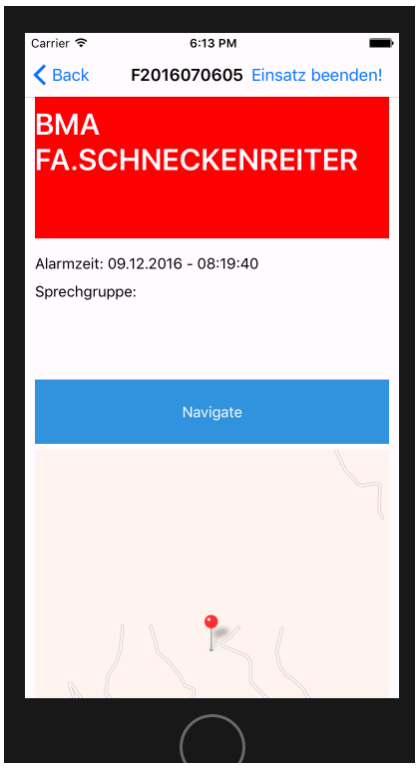



*Abbildung 4-8: iOS Detailansicht*

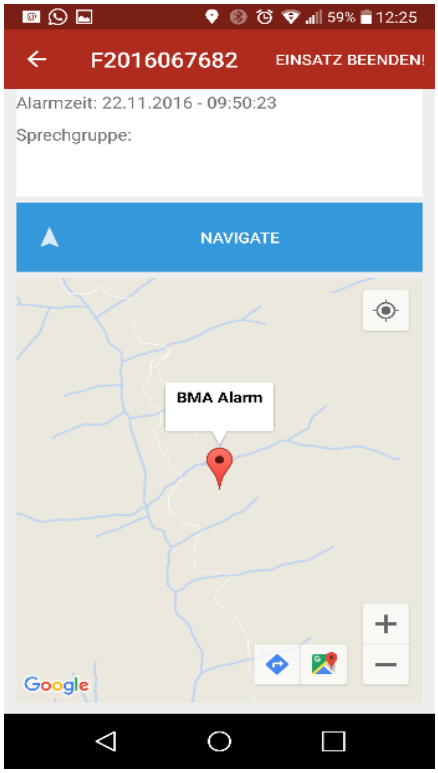



*Abbildung 4-12: Android Detailansicht 2*

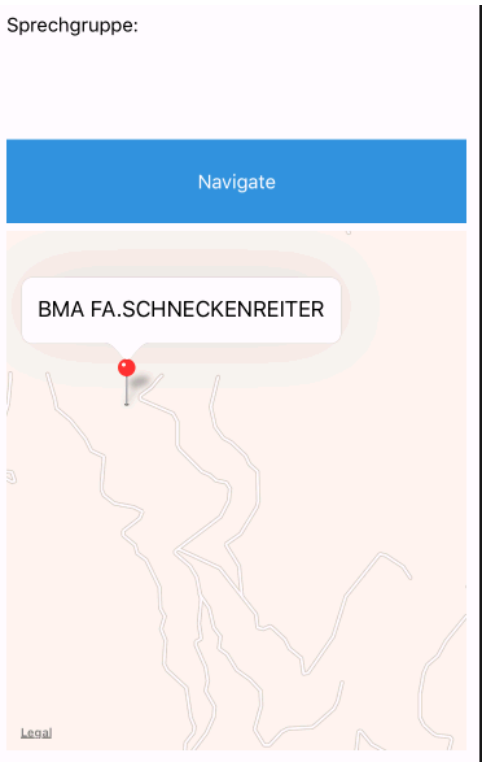



*Abbildung 4-11: iOS Detailansicht 2*

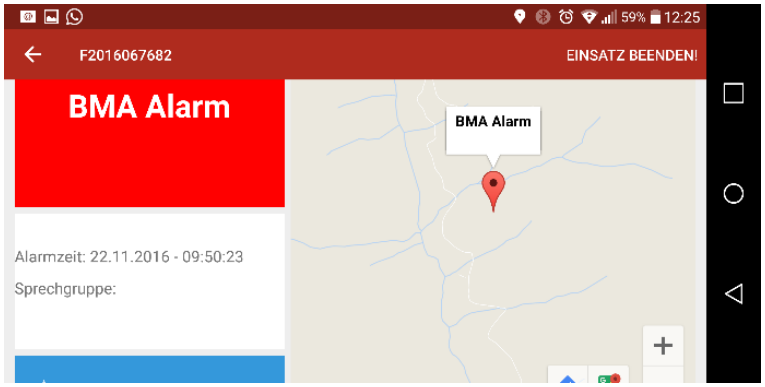



*Abbildung 4-13: Android Detailansicht horizontal*

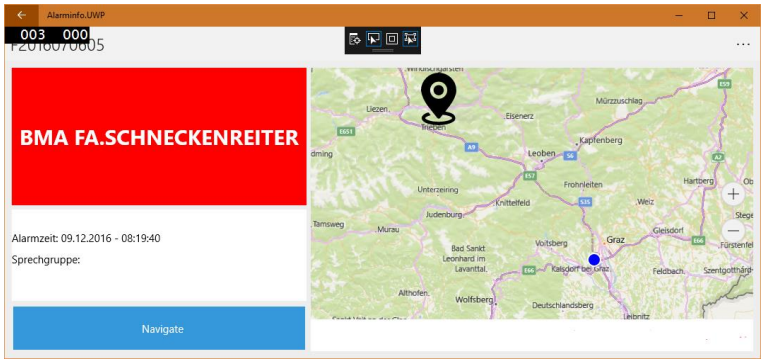



*Abbildung 4-14: UWP Detailansicht horizontal*

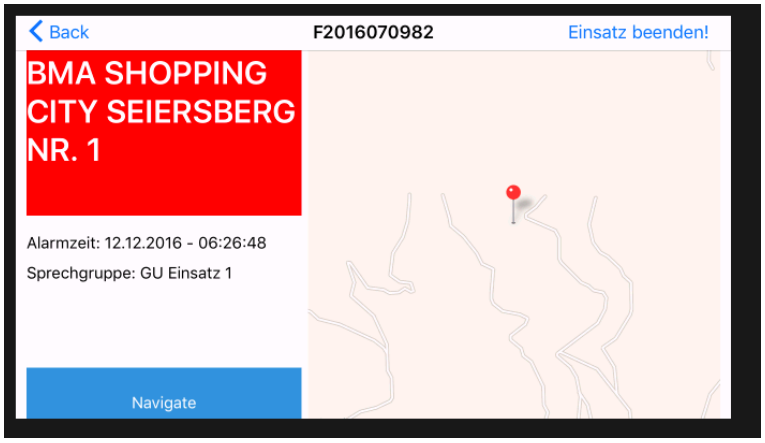



*Abbildung 4-15: iOS Detailansicht horizontal*

### A03 – Einsatz beenden

In der Detailansicht kann der aktuelle Einsatz durch die BenutzerInnen beendet werden. Diese Funktionalität kann über den Button in der Navigationsleiste aufgerufen werden. Nach Betätigung des Buttons „Einsatz beenden“, wird dem Anwender, der Anwenderin, eine Nachricht angezeigt, die bestätigt, dass der Einsatz beendet wurde.

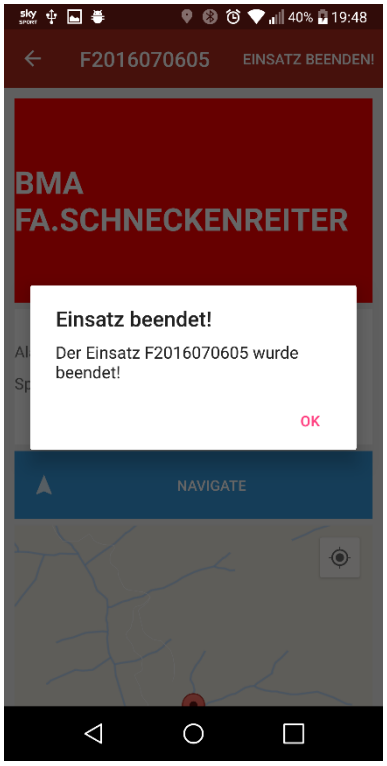



*Abbildung 4-18: Android Einsatz beenden*

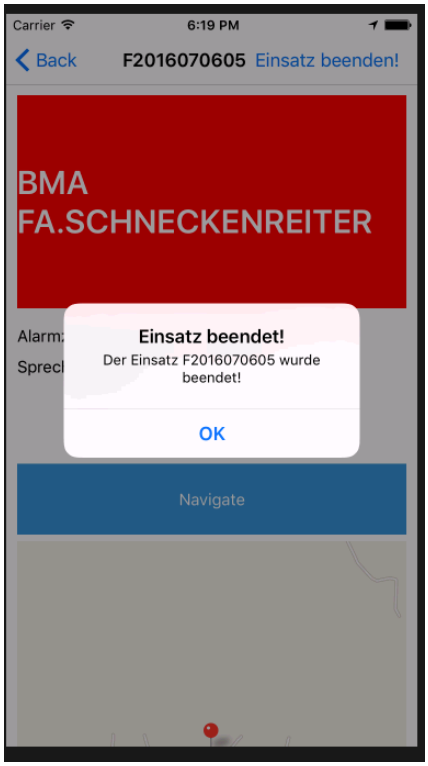



*Abbildung 4-16: iOS Einsatz beenden*

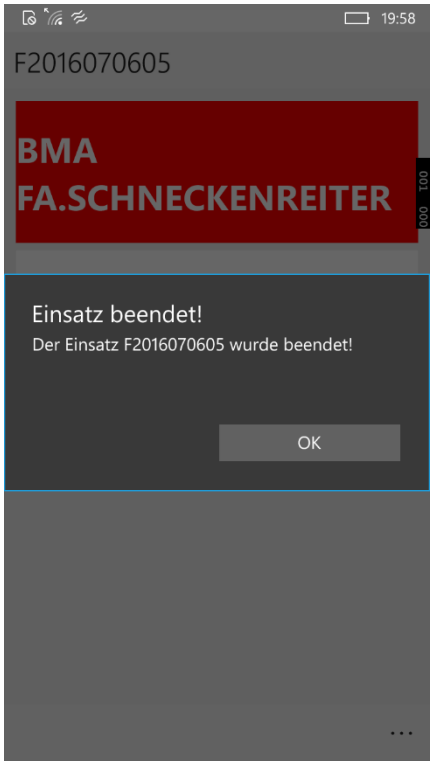



*Abbildung 4-17: UWP Einsatz beenden*

### A04 – Navigation zu aktuellem Einsatz

Neben den Informationen und der kleinen Karte hat der Benutzer, die BenutzerIn auch die Möglichkeit, direkt aus der Applikation eine Navigation zu starten. Dafür muss der Button „Navigation“ gedrückt werden. Unter Android öffnet sich in weiterer Folge ein kleines Fenster, indem ausgewählt werden muss, welche zur Verfügung stehende Applikation, zur Navigation ausgewählt werden soll.

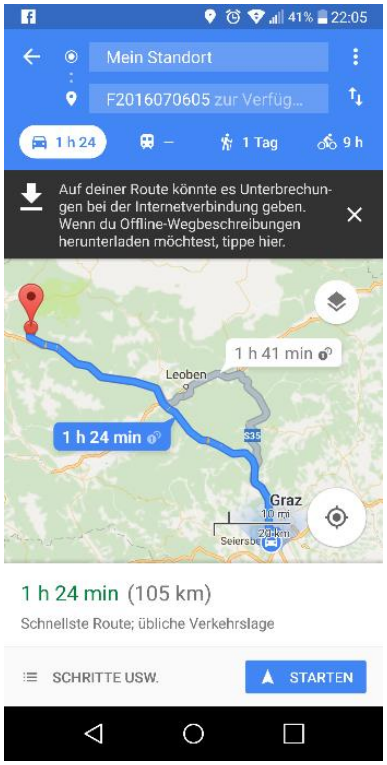



*Abbildung 4-21: Android Navigation*

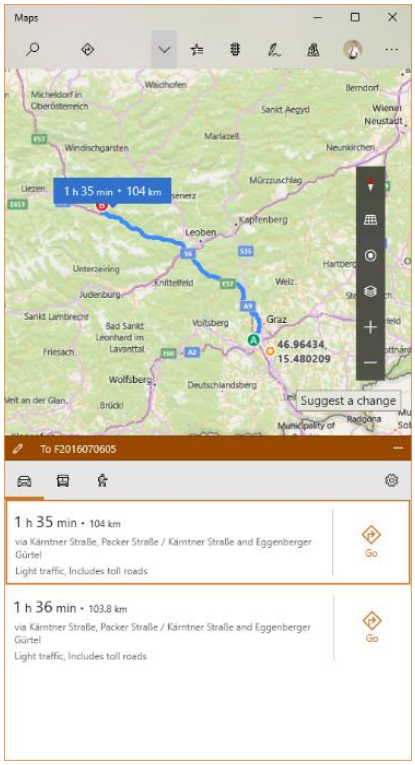



*Abbildung 4-20: UWP Navigation*

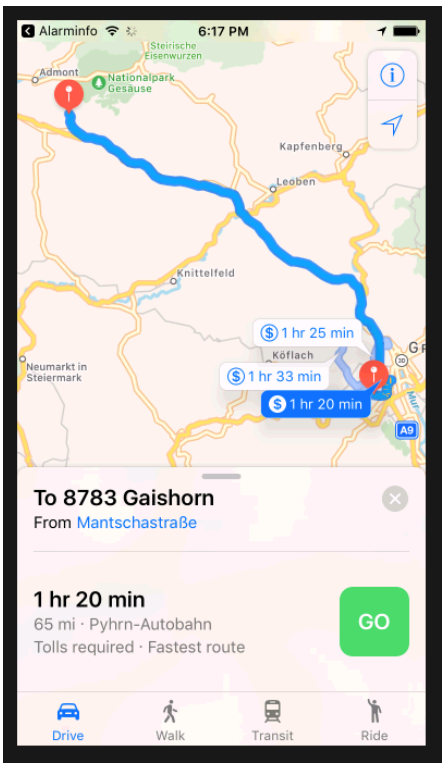



*Abbildung 4-19: iOS Navigation*

### A05 – Liste der Einsätze anzeigen

Eine weitere wichtige Ansicht, die den BenutzerInnen zur Verfügung gestellt wird, ist eine Liste der abgeschlossenen Einsätze. Die Unterscheidung der technischen- und Feuereinsätzen wird mittels eines kleinen Balkens auf der rechten Seite bestimmt. Zusätzlich wird der Name des Einsatzes und das Datum angezeigt. Wenn in dieser Liste ein Einsatz selektiert wird, öffnet sich die Detailansicht mit dem gewählten Einsatz und es können somit alle weiteren Informationen eingesehen werden.


*Abbildung 4-24: Android Einsatzliste*

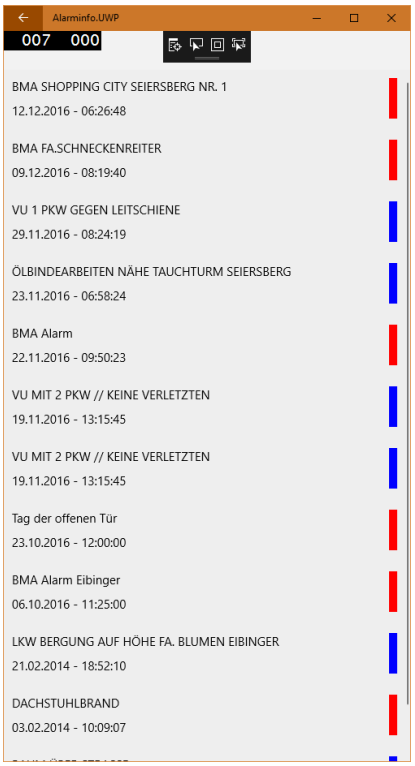



*Abbildung 4-22: UWP Einsatzliste*

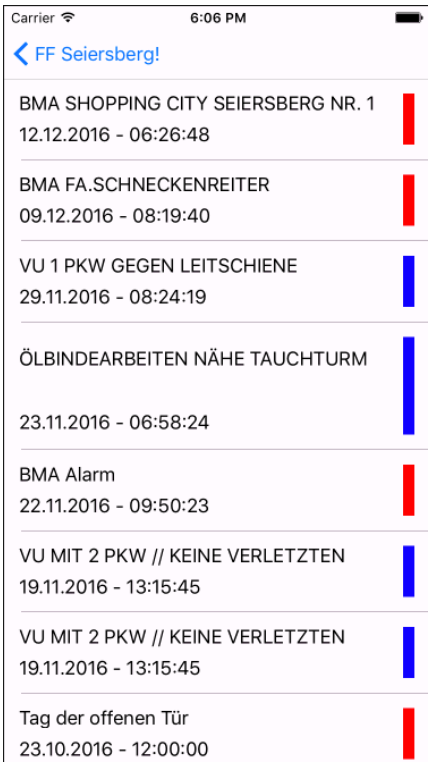



*Abbildung 4-23: iOS Einsatzliste*

Eine weitere wichtige Ansicht die keine direkte Anforderung stellt ist das Einstellungsmenü. Auf dieser Seite haben die BenutzerInnen die Möglichkeit den aktuellen Server zu ändern und den Token an Administrator zu senden. Dieser Token ist notwendig, damit das mobile Endgerät in weiterer Folge Notifications vom Server zugeschickt bekommt.

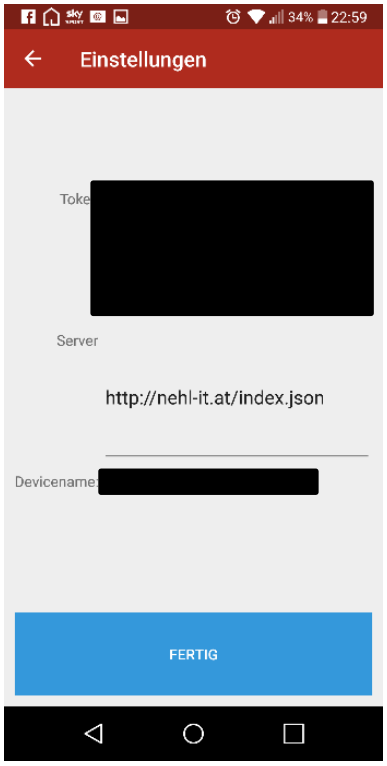



*Abbildung 4-27: Android Einstellungen*


*Abbildung 4-26: UWP Einstellungen*


*Abbildung 4-25: iOS Einstellungen*

### 4.1.4 Bewertung des Prototyps

Im letzten Abschnittes dieses Unterkapitels wird die Bewertung des Prototyps durchgeführt. Hierbei wird vor allem der Fokus auf die Entwicklung der App gelegt und kurz zusammengefasst welche Erfahrungen der Entwicklung dieser Software sammeln konnte und welche Blockaden aufgetreten sind.

Eine Businessanwendung mit Anforderungen, wie es bei diesem Prototyp der Fall war, kann mittlerweile auf sehr einfache Art und Weise plattformunabhängig umgesetzt werden, ohne dass große Herausforderungen bewältigt werden müssen. Die größten Hürden, die beim Entwickeln dieses Prototyps entstanden sind, waren jene, dass Abhängigkeiten zu mehreren Personen bzw. Organisationen entstanden sind und somit bei Änderungen viele Nacharbeiten getätigt werden mussten. Um diese Problematik dem Lesenden kurz aufzeigen zu können werden zwei große Herausforderungen näher beschrieben. Wie bereits im Abschnitt 4.1.3 näher beschrieben wurde, gibt es bei Xamarin eine große Anzahl an sogenannten PlugIns die zum Einsatz kommen, wenn gewisse gerätespezifischen Features benötigt werden. So wurde beispielsweise das „SettingsPlugin“ von James Montemagno verwendet, welches die Möglichkeit bietet Daten wie zum Beispiel ob ein aktiver Alarm vorhanden ist, gespeichert werden können. Damit diese Funktionalität zentral verwendet werden kann und in weiterer Folge auf allen Plattformen funktioniert, muss diese Plugin installiert werden. Somit muss der native Zugriff auf den Telefonspeicher nicht extra implementiert werden. Das Problem an solchen Erweiterungen ist jenes, dass bei großen Änderungen viele Nacharbeiten gemacht werden müssen. Da vor kurzem die Architektur des Plugins auf Dot-net Standard 2.0 umgestellt wurde, wurden einige

Funktionalitäten geändert und mussten im Prototypen nachgezogen werden (Montemagno, SettingsPlugin, 2017).

Eine weitere Problematik, die in der Entwicklungsphase dieses Prototyps aufgetreten ist, war jene der Änderung des Bing-Dienstes. Dieser Dienst wird benötigt, wenn UWP Applikationen eine Kartenansicht in der App anzeigen sollen. Da Microsoft vor kurzer Zeit ein neues Portal eingeführt hat und in diesem Zug das veraltete Bing-Portal deaktiviert hat, musste dies für UWP neu implementiert werden. Diese Änderungen treffen aber nicht nur EntwicklerInnen die eine Cross-Plattform-Anwendung entwickeln, sondern alle Developer die diesen Bing-Service nutzen.

Diese beiden Beispiele zeigen sehr gut, wie problematisch Abhängigkeiten zu Dritten sein können. Aus diesem Grund muss immer darauf geachtet werden, ob es beispielsweise sinnvoll ist eine Erweiterung zu installieren, oder diese Eigenschaft selbst, nativ, für alle drei Plattformen zu entwickeln. Bei Änderungen, wie jener von Microsoft, müssen die EntwicklerInnen die Applikation aktualisieren oder auf den Dienst verzichten.

## 4.2 Experteninterview

In diesem Abschnitt erfolgt die Beschreibung der durchgeführten Experteninterviews. Im Rahmen dieser Interviews soll herausgefunden werden, inwieweit andere Software-EntwicklerInnen Erfahrungen im Bereich der Cross-Plattform-Entwicklung sammeln konnten.

Zu Beginn wird dem Lesendem eine kurze theoretische Einführung in den Bereich der Interviews vorgestellt. Dabei werden die Unterschiede zwischen einer qualitativen und einer quantitativen Befragung beschrieben. Anschließend werden der Aufbau des Experteninterviews und die zu untersuchendende Hauptfragestellung näher beschrieben. Danach werden die Antworten analysiert und kurz zusammengefasst. Die Ergebnisse dieses Abschnitts und der Erfahrungsbericht des Autors dieser Arbeit bilden die Basis für die Beantwortung der Forschungsfrage.

Neben der Forschungsfrage wurden auch Hypothesen aufgestellt die im Zuge dieser Arbeit bestätigt beziehungsweise wiederlegt werden sollen. Diese Hypothesen lauten

- H1: Durch die Entwicklung einer Cross-Plattform-Applikation kann die Effizienz in Hinblick auf die Entwicklung gesteigert werden.
- H0: Die Entwicklung mit einer Cross-Plattform-Technologie hat keinen Einfluss auf die Effizienz.
- H1: Durch die Entwicklung einer Cross-Plattform-Applikationen müssen gewisse Kompromisse eingegangen werden, da nicht alle Funktionalitäten abgebildet werden können.
- H0: Bei der Entwicklung einer Cross-Plattform-Applikation müssen keine Kompromisse eingegangen werden, da jegliche Funktionalitäten die nativ umgesetzt werden können, auch mit einer Cross-Plattform-Applikation umgesetzt werden können.

### 4.2.1 Recherche der einzelnen Interviewarten

In diesem Abschnitt wird ein Überblick über Interviews als wissenschaftliche Befragungsmethodik gegeben. Zu Beginn werden die qualitativen und quantitativen Befragungen im Allgemeinen vorgestellt und wie diese sich unterscheiden. Nach dieser Einführung werden die verschiedenen Arten eines Interviews angeführt und kurz erläutert. Danach werden ein paar wesentliche Rahmenbedingungen, die bei Interviews erfüllt werden müssen, näher erläutert.

**Quantitative Befragung**

Ziel der quantitativen Befragung ist jenes, eine möglichst große Anzahl von Personen zu befragen. Hierfür werden in der Regel standardisierte Methoden eingesetzt. Das heißt, dass die Befragten nicht frei angeben können, was ihnen wichtig erscheint, sondern müssen Raster ausfüllen. Beispiele dafür wären Fragebögen, bei denen die Zielpersonen ankreuzen müssen wie sehr ein gewisser Bereich gefallen hat oder nicht.

**Qualitative Befragung**

Bei qualitativen Befragungen hingegen werden sogenannte Experten befragt und Ziel ist es, dass nicht eine standardisierte Untersuchung durchgeführt wird. So werden hierbei meist offene Fragen gestellt und die Befragten können weitgehend frei erzählen. Erhebungstechniken die hierbei eingesetzt werden sind beispielsweise Experteninterviews, teilnehmende Beobachtungen oder Gruppendiskussionen.

Da es sich bei dieser Arbeit um ein sehr spezielles Thema handelt und diese Problematik zwar sehr verbreitet ist, aber meist nicht in dieser Form gelöst wird, ist eine quantitative Befragung nur sehr schwer möglich, da nicht die notwendige Masse an TeilnehmerInnen gefunden werden kann. Stattdessen werden einige Experten interviewt werden, damit diese über ihre Erfahrungen und Probleme erzählen können.

In der Literatur gibt es für diese Ausgangssituation unterschiedliche Ansätze, wie die verschiedenen Interviewarten unterschieden werden können. Eine sehr weitverbreite Unterscheidung unterteilt die Interviews in folgende drei Arten (Marquardt, 2007). Die folgende Aufzählung wurde aus der ursprünglichen Quelle übernommen:

- Wenig strukturiertes Interview. Bei dieser Art der Befragung hat der Interviewer oder die Interviewerin keinen oder nur einen sehr groben Leitfaden. Somit kann die gesamte Befragung sehr stark in die Tiefe und Breite gehen und entspricht eher einer offenen Konversation. Der Interviewer oder die Interviewer haben die Möglichkeit den gesamten Verlauf des Gesprächs individuell zu steuern, sodass sich die jeweils nächste Frage aus der Antwort der vorherigen Frage ergeben kann.
- Stark strukturiertes Interview. Bei dieser Form handelt es sich um eine Befragung, bei der ein Fragebogen zur Verfügung steht. Die Anzahl der Fragen steht bereits im Vorhinein fest. Außerdem werden auch Inhalt und Reihenfolge der Fragen vorab bestimmt. Somit hat der Interviewer, die Interviewerin nicht die Möglichkeiten flexibel auf gewissen Antworten des Interviewpartners einzugehen.

- Teilstrukturiertes Interview. Hierbei handelt es sich um Mischform. Es können vorgefertigte Fragen in Form eines Leitfadens verwendet werden. Diese Fragen können bereits ausformuliert sein, müssen jedoch nicht in einer bestimmten Reihenfolge abgearbeitet werden. Der große Vorteil gegenüber dem stark strukturiertem Interviews ist jener, dass weitere Themen im Verlauf des Gesprächs mitaufgenommen und auch vertieft werden können.

Neben diesen vorgestellten Interviewformen gibt es noch eine vierte sehr verbreitete Form des Interviews. Hierbei handelt es sich um ein sogenanntes Experteninterview. Bei dieser Art des Interviews werden Personen herangezogen, die in einem gewissen Bereich als ExpertInnen bezeichnet werden. Es eignet sich vor allem dann, wenn für eine bestimmte Problemstellung die Sichtweise eines Experten oder einer Expertin miteinbezogen werden soll (Bogner, Littig, & Menz, 2013).

Die Definition welche Personen als Experten bezeichnet werden, wird in der Literatur nicht eindeutig definiert. Die Definition von Dr. Richard Lackes (Lackes, 2017) baut der beruflichen Position oder der beruflichen Stellung der Experten. Hierbei werden Personen als Experten bezeichnet, die wegen dieses Berufs über Wissen besonderer Art in einem speziellen Bereich verfügen.

Spektrum.de definiert den Begriff wie folgt (Spektrum.de, 2000):

> *„Expertenwissen, E expert knowledge, Bezeichnung für das Wissen, über das Personen (Experten) verfügen, die seit Jahren auf einem bestimmten Gebiet arbeiten. Das Expertenwissen besteht sowohl aus dem spezifischen Fachwissen (aus Büchern erlernbar) als auch aus Erfahrungswissen (wird nur im Laufe der Tätigkeitsausführung erworben). Es zeichnet sich durch eine besondere Art der kognitiven Gliederung aus: das Fachwissen wird dabei nicht mehr nur nach dem taxonomischen Aufbau gegliedert, sondern auch nach anderen Kriterien, die durch erworbene Erfahrungen Bedeutung gewonnen haben. - In der künstlichen Intelligenz wird Expertenwissen als Grundlage für die Erstellung von Expertensystemen verwendet.“*

Die vorliegende Arbeit bezeichnet, ähnlich wie die oben angeführten Quellen, jene Personen als Experten, die in einem speziellen Gebiet berufliche Erfahrungen sammeln konnten, oder in diesem Fachgebiet Veröffentlichungen nachweisen können.

Da für die Beantwortung der Forschungsfrage Personen interviewt werden sollten, die in diesem Bereich Erfahrungen sammeln konnten, wird hierfür das sogenannte Experteninterview durchgeführt werden. Als Expertinnen und Experten werden SoftwareenwicklerInnen befragt, die bereits unterschiedlichste Erfahrungen im Bereich der plattformunabhängigen Softwareentwicklung sammeln konnten.

### 4.2.2 Konzept und Aufbau

Die durchgeführten Experteninterviews erfolgten bei sogenannten Meetups in Graz. Diese Meetups werden von einer kleinen Developer-Group aus Graz, bei der der Autor dieser Arbeit

selbst Mitglied ist, veranstaltet. Auch hier stand im Jahre 2017 bei jedem Treffen die Thematik der plattformunabhängigen Softwareentwicklung im Fokus. Beim Gespräch selbst wird zu Beginn versucht mit den befragten Teilnehmerinnen etwas Smalltalk zu betreiben, damit eine entspannte Gesprächsatmosphäre geschaffen werden kann. Anschließend wird den ExpertInnen erklärt, was das Ziel der Befragung ist und wofür diese Technologien zum Einsatz kommen können.

Für die anschließende Befragung wird ein zuvor zusammengestellter Leitfaden verwendet um eine gewisse Struktur für das Interview zu haben damit die Befragung nicht vom Thema abschweift. Somit wird hier auf die Form des teilstrukturierten Interviews zurückgegriffen. In diesem Leitfaden sind die Hauptfragestellungen mit den Themen notiert, die in der Befragung angesprochen werden sollen. Die einzelnen Fragen ergeben sich aus der theoretischen Auseinandersetzung mit Cross-Plattform-Entwicklung. Die im Leitfaden erstellten Fragen, müssen nicht alle zwingend beantwortet werden, da gewisse Fragen vom Ablauf des Interviews abhängig sind.

Damit den einzelnen Befragten eine gewisse Hilfestellung geboten werden kann, werden zur jeder Frage vordefinierte Antwortmöglichkeiten zur Verfügung gestellt, die aus den Vorarbeiten in dieser Arbeit abgeleitet werden. Da es sich bei der Befragung um ein Experteninterview mit offener Fragestellung handelt und nur eine geringe Anzahl an Interviewpartnern gewählt wird, birgt dies die Gefahr, dass aus der Befragung keine verwendbaren Ergebnisse abgeleitet werden können. Zum anderen stellen die vordefinierten Antwortmöglichkeiten einfache Optionen dar, um die TeilnehmerInnen für weitere Ideen inspirieren zu können. Neben diesen Antwortmöglichkeiten, wird den Befragten abschließend eine offene Frage gestellt, um weitere oder tiefergreifende Einblicke in die Meinung der ExpertInnen zu bekommen.

Die im Leitfaden ausgearbeiteten Fragestellungen unterteilen sich in zwei Teilbereiche auf. Der erste Teilbereich betrifft grundsätzlich die Thematik der plattformunabhängigen Softwareentwicklung. Beim zweiten Teilbereich liegt der Fokus speziell auf den Erfahrungen, die die TeilnehmerInnen, in diesem Bereich, sammeln konnten. Zusammenfassend ergeben sich also folgende Hauptfragestellungen

1. Was spricht für eine plattformunabhängige Softwareentwicklung?
2. Welche Einstiegsbarrieren sind für Unternehmen vorhanden?
3. Haben Sie in Ihrem Unternehmen bereits eine Business-Applikation mit einer Cross-Plattform-Technologie entwickelt?
4. Welche Erfahrungen haben Sie selbst in diesem Bereich sammeln können?
5. Welche Erfahrungen konnten Sie bereits mit Xamarin sammeln?

Da es sich bei der Form des Interviews um ein teilstrukturiertes Interview handelt, spielt die Nummerierung der Fragen keine konkrete Rolle. Diese dienen lediglich dem einfacheren Referenzieren im nachfolgenden Text, da auf einzelne Fragen und deren vordefinierte Antwortmöglichkeiten konkreter eingegangen wird.

Das Interview selbst wird mit allen TeilnehmerInnen Vorort und unter vier Augen mit dem Autor dieser Arbeit, als Interviewer, durchgeführt. Zu Beginn der Befragung werden die befragten

Personen darauf aufmerksam gemacht, dass die Teilnahme freiwillig und anonymisiert erfolgt. Damit das Gesprächsklima gelockert wird, wird zu Beginn über das Vorhaben des Autors dieser Arbeit berichtet und somit bekommen die TeilnehmerInnen einen kurzen Überblick über die Thematik der Befragung.

**Fragestellung 1 - Was spricht für eine plattformunabhängige Softwareentwicklung?**

Über diese Frage wird ermittelt, wie die befragte Person plattformunabhängige Softwareentwicklung interpretiert. So kann herausgefunden werden, ob die TeilnehmerInnen unter Cross-Plattform-Entwicklung eine ähnliche Vorstellung haben, wie diese zu Beginn dieser Arbeit definiert wurde. Das Weitern wird herausgefunden, ob die befragten Personen diese Thematik unterstützen oder ob sie der Meinung sind, dass diese Technologie nur ein sogenannter Hype ist. Die vordefinierten Antwortmöglichkeiten lauten wie folgt:

- Durch diese Technologie haben EntwicklerInnen den Vorteil, dass eine gemeinsame Codebasis geschaffen werden kann.
- Es besteht die Möglichkeit mit einer entwickelten Applikation eine Vielzahl an AnwenderInnen zu erreichen.
- Die Applikation hat auf jeder entwickelten Plattform die gleichen Funktionalitäten, da diese für alle Plattformen zur gleichen Zeit entwickelt werden können.

**Fragestellung 2 - Welche Einstiegsbarrieren sind für Unternehmen vorhanden?**

Mit Hilfe der zweiten Frage wird ermittelt, ob Unternehmen großen Aufwand betreiben müssen, um eine Cross-Plattform-Applikation zu entwickeln beziehungsweise ob diese Art der Entwicklung leicht umsetzbar ist. Durch die Beantwortung dieser Frage kann ebenfalls ermittelt werden ob die teilnehmende Person bereits positive beziehungsweise negative Erfahrung sammeln konnte. Die vordefinierten Antwortmöglichkeiten lauten:

- Die meisten Technologien wie beispielsweise Xamarin können von Unternehmen kostenlos benutzt werden somit gibt es in finanzieller Hinsicht keine Barrieren.
- Viele Technologien haben spezielle Eigenschaften, somit wird spezielles Fachwissen benötigt.
- Unternehmen entscheiden sich meist für eine Testplattform um zu sehen ob die Applikation einen Mehrwert mit sich bringt. Danach ist für viele der Umstieg auf eine Cross-Plattform-Technologie zu aufwändig.

**Fragestellung 3 - Haben Sie in Ihrem Unternehmen bereits eine Business-Applikation mit einer Cross-Plattform-Technologie entwickelt?**

Durch diese Frage soll herausgefunden werden, ob die Experten und Expertinnen bereits im Unternehmen Erfahrungen sammeln können. Mit Hilfe dieser Antworten, kann in weiterer Folge herausgefunden werden, ob Unternehmen bereits auf plattformunabhängige Softwareentwicklung setzen beziehungsweise offen für neue Technologien sind. Diese Fragestellung wird eingegrenzt, indem die Befragten im Vorfeld aufgeklärt werden, dass sich

diese – wie auch alle anderen Fragen – auf den Bereich der mobilen Softwareentwicklung bezieht. Folgende Teilfragen mit jeweils vordefinierten Antworten werden gestellt:

- Welche Funktionalität hat ihre Cross-Plattform-Applikation?
    - Mit Hilfe dieser App haben unsere Kunden die Möglichkeit, persönliche Daten abzurufen, oder ihren Berater direkt zu kontaktieren
- Wie haben Sie die Applikation umgesetzt
    - Hybrid App.
    - native Applikation (Unity, Xamarin).
- Benötigt ihre App eine hohe Performance?
    - Nein, da es sich um eine einfache Business Applikation handelt, die Daten abruft und Daten an den Server schickt.
    - Ja, wir entwickeln Cross-Plattform-Spiele.

**Fragestellung 4 - Welche Erfahrungen haben Sie selbst in diesem Bereich sammeln können?**

Über diese Frage sollen die Erfahrungen, die die Teilnehmenden, bereits sammeln konnten in Erfahrung gebracht werden. Abhängig vom Interesse und dem Vorwissen der Expertin oder des Experten werden mehr oder weniger Teilfragen in einem höheren oder niedrigeren Detailgrad gestellt. Des Weiteren werden die Befragten darum gebeten, zu Berichten welche Funktionalitäten die Applikationen haben, die sie bereits mit einer Technologie wie beispielsweise Xamarin umgesetzt haben. Folgende Teilfragen mit jeweils vordefinierten Antworten werde gestellt:

- Wie wurde die App umgesetzt?
    - Web-App
    - Hybrid Applikation
    - native Xamarin-Applikation
- Wie greifen Sie auf gerätespezifische Features zu?
    - Applikation braucht keine
    - Framework bietet die Möglichkeit auf Features, die wir benötigen zuzugreifen
    - Für native Eigenschaften, verwenden wir Plugin (Eigenentwicklung/Opensource), die es ermöglichen gerätespezifische Features zu verwenden
- Welche Probleme traten bei vergangenen Projekten auf?
    - Frameworkupdates von Herstellern bremsten das Release
    - Entwicklungsumgebung oft noch in einer Betaphase

- Hilfestellungen durch Foren (zum Beispiel stackoverflow) nur eingeschränkt vorhanden

**Fragestellung 5 - Welche Erfahrungen konnten Sie bereits mit Xamarin sammeln?**

Mit dieser abschließenden Frage soll herausgefunden werden ob die befragten Personen bereits Erfahrung mit der Technologie "Xamarin" sammeln konnten. Die Beantwortung dieser Frage hat eine große Priorität, da diese Fragestellung direkt mit der Untersuchung dieser Arbeit zu tun hat. Dadurch kann herausgefunden werden ob die Experten und Expertinnen über diese Technologie Erfahrungen berichten können und somit bei der Beantwortung der Forschungsfrage unterstützen können. Da bei dieser Frage die ExpertInnen speziell über die Erfahrung berichten sollen, die sie damit gemacht haben, gibt es für diese Fragestellung keine vorgefertigten Antwortmöglichkeiten.

### 4.2.3 Quantitative Auswertung

Bei der quantitativen Auswertung werden die Antwortmöglichkeiten zu den Haupt- und Teilfragestellungen analysiert. Bei dieser Auswertung wird abgezählt, wie viele der TeilnehmerInnen eine der vordefinierten Antwortmöglichkeiten gewählt hat. Dabei werden diese in zwei Kategorien "Ja" und "Nein" kategorisiert. Sobald mehr als die Hälfte der Befragten, somit drei von 5 ExpertInnen, eine Antwortmöglichkeit mit "Ja" beantworten, wird die jeweilige Option als wichtigen Faktor eingestuft. Die Ergebnisse, die aus der Befragung entstanden sind, werden in Tabelle 4-2 bis Tabelle 4-5 aufgelistet.

| | 1 | 2 | 3 | 4 | 5 | Anzahl | Prozent |
|---|---|---|---|---|---|---|---|
| Durch diese Technologie haben EntwicklerInnen den Vorteil, dass eine gemeinsame Codebasis geschaffen werden kann | Ja | Ja | Ja | Ja | Ja | 5 | 100% |
| Es besteht die Möglichkeit mit einer entwickelten Applikation eine Vielzahl an AnwenderInnen zu erreichen. | Ja | Ja | Ja | Ja | Ja | 5 | 100% |
| Die Applikation hat auf jeder entwickelten Plattform die gleichen Funktionalitäten, da diese für alle Plattformen zur gleichen Zeit entwickelt werden können. | Ja | Ja | Nein | Ja | Ja | 4 | 80% |

*Tabelle 4-2: Antworten zur Frage 1*

| | 1 | 2 | 3 | 4 | 5 | Anzahl | Prozent |
|---|---|---|---|---|---|---|---|
| Die meisten Technologien wie beispielsweise Xamarin können von Unternehmen kostenlos benutzt werden | Ja | Ja | Ja | Ja | Ja | 5 | 100% |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| somit gibt es in finanzieller Hinsicht keine Barrieren. | | | | | | | |
| Viele Technologien haben spezielle Eigenschaften, somit wird spezielles Fachwissen benötigt. | Ja | Ja | Ja | Ja | Nein | 4 | 80% |
| Unternehmen entscheiden sich meist für eine Testplattform um zu sehen ob di Applikation einen Mehrwert mit sich bringt. Danach ist für viele der Umstieg auf eine Cross-Plattform-Technologie zu aufwändig. | Ja | Ja | Ja | Ja | Nein | 4 | 80% |

*Tabelle 4-3: Antworten zur Frage 2*

| | 1 | 2 | 3 | 4 | 5 | Anzahl | Prozent |
|---|---|---|---|---|---|---|---|
| Mit Hilfe dieser App haben unsere Kunden die Möglichkeit, persönliche Daten abzurufen, oder ihren Berater direkt zu kontaktieren. | Ja | Nein | Ja | Ja | Ja | 4 | 80% |
| Hybrid App. | Ja | Nein | Nein | Ja | Ja | 3 | 60% |
| native Applikation (Unity, Xamarin). | Ja | Nein | Ja | Ja | Ja | 5 | 80% |
| Nein, da es sich um eine einfache Business Applikation handelt, die Daten abruft und Daten an den Server schickt. | Ja | Nein | Ja | Ja | Nein | 3 | 60% |
| Ja, wir entwickeln Cross-Plattform-Spiele. | Nein | Nein | Nein | Nein | Ja | 1 | 20% |

*Tabelle 4-4: Antworten zur Frage 3*

| | 1 | 2 | 3 | 4 | 5 | Anzahl | Prozent |
|---|---|---|---|---|---|---|---|
| Web-App | Ja | Ja | Nein | Ja | Ja | 4 | 80% |
| Hybrid Applikation | Ja | Ja | Nein | Nein | Nein | 2 | 40% |
| native Xamarin-Applikation | Ja | Ja | Ja | Ja | Ja | 5 | 100% |
| Applikation braucht keine | Ja | Nein | Nein | Ja | Nein | 2 | 40% |
| Framework bietet die Möglichkeit auf Features, die wir benötigen zuzugreifen | Ja | Ja | Ja | Ja | Ja | 5 | 100% |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Für native Eigenschaften, verwenden wir Plugin (Eigenentwicklung/Opensource), die es ermöglichen gerätespezifische Features zu verwenden | Ja | Ja | Ja | Ja | Ja | 5 | 100% |
| Frameworkupdates von Herstellern bremsten das Release | Ja | Ja | Ja | Ja | Ja | 5 | 100% |
| Entwicklungsumgebung oft noch in einer Betaphase | Ja | Ja | Ja | Ja | Ja | 5 | 100% |
| Hilfestellungen durch Foren (zB stackoverflow) nur eingeschränkt vorhanden | Ja | Ja | Ja | Ja | Ja | 5 | 100% |

*Tabelle 4-5: Antworten zur Frage 4*

### 4.2.4 Qualitative Auswertung der Ergebnisse

Neben der quantitativen Auswertung wird auch eine qualitative Auswertung der Befragung durchgeführt. Wie im Abschnitt 4.2.1 erläutert wird, handelt es sich bei dem Interview um ein leitfadengestütztes Experteninterview mit offenen Fragen. Da bei offenen Fragen keine klassische quantitative Auswertung möglich ist, werden diese Antworten mithilfe des hermeneutischeren Zirkels interpretiert (Heckmann, 1992). Damit ein hermeneutischer Zirkel eingesetzt werden kann muss zuerst ein gewisses Vorwissen für die Thematik aufgebaut werden, damit diese in weiterer Folge richtig interpretiert werden kann.

Gadamer (2010) schreibt, dass der hermeneutische Zirkel dem Verhältnis zwischen der Bedeutungsganzheit eines Textes und eines Bedeutungsteils entspricht. Damit ein Text als Ganzes verstanden werden kann, müssen zuerst der Sinn der einzelnen Teile verstanden werden. Teile und die Ganzheit stehen damit zueinander in einem sogenannten Zirkelverhältnis.

Die wissenschaftliche kontrollierte Interpretation stellt eine höhere Form des Verständnisses dar, die durch den hermeneutischen Zirkel gerechtfertigt wird. Dabei handelt es sich um eine Zirkelbewegung, bei der Einzelelemente nur aus dem Gesamtzusammenhang verständlich sind und das Ganze sich wiederrum nur aus den Teilen ergibt. Damit ein Text verstanden werden kann, muss ein gewisses Vorverständnis vorhanden sein. Somit ist es nicht möglich, dass ein Text ohne gewisses Vorwissen interpretiert werden kann. Sobald eine Wissensbasis aufgebaut wurde, kann der Text analysiert und interpretiert werden. Durch das aufgebaute Verständnis ermöglicht es dem Interpreten ein besseres Textverständnis aufzubauen. "Nur wo der Interpret sich selbst in der Wirklichkeit versteht, die erkannt werden soll, kam, es zu dem Austausch kommen, in dem das Vor-Verständnis in wiederholtem Wechsel von dem Textsinn überwunden wird und die Wahrheit des Textes sich durchsetzt" (Heidegger 1963).

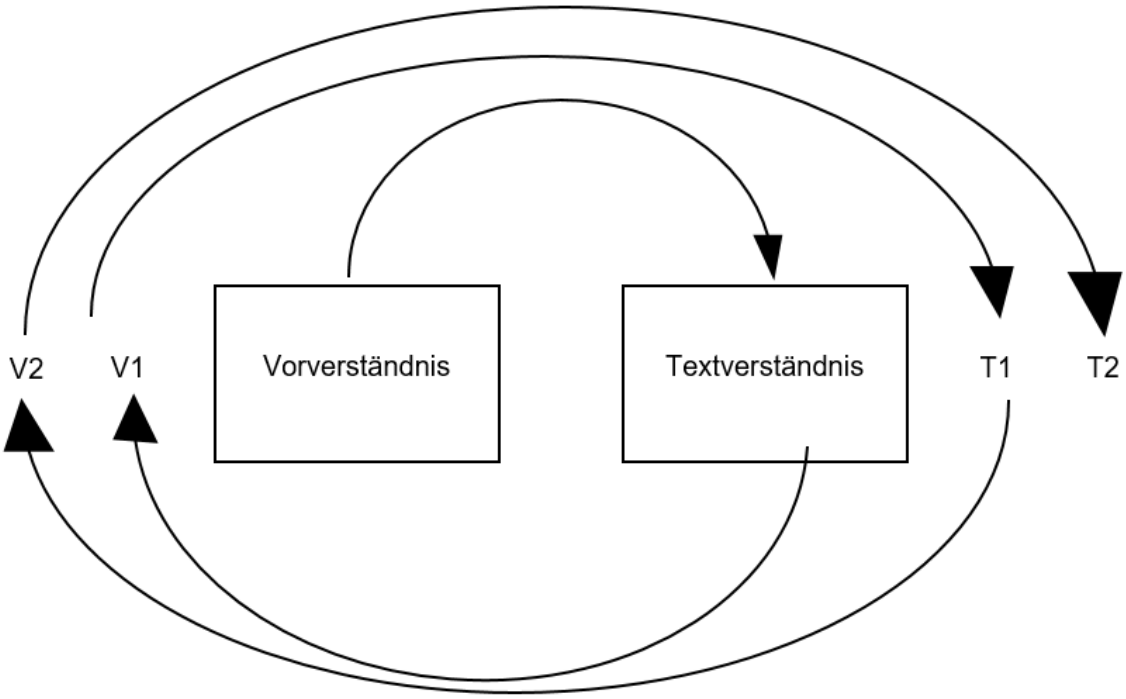



*Abbildung 4-28: hermeneutischer Zirkel (Gadamer, 2010)*

Damit die Ergebnisse, die durch die Interviews entstanden sind, laut des hermeneutischen Zirkels, richtig interpretiert werden können, muss somit ein gewisses Vorverständnis im Bereich der plattformunabhängigen Softwareentwicklung, gängige Plattformen und Technologien die eingesetzt werden aufgebaut werden. Des Weiteren muss abgegrenzt werden, wann von einem mobilen Endgerät gesprochen wird. Da der Autor dieser Arbeit im Abschnitt 2.2 die einzelnen Plattformen und Technologien vorgestellt hat und die mobilen Geräte im Kapitel 2.1.1 definiert hat, wurde ein gewisses Vorwissen aufgebaut, damit die Antworten der ExpertInnen interpretiert werden können.

Die Ergebnisse der Interviews sind im Anhang A detailliert aufgelistet. Wie bereits im Abschnitt 4.2.2 erwähnt, werden die Interviews aus Datenschutzgründen anonymisiert und die einzelnen ExpertInnen als Testperson eins bis Testperson fünf genannt. Im folgenden Teil dieses Abschnittes erfolgt eine Interpretation des Autors basierend auf dem hermeneutischen Zirkel statt. Die Bewertung der einzelnen Antworten wird von allen fünf Testpersonen zusammengefasst und jede Frage in derselben Reihenfolge, die im Abschnitt 4.2.2. festgelegt wird, präsentiert und mit Fachliteratur bestätigt. Das Weitern werden die einzelnen Fragen hier erneut erwähnt, damit dem Lesendem die Fragestellung erneut in Erinnerung gerufen wird. Auf die vordefinierten Antwortmöglichkeiten, wird in diesem Abschnitt nicht mehr weiter eingegangen, da diese bereits in der quantitativen Auswertung analysiert werden.

**Fragestellung 1**

*Was spricht für eine plattformunabhängige Softwareentwicklung?*

Der große Vorteil der plattformunabhängigen Softwareentwicklung ist jener, dass mit einer gemeinsamen Codebasis verschiedene Betriebssysteme angesprochen werden können.

Heutzutage ist es mittlerweile auch möglich, dass nicht nur eine gewisse Codebasis geteilt wird, sondern dass bereits ganze Applikationen zu 100% geteilten Code verwenden. Vor allem in diesem Bereich können viele positive Erfahrungen mit Xamarin gesammelt werden, da diese Technologie mit Xamarin.Forms einer Oberflächenbeschreibung entwickelt haben, die für die Plattformen Android, iOS und UWP verwendet werden kann. Somit müssen EntwicklerInnen nicht mehr die Oberflächen der einzelnen Plattformen getrennt entwickeln. Mounaim Latif, Younes Lakhrissi und El Habib Nfaoui (2017) beschreiben einen ähnlichen Vorteil dieses Ansatzes. Es bringt einen großen Vorteil mit sich, wenn eine gewisse Codebasis und gewisse Komponenten für mehrere Plattformen verwendet werden können.

**Fragestellung 2**

*Welche Einstiegsbarrieren sind für Unternehmen vorhanden?*

Für Unternehmen die im Bereich der Softwareentwicklung tätig sind, bringen Technologien wie beispielsweise Xamarin keine Barrieren mit sich, da es sich hierbei um ein Open-Source-Produkt handelt, welches von allen Unternehmen gratis verwendet werden kann. Einstiegsbarrieren die in gewissen Unternehmen auftreten können, haben nicht direkt mit Cross-Plattform-Technologien zu tun, sondern treten meist auf, da die Infrastruktur für mobile Softwareentwicklung generell nicht gegeben ist. So ist die größte Hürde jene, dass Auftraggeber in einer gewissen Zeit eine Applikation für ihre KundInnen anbieten möchte, aber nicht daran denkt, dass für die Entwicklung mobile Applikation meist die Infrastruktur angepasst beziehungsweise auf einen neuen Stand gebracht werden muss. Viele Unternehmen besitzen kaum Services die mit mobilen Apps kommunizieren könnten und somit muss zuerst in diesem Bereich gearbeitet werden. Diese Hürden haben aber nicht primär mit einer plattformunabhängigen Technologie zu tun.

**Fragestellung 3**

*Haben Sie in Ihrem Unternehmen bereits eine Business-Applikation mit einer Cross-Plattform-Technologie entwickelt?*

Bei der dritten Fragestellung, sollte herausgefunden werden, inwieweit Unternehmen bereits Business-Applikationen mit einer Cross-Plattform-Technologie entwickelt haben. Die Auswertung dieser Frage hat ergeben, dass Experten und Expertinnen die in einem kleinen Softwareteam arbeiten beziehungsweise bei denen der Fokus im Unternehmen in der Softwareentwicklung liegt mehr Erfahrungen sammeln konnten, als EntwicklerInnen die in einem Unternehmen tätig sind, wo der Fokus beispielsweise bei Herstellung neuer Fahrzeuge liegt. ExpertInnen die bereits mit plattformunabhängiger Softwareentwicklung Erfahrung sammeln konnten, sind von dem Konzept hinter diesen Technologien begeistert. So sind sich ExpertInnen einig, dass Technologien wie Xamarin beispielsweise den Vorteil mit sich bringen, sehr schnell und einfach eine gewisse Anzahl an BenutzerInnen ansprechen kann. Des Weiteren können mit diesen Technologien sehr viel Codezeilen eingespart werden, da durch die gemeinsame Codebasis viele Methoden beziehungsweise Klassen für mehrere Plattformen zur Verfügung gestellt werden können.

**Fragestellung 4**

*Welche Erfahrungen haben Sie selbst in diesem Bereich sammeln können?*

Durch die vierte Fragestellung wurde herausgefunden, inwieweit ExpertInnen Erfahrungen in diesem Bereich sammeln konnte. Mit Cross-Plattform-Entwicklung konnte bereits sehr viel Erfahrung gesammelt werden und es wurden bereits unterschiedliche Technologien wie beispielsweise Unity, Xamarin aber auch Angular eingesetzt. Dabei waren die Anforderungen, die an die Applikationen gestellt wurden, sehr unterschiedlich. So mussten beispielsweise gewisse ExpertInnen wenig bis keine gerätespezifischen Funktionalitäten verwenden, aber es wurden auch Apps entwickelt, die auf spezielle Eigenschaften vom jeweiligen mobilen Endgerät angewiesen waren.

**Fragestellung 5**

*Welche Erfahrungen konnten Sie bereits mit Xamarin sammeln?*

Es konnten bereits einige Projekte mit Xamarin umgesetzt werden und die Erfahrungen die dabei gesammelt werden konnten sind durchwegs positiv. Diese Technologie bietet vor allem im Zusammenhang mit Xamarin.Forms den großen Vorteil mit sich, dass durch wenig Aufwand eine Applikation entwickelt werden kann, die auf mehreren Plattformen installiert werden kann. Einer der größten Nachteile bei Technologien wie beispielsweise Xamarin ist jener, dass gewisse Abhängigkeiten zu anderen Entwicklerteams entstehen und dadurch bei Frameworkaktualisierungen immer wieder nacharbeiten entstehen können. Diese Abhängigkeiten werden noch dazu erhöht, wenn Erweiterungen von anderen EntwicklerInnen installiert werden, die es einem ermöglichen gewisse gerätespezifische Funktionen zu verwenden. Hier hat es im Jahr 2017 durch den Release von Dot-Net-Standard 2.0 eine große Umstellung seitens Xamarin gegeben. Dadurch haben viele Entwicklerteams ihre PlugIns ebenfalls aktualisiert und gewisse Funktionalitäten geändert, erweitert oder ausgebaut. Dadurch mussten EntwicklerInnen, die diese Erweiterungen in Verwendung haben, diese Änderungen aktualisieren. Somit kann gesagt werden, dass gewisse Funktionalitäten die die Technologie nicht standardmäßig anbietet am besten selbst entwickelt werden sollten, da das Projekt somit nicht von fremden EntwicklerInnen abhängig ist. Wenn dieser eine Aspekt, der bei jeder Technologie früher oder später Auftritt, nicht näher betrachtet wird, kann über diese Technologie durchaus positiv berichtet werden, da es Möglichkeiten schafft, die bis dato nicht denkbar waren. Vor allem durch den im Mai veröffentlichten Artikel, in dem Microsoft schreibt, dass sie in Zukunft nicht nur mobile Endgeräte ansprechen wollen, sondern auch Desktop-Clients die Windows 7 oder MAC os X installiert haben.

## 4.3 Erfahrungsbericht

In diesem Abschnitt erfolgt ein Erfahrungsbericht des Autors. Hierbei werden Erfahrungen, die der Verfasser dieser Arbeit bereits sammeln konnte, kurz zusammengefasst und mittels kurzen Codesnippets gezeigt, wie beispielsweise eine Oberfläche mit Xamarin.Forms aufgebaut ist. Neben diesen Beispielen wird auch über den Fortschritt von Xamarin berichtet und kurz aufgezeigt, welche Vorteile diese Technologie mit sich bringt.

Die Cross-Plattform-Technologie Xamarin hat sich als großes Ziel gesetzt, alle gängigen Plattformen im mobilen- und Desktop-Bereich zu unterstützen und somit EntwicklerInnen die Möglichkeit zu geben, mit wenig Entwicklungsaufwand so eine große Anzahl an BenutzerInnen zu erreichen.

Xamarin hat in den letzten zwei Jahren, seitdem das Unternehmen von Microsoft aufgekauft wurde, eine sehr schnelle Weiterentwicklung durchlebt und so gibt es heutzutage bereits sehr viele Möglichkeiten bei der Entwicklung. So haben beispielsweise die EntwicklerInnen von Visual Studio die Entwicklungsumgebung soweit angepasst, dass das Visual Studio sehr viele Hilfestellungen bei der Entwicklung von Xamarin bietet. So gibt es seit einiger Zeit eine Designansicht, die es den EntwicklerInnen ermöglicht, ohne dass die Applikation auf ein Android bzw. iOS Device installiert ist, eine kurze Vorschau zu bekommen, wie die Oberfläche aussehen wird. Damit dieser Previewer funktioniert, muss für Android eine 64-Bit-Version auf dem Client installiert werden. iOS setzt voraus, dass die Applikation auf einem MAC-Client gebildet wird. Somit müssen die Anwendungen entweder auf einem Apple-Gerät entwickelt werden, oder zumindest der Windows-Client mit einem Apple-Client verbunden sein, damit dies App in weiterer Folge auf dem Apple-Gerät gebildet werden kann.  Abbildung 4.14 zeigt die Vorschau für ein Android Gerät.

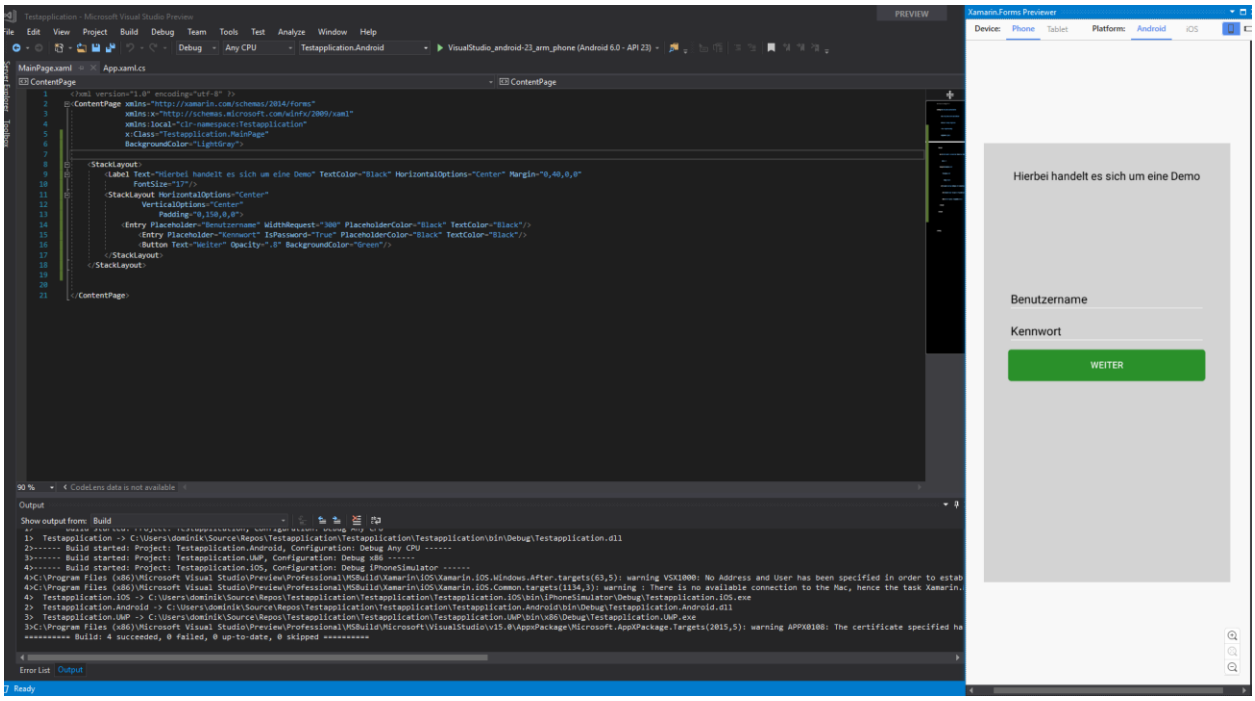



*Abbildung 4-29: Xamarin.Forms-Previewer*

Neben dieser Möglichkeit hat Microsoft bei der Build im Mai eine neue Möglichkeit vorgestellt, um eine Applikation auf dem Smartphone darstellen zu können. Im Mai wurde der Xamarin-Live-Player veröffentlicht. Dieser bietet die Möglichkeit, dass wenn der Computer, auf dem die Software entwickelt wird, im gleichen WLAN ist, wie das mobile Endgerät, die Applikation über diese WLAN-Verbindung auf das Smartphone zu pushen. Diese Art der Installation bietet den großen Vorteil, dass für iOS kein Apple-Client benötigt wird. Es muss lediglich die App aus dem App- bzw. Play-Store heruntergeladen werden und im nächsten Schritt der QR-Code fotografiert

werden. Dadurch wird das mobile Endgerät mit Visual Studio verbunden, und die Applikation, die entwickelt wird, wird direkt auf dem Smartphone installiert und es können gewisse Änderungen durchgeführt werden ohne, dass das App erneut zur Verfügung gestellt wird. So kann beispielsweise die Textfarbe von einem Label geändert werden und sobald die Änderung durchgeführt wurde, wird die neue Farbe am mobilen Endgerät angezeigt.

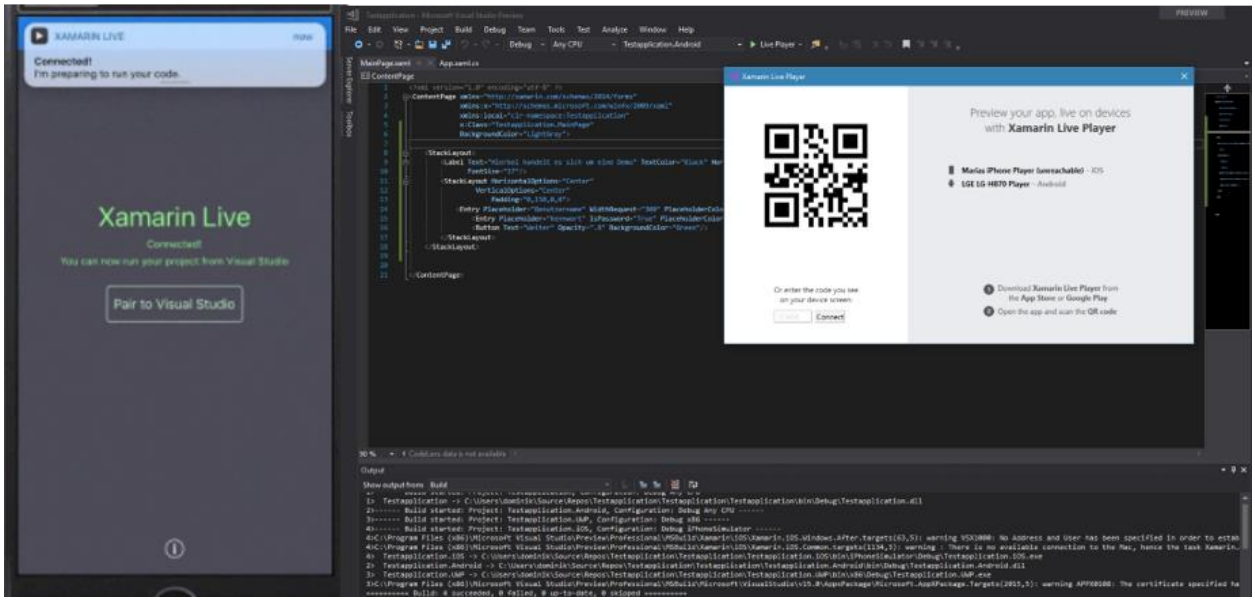



*Abbildung 4-30: Xamarin Live Player*

Neben diesen beiden Neuerungen hat Xamarin bei der Build auch eine weitere Neuerung vorgestellt. In Zukunft soll es möglich sein, dass EntwicklerInnen mit einer Xamarin.Form-Applikation nicht nur mobile Endgeräte ansprechen kann, sondern dass auch Clients die Windows 7 oder Mac OS x als Betriebssystem installiert haben, solche Applikationen starten können. Dadurch will Microsoft einen weiteren Schritt in Richtung plattformunabhängiger Softwareentwicklung machen.

Da der in Abbildung 4-30 gezeigte Previewer noch in einer Beta-Phase ist und nur im Zusammenhang mit Visuel Studio 2017 Preview zur Verfügung steht, können nicht alle EntwicklerInnen auf diese Lösung zurückgreifen. Wenn einem Programmierer oder einer Programmiererin die Anschaffung eines MAC zu teuer ist, haben diese die Möglichkeit ein Gerät über diverse Cloud-Anbieter zu beschaffen. So bietet beispielsweise MacinCloud virtuelle MAC-Server an auf denen ein MAC OS X als Betriebssystem läuft. Dafür muss einfach bei der Bestellung einer virtuellen Maschine die Funktion "remote build port" aktiviert werden und damit lässt sich dieser Client mit dem Visual Studio auf dem Entwicklungscomputer verbinden. Nach der Registrierung bekommt der Entwickler beziehungsweise die Entwicklerin den Zugang per E-Mail zugeschickt.

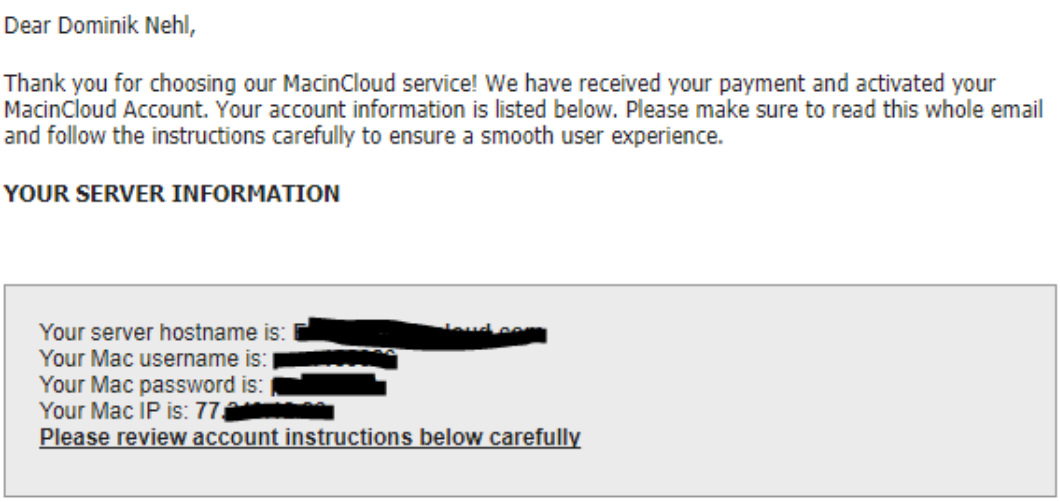



*Abbildung 4-31: Zugangsdaten MacinCloud*

Die in Abbildung 4-32 gezeigten Daten müssen im nächsten Schritt im Visual Studio eingetragen werden, damit eine aufrechte Verbindung mit dem virtuellen Client hergestellt werden kann.

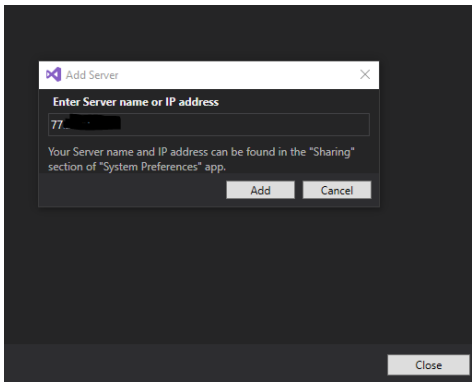



*Abbildung 4-32: Add Server in Visual Studio*

Bei erfolgreicher Verbindung wird im Visual Studio, in der Toolleiste, ein grüner Monitor angezeigt. Siehe Abbildung 4-33.


*Abbildung 4-33: erfolgreiche Verbindung*

Durch Cloudanbieter wie beispielsweise MacinCloud, können EntwicklerInnen heutzutage problemlos die Applikationen auf einem Apple-Gerät testen, ohne dafür extra Hardware zu kaufen. Diese Möglichkeit bietet eine weitere große Flexibilität.

Diese Neuerungen, die im Jahre 2017 vorgestellt wurden, haben dem Entwickler des im Abschnitt 4.1 vorgestellten Prototypen sehr geholfen. So konnte beispielsweise sehr viel Zeit beim Designen der Oberfläche gespart werden. Vor allem da der Entwickler auf einen Blogeintrag gestoßen ist, der es einem ermöglicht, trotz dem Designpatter MVVM, eine Vorschau im Xamarin-Previewer zu sehen, da es normalerweise nicht möglich ist, für eine dynamisch aufgebaute Seite, eine Vorschau zu bekommen. So schreibt Montemagno (2016) in seinem Blog über einen ViewModelLocater, der es einem ermöglicht Testdaten zur Verfügung zu stellen, die im Xamarin-

Previwerer angezeigt werden können ohne dass die Applikation gestartet wird (Montemagno, Montemagno.com, 2016).

Eine weitere großartige Möglichkeit die Xamarin.Forms bietet, ist jene, dass die Beschreibung der Oberfläche auf XAML basiert. So können sehr dynamisch Oberflächen designt werden, aber auch auf einfache Art und Weise eigene Controls erstellt werden. Dabei hat Microsoft die großen Vorteile von WPF, die ebenfalls mit XAML beschrieben wird, versucht zu übernehmen. So ist es einem Entwickler beziehungsweise einer Entwicklerin möglich, durch "Bindable Properties" einfach Controls neue Eigenschaften zu geben, die diese standardmäßig nicht mit sich bringen (Huber, 2015). Ein Beispiel dafür, wäre der unter Android sehr bekannte "Floating Action Button", oder auch kurz FAB genannt.

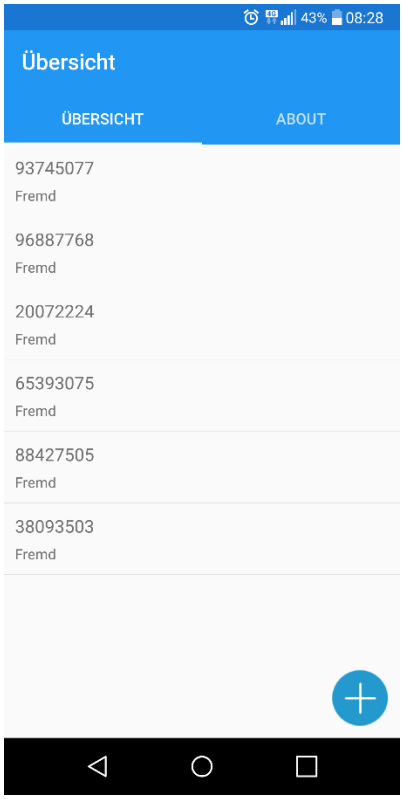



*Abbildung 4-34: FAB-Beispiel*

Damit dieser Button unter Xamarin.Forms erstellt werden kann, wird ein neues Control erzeugt, welches von dem Control Image abgeleitet wird. Damit das gewünschte Verhalten erreicht werden kann, muss die Eigenschaft "Command" hinzugefügt werden, die es ermöglicht auf eine Funktion in einer Klasse aufzurufen.

Damit ein Button wie in Abbildung 4-34 gezeigt, erstellt werden kann, muss folgendes Code-Snippet in XAML erstellt werden.

```
<customControls:ImageButton
                        Source="{markupExtensions:PlatformImage SourceImage='IconFAB'}"
                        Command="{Binding AddCommand}"
                        AbsoluteLayout.LayoutFlags="PositionProportional"
                        AbsoluteLayout.LayoutBounds="1.0,1.0,-1,-1"
                        Margin="10"/>
```

*Listing 4-1: FAB Xaml Code*

Die neuerzeugte Klasse "ImageButton" wird im Listing 4.2 kurz vorgestellt.

```
public class ImageButton : Image
    {
        public static readonly BindableProperty CommandProperty =
            BindableProperty.Create("Command", typeof(ICommand), typeof(ImageButton),
null);

        public static readonly BindableProperty CommandParameterProperty =
            BindableProperty.Create("CommandParameter", typeof(object),
typeof(ImageButton), null);

        public event EventHandler ItemTapped = (e, a) => { };

        public ImageButton()
        {
            Initialize();
        }

        public ICommand Command
        {
            get { return (ICommand)GetValue(CommandProperty); }
            set { SetValue(CommandProperty, value); }
        }

        public object CommandParameter
        {
            get { return GetValue(CommandParameterProperty); }
            set { SetValue(CommandParameterProperty, value); }
        }

        private ICommand TransitionCommand
        {
            get
            {
                return new Command(async () =>
                {
                    AnchorX = 0.48;
                    AnchorY = 0.48;
                    await this.ScaleTo(0.8, 50, Easing.Linear);
                    await Task.Delay(100);
                    await this.ScaleTo(1, 50, Easing.Linear);
                    Command?.Execute(CommandParameter);

                    ItemTapped(this, EventArgs.Empty);
                });
            }
        }

        public void Initialize()
        {
            GestureRecognizers.Add(new TapGestureRecognizer
            {
                Command = TransitionCommand
            });
        }
    }
```

*Listing 4-2: FAB Klasse*

Auch dieses Beispiel zeigt sehr gut, welche Flexibilität Xamarin mit sich bringt und wie schnell neue Controls geschaffen werden können, wenn die von Microsoft definierten Controls nicht ausreichen.

Microsoft konzentriert sich nicht nur alleine auf Xamarin, sondern versucht diese Technologie mit anderen Technologien zu verbinden und somit EntwicklerInnen weitere Möglichkeiten zu bieten. Dies spiegelt sich auch in der Verbindung zwischen Xamarin und Azure wieder und so ist es mittlerweile möglich, einen Notificationservice für Applikationen zu implementieren der über Azure läuft. Des Weiteren ist es auch möglich die in Azure eingeführte Active Directory zu verwenden und somit die Anmeldung über dieses Service zu nutzen (Xamarin.com, 2017).

Ein weiteres Feature, welches von Microsoft vorgestellt wurde, ist das Mobile Center. Mithilfe dieser Plattform, wird EntwicklerInnen die Möglichkeit geboten, die Applikation mit verschiedensten mobilen Endgeräten zu Testen. Hierfür wird ein gewisses Testszenario entwickelt und dann können im Web die verschiedensten Endgeräte gewählt werden, auf denen in weiterer Folge die Applikation getestet wird. Sobald die Applikation auf einem Gerät einen Fehler bekommt, wird dieser protokolliert und kann von den EntwicklerInnen ausgewertet werden (Tendulkar, 2017).

Da Microsoft ständig an der Weiterentwicklung von Xamarin und Xamarin.Forms arbeitet und versucht eine plattformunabhängige Softwareentwicklung zu erleichtern, wird ständig am Framework gearbeitet und versucht die neuesten Standards zu implementieren. Aus diesem Grund wurde Mitte November das von Microsoft eingeführte Dot-Net-Standard für Xamarin.Forms eingeführt (Boggan, 2017).

Diese Aktualisierung bringt viele Vorteile mit sich. So können nun Frameworks verwendet werden, die nicht mehr nur für einen gewissen Teilbereich entwickelt werden, sondern diese neue Form kann für verschiedenste Plattformen verwendet werden. Wie in Abbildung 4.35 gezeigt wird, kann dieses Framework für alle Applikationen, die mit Dot-Net entwickelt werden, verwendet werden (Landwerth, 2016).

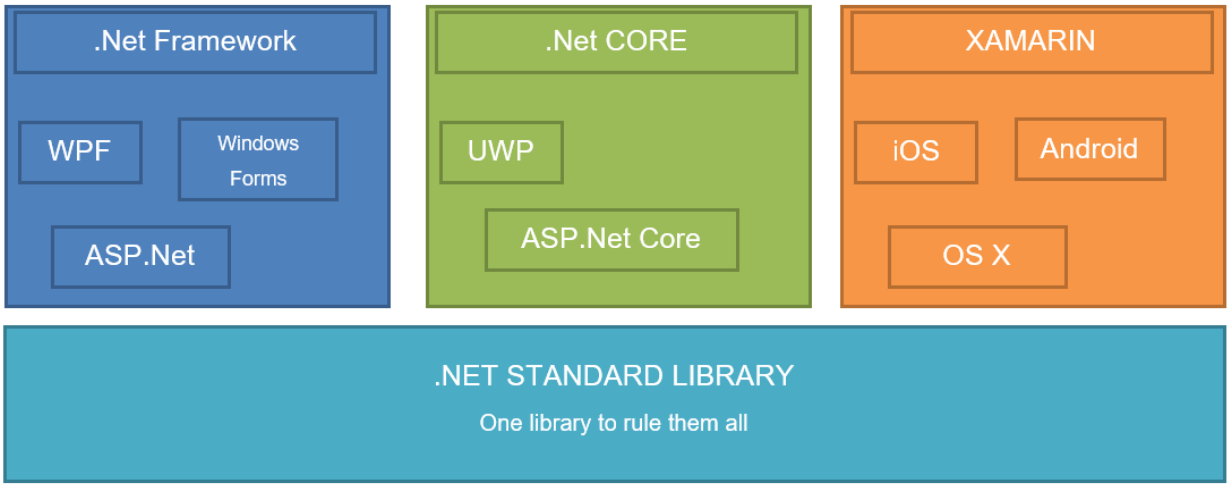



*Abbildung 4-35: Architektur der einzelnen Frameworks (Landwert, 2016)*

Diese Aktualisierung hat sehr viele Vorteile mit sich gebracht, aber es mussten dadurch auch einige Anpassungen bei dem Prototyp, der im Zuge dieser Arbeit erstellt wurde, gemacht werden. So wurde beispielsweise ein Plugin von einem Entwickler verwendet, welches ermöglicht, dass gewisse Daten lokal am mobilen Endgerät gespeichert werden können. Der Entwickler dieses Plugins hat seine Frameworks ebenfalls auf Dot-Net-Standard aktualisiert und dadurch wurde die

zuvor verwendete Typsicherheit, die mittels Genercis umgesetzt wurde entfernt (Montemagno, Montemagno.com, 2017).

Die Erfahrungen die der Autor bei der Entwicklung des Prototyps machen konnte, konnten bereits auch andere ExpertInnen sammeln. So schreibt beispielsweise Achal (2016) über die Vorteile von Xamarin. Neben den bereits erwähnten Vorteilen schreibt er ebenso über den LINQ-Support, der eine sehr einfachen Zugriff auf Datenbanken ermöglicht. Er spricht ebenfalls über das bereits kurz erwähnte Design-Pattern MVVM, welches sehr viele Vorteile im Bereich der dynamischen Oberflächenprogrammierung und der Wartbarkeit beziehungsweise der Erweiterbarkeit mit sich bringt (Achal, 2016).

Weitere EntwicklerInnen bestätigen die Erfahrungen die der Autor dieser Arbeit sammeln konnte und schreiben über ähnliche Vorteile dieser Technologie. Vor allem die Entwicklungsumgebung die Microsoft mit Visual Studio für Windows und mittlerweile auch für MAC OS bietet, bietet kaum eine andere Technologie. So müssen EntwicklerInnen, die bereits im Dot-Net-Bereich arbeiten, keine neue Entwicklungsumgebung kennenlernen und können in der Sprache, in der sie bisher gearbeitet haben weiterhin programmieren (Hermes, 2015). Diese zwei großen Vorteile bieten andere plattformunabhängigen Technologien wie beispielsweise Angular nicht. Damit in dieser Technologie entwickelt werden kann, müssen entweder gewisse Erweiterungen bei Visual Studio installiert werden, oder aber auf gewisse andere Programme wie beispielsweise Visual Studio Code zurückgegriffen werden.

Zusammenfassend kann gesagt werden, dass Xamarin im Bereich der Cross-Plattform-Entwicklung sehr viele Vorteile mit sich bringt, und in Betracht gezogen werden sollte, wenn es sich um die Programmierung einer Software handelt, die auf allen gängigen Plattformen funktionieren soll. Vor allem wenn im Unternehmen bereits ein gewisses Knowhow im Bereich Dot-Net vorhanden ist und bereits Applikationen mit C-Sharp in Kombination mit XAML umgesetzt wurden, bietet Xamarin den großen Vorteil, dass diese Technologie viele Ähnlichkeiten besitzt. Diese Meinung teilen auch sehr viele ExpterInnen, die sich mit plattformunabhängiger Softwareentwicklung beschäftigen und wurde durch die ExpterInneninterviews, die im Abschnitt 4.2.4 zusammengefasst wurden, bestätigt.

# 5 ABSCHLUSS DER ARBEIT

Im Abschlusskapitel werden zu Beginn die Ergebnisse dieser Arbeit erläutert und zusammengefasst. Außerdem wird die Forschungsfrage beantwortet werden. Im letzten Abschnitt dieses Kapitels wird auf eine mögliche Fortsetzung der Studie näher eingegangen.

## 5.1 Beantwortung der Forschungsfrage

In diesem Abschnitt des Kapitels wird erneut auf die Forschungsfrage eingegangen und kurz zusammengefasst welche Erfahrungen andere EntwicklerInnen in diesem Bereich sammeln konnten und welche Kenntnisse bei der Entwicklung des vorgestellten Prototyps erlernt werden konnten. Zu Beginn wird kurz zusammengefasst werden wie diese Arbeit aufgebaut war und welche Ergebnisse sich daraus ergeben haben.

Zu Beginn dieser Arbeit wurden dem Lesendem die theoretischen Grundlagen und Möglichkeiten, die im Bereich der mobilen Entwicklung existieren nähergebracht. Ziel der Literaturrecherche war jenes, den Begriff mobiles Endgerät einzugrenzen und in weiterer Folge festzulegen welche Plattformen für die Arbeit relevant sind. Diese Eingrenzung wurde zum einen auf Grund der Recherche auf die gängigsten Plattformen eingegrenzt und zum anderen unter der Berücksichtigung der Anforderungen an den Prototyp definiert. So wurde festgelegt, dass eine Technologie verwendet werden muss, die es ermöglicht eine Applikation auf Android, iOS und Windows-Mobile zu verwenden. Somit wurden im theoretischen Teil der Arbeit die einzelnen Plattformen und deren Geschichte näher vorgestellt und danach Technologien beschrieben, die EntwicklerInnen die Möglichkeit bieten, mittels einer Applikationen alle drei Plattformen anzusprechen. Diese Anforderung, hat die Auswahl der geeigneten Technologie sehr schnell auf wenige eingegrenzt und es musste nur mehr definiert werden, ob eine native Applikation oder eine sogenannte Hybrid-App entwickelt wird. Da einige gerätespezifischen Eigenschaften benötigt wurden, wurde gemeinsam mit den Stakeholdern beschlossen, dass eine native Applikation entwickelt werden soll, die für alle drei Plattformen zur Verfügung gestellt wird.

Im empirischen Teil dieser Arbeit wurden zu Beginn Anforderungen an einen möglichen Prototyp zusammen mit den Stakeholdern definiert und danach ein Prototyp erstellt, damit der Autor dieser Arbeit gewisse Erfahrungen in Bezug auf die plattformunabhängige Softwareentwicklung sammeln konnte. Nachdem dieser Prototyp erstellt wurde, wurde im nächsten Abschnitt zuerst erläutert wie eine mögliche Befragung aussehen könnte, und welche Arten des Interviews zur Verfügung stehen. Da es sich bei dieser Arbeit um ein sehr spezielles Thema handelt und derzeit nur eine gewisse Anzahl an möglichen TeilnehmerInnen vorhanden ist, wurde ein qualitatives Experteninterview durchgeführt. Da diese Interviews eine gewisse Struktur mit sich bringen sollten, aber den TeilnehmerInnen trotzdem gewisse Freiheiten gegeben werden sollten, wurde eine Technik eingesetzt, bei der der Interviewer beziehungsweise die Interviewerin einen Leitfaden benutzt, damit alle notwendigen Informationen vom Teilnehmenden in Erfahrung gebracht werden können.

Die Forschungsfrage die im Zuge dieser Arbeit aufgestellt lautet wie folgt:

*Inwieweit ist effiziente Entwicklung von Business Applikation mit aktuellen Cross Plattform-Technologien möglich?*

Basierend auf die Ergebnisse die einerseits durch die ExpertInneninterviews und andererseits durch den Erfahrungsbericht gesammelt werden konnten, ist davon auszugehen, dass eine effiziente Entwicklung von Business Applikationen mit aktuellen Cross-Plattform-Technologien wie beispielsweise Xamarin möglich ist. Diese Annahme lässt sich aus den Ergebnissen der Befragungen mit den ExptertInnen und des Erfahrungsberichtes, bei dem die Aussagen mittels Literatur bestätigt wurden, ableiten. Die mögliche Effizienzsteigerung konnte in dieser Arbeit nicht operationalisierbar werden, aber die Ergebnisse der Interviews, der Erfahrungsbericht und diverse Onlinequellen zeigen jedoch, dass eine Effizienzsteigerung definitiv spürbar ist, da durch Technologien wie beispielsweise Xamarin, weniger Programmieraufwand für einzelne SoftwareentwicklerInnen entsteht und der entwickelte Source-Code, eine gemeinsame Codebasis hat und nicht für verschiedene Systeme entwickelt werden muss.

Die Ergebnisse dieser Arbeit haben auch gezeigt, dass die aufgestellten Forschunghypothesen in Abschnitt 4.2 akzeptiert werden können. Die bei den Interviews gewonnenen Ergebnisse zeigen zwar das gewisse Unterschiede zwischen den einzelnen Befragten existieren, aber diese sich generell nicht signifikant unterscheiden. Diese Ergebnisse haben auch dazu geführt, dass die zuvor aufgestellten Nullhypothesen verworfen werden können, da alle Befragten und der Erfahrungsbericht bestätigen, dass zum einen die Effizienz gesteigert werden kann und dass zum anderen gewisse Kompromisse eingegangen werden müssen, wenn auf gewisse gerätespezifischen Funktionalitäten zugegriffen werden soll.

## 5.2 Ausblick

Der Prototyp der im Zuge dieser Arbeit entwickelt wurde zeigt, dass eine gewisse Effizienzsteigerung möglich ist, wenn eine plattformunabhängige Technologie eingesetzt wird. Des Weiteren wird durch die Interviews, die durchgeführt wurden, bestätigt, dass auch ExpertInnen der Meinung sind, dass eine Effizienzsteigerung möglich ist. Der Erfahrungsbericht, der vom Autor dieser Arbeit verfasst wurde, bestätigt auch diese Ergebnisse und festigt diese mit weiteren Online-Quellen von weiteren EntwicklerInnen.

Damit diese Ergebnisse weiter gefestigt werden könnten, wäre ein Feldversuch sehr hilfreich. Dabei würden mehrere SoftwareentwicklerInnen, die eine ähnliche Erfahrung aufweisen können, die selbe Anforderung gestellt bekommen und diese für zuvor definierte Plattformen entwickeln. Diese Testgruppen würden dann ein zwei Gruppen aufgeteilt werden. Die eine Gruppe muss diese Anforderungen mittels einer plattformunabhängigen Technologie entwickeln und die zweite Gruppe, muss die Applikation nativ für jede Plattform entwickeln. Die dafür gebrauchte Entwicklungszeit wird dokumentiert und in weiterer Folge ausgewertet. Wenn hierbei eine Gruppe befragt wird, die groß genug ist um signifikante Ergebnisse zu erzielen, kann allgemein festgelegt werden, ob eine Cross-Plattform-Technologie, wie beispielsweise Xamarin, tatsächlich eine Effizienssteigerung mit sich bringt. Damit diese Studie wiederrum eine gewisse Gültigkeit mit sich bringt, muss jedoch zuvor wieder definiert werden, ab welchen Zeitpunkt von einem mobilen

Endgerät gesprochen werden kann und welche Anforderungen benötigt werden, damit von einer Business-Applikation gesprochen werden kann.

# ANHANG A - Interviews

Interviewpartner 1 – arbeitet in einem kleinen Softwareteam

1. Was spricht für eine plattformunabhängige Softwareentwicklung?
    - Durch diese Technologie haben EntwicklerInnen den Vorteil, dass eine gemeinsame Codebasis geschaffen werden kann.
    - Es besteht die Möglichkeit mit einer entwickelten Applikation eine Vielzahl an AnwenderInnen zu erreichen.
    - Die Applikation hat auf jeder entwickelten Plattform die gleichen Funktionalitäten, da diese für alle Plattformen zur gleichen Zeit entwickelt werden können.

Der erste Interviewpartner ist ein großer Fan von Cross-Plattform-Entwicklung und konnte hatte bereits einige Vorträge bei Xamarin-Developer-Mettups halten dürfen. Der große Vorteil dieser Technologie ist jener, dass eine gemeinsame Codebasis geschaffen wird und meist nur mehr das Interface für die einzelnen Plattformen angepasst werden muss.

2. Welche Einstiegsbarrieren sind für Unternehmen vorhanden?
    - Die meisten Technologien wie beispielsweise Xamarin können von Unternehmen kostenlos benutzt werden somit gibt es in finanzieller Hinsicht keine Barrieren.
    - Viele Technologien haben spezielle Eigenschaften, somit wird spezielles Fachwissen benötigt
    - Unternehmen entscheiden sich meist für eine Testplattform um zu sehen ob di Applikation einen Mehrwert mit sich bringt. Danach ist für viele der Umstieg auf eine Cross-Plattform-Technologie zu aufwändig.

Der erste Experte, der befragt wurde, arbeitet in einem kleinen Softwareteam, welches bereits einige Aufträge für große Unternehmen umsetzen durfte. Die größten Probleme die bei ihm im Unternehmen aufgetreten sind, waren Probleme mit der Infrastruktur. Einige Unternehmen hatten noch keine Services, die für mobile Anwendungen, bereit waren. Diese Hürde wäre aber auch aufgetreten, wenn beispielsweise native Applikation eingesetzt werden. Sie hatten auch schon ein Projekt, bei dem nur eine Android Applikation existiert hat. Diese wurde nicht mehr weiterentwickelt und stattessen die neue App, mit einer gemeinsamen Codebasis für Android und iOS veröffentlicht. Dies wurde vom Kunden und den AnwenderInnen sehr positiv angenommen bzw. haben die Android BenutzerInnen keinen wirklichen Unterschied gemerkt.

3. Haben Sie in Ihrem Unternehmen bereits eine Business-Applikation mit einer Cross-Plattform-Technologie entwickelt?
    - Welche Funktionalität hat ihre Cross-Plattform-Applikation?
        - Mit Hilfe dieser App haben unsere Kunden die Möglichkeit, persönliche Daten abzurufen, oder ihren Berater direkt zu kontaktieren
    - Wie haben Sie die Applikation umgesetzt

    - Hybrid App
    - native Applikation (Unity, Xamarin)
- Benötigt ihre App eine hohe Performance?
    - Nein, da es sich um eine einfache Business Applikation handelt, die Daten abruft und Daten an den Server schickt

Es wurden bereits Applikationen entwickelt, bei denen sich die Kunden einloggen konnten und auf persönliche Daten Zugriff hatten. Des Weiteren wurde eine Applikation entwickelt in der die BenutzerInnen nach bestimmten Dingen suchen können und dann eine Liste von Vorschlägen angezeigt wird.

Bei Projekten, bei denen eine mobile Applikation gewünscht ist, wird von diesem Unternehmen hauptsächlich Xaramin verwendet, da diese den großen Vorteil mit sich bringt, dass die Applikation auf Android und iOS ausgeliefert werden kann. Ein weiterer Faktor ist jener, dass die Testperson bereits sehr viel Know-How im C# und XAML hat und somit keine großen Schwierigkeiten in Bezug der Programmiersprache existieren.

4. Welche Erfahrungen haben Sie selbst in diesem Bereich sammeln können?
    - Wie wurde die App umgesetzt?
        - Web-App
        - Hybrid Applikation
        - native Xamarin-Applikation
    - Wie greifen Sie auf gerätespezifische Features zu?
        - Applikation braucht keine
        - Framework bietet die Möglichkeit auf Features, die wir benötigen zuzugreifen
        - Für native Eigenschaften, verwenden wir Plugin (Eigenentwicklung/Opensource), die es ermöglichen gerätespezifische Features zu verwenden
    - Welche Probleme traten bei vergangenen Projekten auf?
        - Frameworkupdates von Herstellern bremsten das Release
        - Entwicklungsumgebung oft noch in einer Betaphase
        - Hilfestellungen durch Foren (zB stackoverflow) nur eingeschränkt vorhanden

Die Testperson hat mit seinen Kollegen bereits Hybrid-Applikationen und Xamarin Apps umgesetzt und alle Anwendungen, die entwickelt werden haben Zugriff auf gerätespezifische Funktionen. Die am häufigsten verwendete Funktion, ist der Telefonspeicher. Da die Applikationen meist offlinefähig sein müssen, müssen gewisse Daten lokal auf den Geräten speichern. Aus diesem Grund hat sich Xamarin sehr bewährt, da diese Technologie mit Hilfe dem Framework SQLite sehr einfach Daten lokal am Telefon speichern kann.

Die größte Herausforderung ist es gerade, die einzelnen Applikationen auf Dot-Net Standard 2.0 umzustellen. Microsoft im Mai bekannt gegeben, dass sie Xamarin Applikationen in Zukunft mit der Library von Dot-Net Standard betreiben möchte und somit muss das Unternehmen nun einige Applikationen bzw. Plugins bearbeiten bzw. updaten, damit diese wieder problemlos funktionieren.

5. Welche Erfahrungen konnten Sie bereits mit Xamarin sammeln?

Der Experte hat schon sehr viel Erfahrung mit Xamarin sammeln können und verfolgt diese Technologie bereits seit der ersten Stunde. Er ist von Anfang an von der Idee begeistert gewesen und hat schon einige Projekte mit Xamarin umgesetzt. Leider hatte diese Technologie zu Beginn gewisse Startschwierigkeiten und konnte sich am Markt nicht richtig durchsetzen. Erst als Microsoft Xamarin gekauft hat, war deutlich zu spüren, dass sich sehr viel in diesem Bereich tut. Dadurch wächst auch täglich die Community und es gibt immer mehr Hilfestellungen in Foren wie Stackoverflow oder im Xamarinforum.

Der Befragte ist am meisten davon begeistert, dass diese Technologie es einem ermöglicht, einen Code zu entwickeln und diesen mit ein paar wenigen Clicks auf allen Geräten zu publishen. Außerdem ist er schon sehr gespannt wie es mit Xamarin weiter geht, da nun bekannt gegeben wurde, dass in Zukunft auch Oberflächen für Windows 7 (WPF) und MacOS entwickelt werden können.

Interviewpartner 2

1. Was spricht für eine plattformunabhängige Softwareentwicklung?

    - Durch diese Technologie haben EntwicklerInnen den Vorteil, dass eine gemeinsame Codebasis geschaffen werden kann.
    - Es besteht die Möglichkeit mit einer entwickelten Applikation eine Vielzahl an AnwenderInnen zu erreichen.
    - Die Applikation hat auf jeder entwickelten Plattform die gleichen Funktionalitäten, da diese für alle Plattformen zur gleichen Zeit entwickelt werden können.

Der zweite Experte, der befragt wurde, ist vor knapp einem halben Jahr mit der der Thematik der plattformunabhängigen Softwareentwicklung in Berührung gekommen und ist seitdem von dieser Art der Entwicklung begeistert. Diese Frameworks setzten genau dort an, wo die größten Probleme der derzeitigen Softwareentwicklung entstehen

2. Welche Einstiegsbarrieren sind für Unternehmen vorhanden?

    - Die meisten Technologien wie beispielsweise Xamarin können von Unternehmen kostenlos benutzt werden somit gibt es in finanzieller Hinsicht keine Barrieren.
    - Viele Technologien haben spezielle Eigenschaften, somit wird spezielles Fachwissen benötigt
    - Unternehmen entscheiden sich meist für eine Testplattform um zu sehen ob di Applikation einen Mehrwert mit sich bringt. Danach ist für viele der Umstieg auf eine Cross-Plattform-Technologie zu aufwändig.

Bei ihm im Unternehmen ist es nur sehr schwer möglich, auf neue Technologien – wie diese – Rücksicht zu nehmen, da sie nicht mit Softwareentwicklung Geld verdienen. Sie bekommen täglich neue Anforderungen und müssen diese sehr zeitnah umsetzen. Dadurch ist es sehr schwierig nebenbei eine Infrastruktur für mobile Applikationen zu schaffen. Solange keine primäre Anforderung gestellt wird, wird sich der Befragte und sein Team in seinem Unternehmen sehr schwer tun, auf solche Technologien zurückzugreifen. Des Weiterns haben sie in der Firma nicht das notwendige Fachwissen um derzeit solche Anwendungen zu entwickeln.

3. Haben Sie in Ihrem Unternehmen bereits eine Business-Applikation mit einer Cross-Plattform-Technologie entwickelt?

In unserem Unternehmen selbst haben wir noch keine Business Applikation entwickelt.

4. Welche Erfahrungen haben Sie selbst in diesem Bereich sammeln können?

    - Wie wurde die App umgesetzt?
        - Web-App
        - Hybrid Applikation
        - native Xamarin-Applikation
    - Wie greifen Sie auf gerätespezifische Features zu?

    - Applikation braucht keine
    - Framework bietet die Möglichkeit auf Features, die wir benötigen zuzugreifen
    - Für native Eigenschaften, verwenden wir Plugin (Eigenentwicklung/Opensource), die es ermöglichen gerätespezifische Features zu verwenden
- Welche Probleme traten bei vergangenen Projekten auf?
    - Frameworkupdates von Herstellern bremsten das Release
    - Entwicklungsumgebung oft noch in einer Betaphase
    - Hilfestellungen durch Foren (zB stackoverflow) nur eingeschränkt vorhanden

Die zweite Testperson konnte privat schon einige Erfahrung im Bereich der Cross-Plattform-Entwicklung sammeln. Es wurden bereits erste Projekte mit Xamarin umgesetzt. Des Weiteren wurde auch schon ein Chatbot als Web-App entwickelt, der auf keine Ressourcen des mobilen Endgerätes zurückgreifen muss. Die größte Herausforderung war auch bei der zweiten Testperson jene, dass Microsoft beschlossen hat Xamarin in Zukunft mit Dot-Net-Standard 2.0 zu betreiben.

5. Welche Erfahrungen konnten Sie bereits mit Xamarin sammeln?

Die zweite Testperson konnte schon privat einige Erfahrung mit Xamarin sammeln und hat auch bereits erste eigene PlugIns auf Github veröffentlicht, die anderen EntwicklerInnen die Möglichkeit bietet, gerätespezifische Funktionen mit Xamarin.Forms zu nutzen. Der Befragte ist von er Vision von der Zielsetzung von Microsoft sehr begeistert und hofft, dass sie den eingeschlagenen Weg treu bleiben und weiterhin diesen Weg verfolgen werden.

Interviewpartner 3

1. Was spricht für eine plattformunabhängige Softwareentwicklung?
    - Durch diese Technologie haben EntwicklerInnen den Vorteil, dass eine gemeinsame Codebasis geschaffen werden kann.
    - Es besteht die Möglichkeit mit einer entwickelten Applikation eine Vielzahl an AnwenderInnen zu erreichen.

Die dritte Testperson ist von dem Prinzip der plattformunabhängigen Softwareentwicklung zwar begeistert, ist aber der Meinung, dass es in einigen Bereichen nicht möglich ist bestimmte Funktionalitäten zentral für alle Plattformen abzubilden.

2. Welche Einstiegsbarrieren sind für Unternehmen vorhanden?
    - Die meisten Technologien wie beispielsweise Xamarin können von Unternehmen kostenlos benutzt werden somit gibt es in finanzieller Hinsicht keine Barrieren.
    - Viele Technologien haben spezielle Eigenschaften, somit wird spezielles Fachwissen benötigt
    - Unternehmen entscheiden sich meist für eine Testplattform um zu sehen ob di Applikation einen Mehrwert mit sich bringt. Danach ist für viele der Umstieg auf eine Cross-Plattform-Technologie zu aufwändig.

Die dritte Testperson arbeitet als selbstständiger Softwareentwickler und hatte auch schon kleine Projekte, bei denen solch eine Technologie zum Einsatz gekommen ist. Bei den letzten beiden Projekten wurde Xamarin verwendet, da es sich um sehr schlanke Applikationen gehandelt hat, welche über ein Service JSON-Objekte empfangen, am Gerät aufbereiten und dem Benutzer, der Benutzerin anzeigen. Hier hat sich die Technologie sehr bewährt, da keine komplexen Ansichten gefordert waren und somit sehr schnell eine Applikation für drei Plattformen geschaffen wurde

3. Haben Sie in Ihrem Unternehmen bereits eine Business-Applikation mit einer Cross-Plattform-Technologie entwickelt?
    - Welche Funktionalität hat ihre Cross-Plattform-Applikation?
        - Mit Hilfe dieser App haben unsere Kunden die Möglichkeit, persönliche Daten abzurufen, oder ihren Berater direkt zu kontaktieren
    - Wie haben Sie die Applikation umgesetzt
        - native Applikation (Unity, Xamarin)
    - Benötigt ihre App eine hohe Performance?
        - Nein, da es sich um eine einfache Business Applikation handelt, die Daten abruft und Daten an den Server schickt

Der befragte hat bereits erste Business-Applikationen für Kunden entwickelt und da es sich dabei meist nur um sehr kleine Anwendungen handele, ist die dritte Testperson begeistert von der Flexibilität und der schnellen Entwicklung für mehrere Plattformen

4. Welche Erfahrungen haben Sie selbst in diesem Bereich sammeln können?
    - Wie wurde die App umgesetzt?
        - native Xamarin-Applikation
    - Wie greifen Sie auf gerätespezifische Features zu?
        - Framework bietet die Möglichkeit auf Features, die wir benötigen zuzugreifen
        - Für native Eigenschaften, verwenden wir Plugin (Eigenentwicklung/Opensource), die es ermöglichen gerätespezifische Features zu verwenden
    - Welche Probleme traten bei vergangenen Projekten auf?
        - Frameworkupdates von Herstellern bremsten das Release
        - Entwicklungsumgebung oft noch in einer Betaphase
        - Hilfestellungen durch Foren (zB stackoverflow) nur eingeschränkt vorhanden

Der befragte konnte somit schon einige Erfahrung in diesem Bereich sammeln, hat aber bis jetzt noch keine Web-Apps bzw. hybride Applikationen entwickelt.

Auch bei der dritten Testperson, war das letzte Update von Xamarin ein kleines Problem, da alle Plugins und Frameworks aktualisiert werden mussten, damit die neuen Funktionen eingesetzt werden können.

5. Welche Erfahrungen konnten Sie bereits mit Xamarin sammeln?

Die dritte Testperson konnte in den letzten Monaten sehr viel Erfahrung mit Xamarin sammeln, da einige kleine Projekte bereits mit dieser Technologie umgesetzt wurden. Vor allem die Infrastruktur mit Azure und dem Mobil-Center, welches von Microsoft angeboten wird, hat sehr viele Vorteile. So können zum Beispiel im Mobil-Center verschiedene Handys getestet werden ohne dass ein echtes Gerät zur Verfügung steht.

Interviewpartner 4

1. Was spricht für eine plattformunabhängige Softwareentwicklung?
    - Durch diese Technologie haben EntwicklerInnen den Vorteil, dass eine gemeinsame Codebasis geschaffen werden kann.
    - Es besteht die Möglichkeit mit einer entwickelten Applikation eine Vielzahl an AnwenderInnen zu erreichen.
    - Die Applikation hat auf jeder entwickelten Plattform die gleichen Funktionalitäten, da diese für alle Plattformen zur gleichen Zeit entwickelt werden können.

Die vierte Testperson ist ebenfalls ein selbstständiger Softwareentwickler, der meist im Bereich Webentwicklung tätig ist. Er ist prinzipiell von der Idee sehr begeistert, obwohl er der Meinung ist, dass native Applikationen immer weniger werden und in Zukunft ziemlich alle Applikationen über einen Browser funktionieren und somit keine Apps mehr von AnwenderInnen installiert werden müssen.

2. Welche Einstiegsbarrieren sind für Unternehmen vorhanden?
    - Die meisten Technologien wie beispielsweise Xamarin können von Unternehmen kostenlos benutzt werden somit gibt es in finanzieller Hinsicht keine Barrieren.
    - Viele Technologien haben spezielle Eigenschaften, somit wird spezielles Fachwissen benötigt
    - Unternehmen entscheiden sich meist für eine Testplattform um zu sehen ob di Applikation einen Mehrwert mit sich bringt. Danach ist für viele der Umstieg auf eine Cross-Plattform-Technologie zu aufwändig.

Da es sich bei der vierten Testperson um einen selbstständigen Softwareentwickler handelt, kann er nicht wirklich beurteilen wie groß Einstiegsbarrieren für Unternehmen sind. Er hat den Umstieg auf plattformunabhängige Softwareentwicklung, in seinem Fall meist Webentwicklung, ohne wirkliche Hürden gemeistert.

3. Haben Sie in Ihrem Unternehmen bereits eine Business-Applikation mit einer Cross-Plattform-Technologie entwickelt?
    - Welche Funktionalität hat ihre Cross-Plattform-Applikation?
        - Mit Hilfe dieser App haben unsere Kunden die Möglichkeit, persönliche Daten abzurufen, oder ihren Berater direkt zu kontaktieren
    - Wie haben Sie die Applikation umgesetzt
        - Hybrid App
        - native Applikation (Unity, Xamarin)
    - Benötigt ihre App eine hohe Performance?
        - Nein, da es sich um eine einfache Business Applikation handelt, die Daten abruft und Daten an den Server schickt

Die meisten Projekte, die von der vierten Testperson umgesetzt wurden, sind mobile Webanwendungen, die auch auf einem klassischen Desktop-Client funktionieren. Da sieht er auch den großen Vorteil von Webapplikationen. Er berichtet aber auch das derzeit das erste native Projekt in der Entwicklungsphase ist, und dass die Umsetzung sehr viel einfacher als gedacht ist.

4. Welche Erfahrungen haben Sie selbst in diesem Bereich sammeln können?
    - Wie wurde die App umgesetzt?
        - Web-App
        - native Xamarin-Applikation
    - Wie greifen Sie auf gerätespezifische Features zu?
        - Applikation braucht keine
        - Framework bietet die Möglichkeit auf Features, die wir benötigen zuzugreifen
        - Für native Eigenschaften, verwenden wir Plugin (Eigenentwicklung/Opensource), die es ermöglichen gerätespezifische Features zu verwenden
    - Welche Probleme traten bei vergangenen Projekten auf?
        - Frameworkupdates von Herstellern bremsten das Release
        - Entwicklungsumgebung oft noch in einer Betaphase
        - Hilfestellungen durch Foren (zB stackoverflow) nur eingeschränkt vorhanden

Die meisten Projekte die von dem Befragten ausgeführt wurden, benötigen keine bzw. nur sehr wenig Ressourcen vom mobilen Endgerät. Dadurch sind Web-Apps meist eine sehr gute Lösung, aber wie bereits bei der vorigen Frage erwähnt wurde, wird derzeit eine Applikation mit Xamarin entwickelt, die auf gerätespezifische Eigenschaften zurückgreifen muss.

5. Welche Erfahrungen konnten Sie bereits mit Xamarin sammeln?

Die Testperson hat mit Xamarin noch nicht sehr viel Erfahrung sammeln können, da er sich erst seit kurzem mit der Thematik beschäftigt. Er ist aber prinzipiell sehr begeistert von der Plattform und befürwortet die Umstellung auf Dot-Net-Standard sehr, da er dadurch Frameworks, die er für andere Projekte entwickelt hat, für diese Projekte auch problemlos wiederverwenden kann.

Interviewpartner 5

1. Was spricht für eine plattformunabhängige Softwareentwicklung?
    - Durch diese Technologie haben EntwicklerInnen den Vorteil, dass eine gemeinsame Codebasis geschaffen werden kann.
    - Es besteht die Möglichkeit mit einer entwickelten Applikation eine Vielzahl an AnwenderInnen zu erreichen.
    - Die Applikation hat auf jeder entwickelten Plattform die gleichen Funktionalitäten, da diese für alle Plattformen zur gleichen Zeit entwickelt werden können.

Die letzte Person die interviewt wurde, hat schon einige Erfahrung in diesem Bereich sammeln können und findet die Idee von einer gemeinsamen Codebasis perfekt. So können Unternehmen viel schneller auf Anforderungen vom Kunden reagieren und diese Änderungen zeitgleich veröffentlichen. Durch die gemeinsame Codebasis ist es auch gewährleistet, dass die Funktion auf allen Plattformen die gleiche ist, da alle auf die gleiche Funktionalität zurückgreifen.

2. Welche Einstiegsbarrieren sind für Unternehmen vorhanden?
    - Die meisten Technologien wie beispielsweise Xamarin können von Unternehmen kostenlos benutzt werden somit gibt es in finanzieller Hinsicht keine Barrieren.

Die fünfte Testperson arbeitet in einem kleinen Grazer Softwareunternehmen, die bereits einige Anwendungen für Unternehmen mit dieser Technologie umgesetzt haben. Die befragte ist der Meinung, dass es keine weiteren Hürden entstehen. Sie ist der Meinung, dass egal ob eine klassische native Applikation für zB Android die gleichen Probleme aufbringt wie eine Anwendung die Beispielsweise mit Unity oder Xamarin umgesetzt wird. Die größte Problematik ist dabei die passende Infrastruktur und ein Servicelayer der alle Technologien ansprechen kann.

3. Haben Sie in Ihrem Unternehmen bereits eine Business-Applikation mit einer Cross-Plattform-Technologie entwickelt?
    - Welche Funktionalität hat ihre Cross-Plattform-Applikation?
        - Mit Hilfe dieser App haben unsere Kunden die Möglichkeit, persönliche Daten abzurufen, oder ihren Berater direkt zu kontaktieren
    - Wie haben Sie die Applikation umgesetzt
        - Hybrid App
        - native Applikation (Unity, Xamarin)
    - Benötigt ihre App eine hohe Performance?
        - Ja, wir entwickeln Cross-Plattform-Spiele

Die befragte berichtet bereits von einigen Umsetzungen die sie in ihrem Unternehmen umgesetzt haben. So wurden beispielsweise Anwendungen mit Unity, Xamarin aber auch Angluar umgesetzt. Die Funktionalitäten bei den einzelnen Anwendungen waren sehr verschieden und es sind auch zwischendurch gerätespezifische Features verwendet worden.

Es wurden bereits auch kleine Spiele mithilfe von Unity entwickelt, ohne dass ein spürbarer Performanceverlust entstanden ist.

4. Welche Erfahrungen haben Sie selbst in diesem Bereich sammeln können?
    - Wie wurde die App umgesetzt?
        - Web-App
        - native Xamarin-Applikation
    - Wie greifen Sie auf gerätespezifische Features zu?
        - Applikation braucht keine
        - Framework bietet die Möglichkeit auf Features, die wir benötigen zuzugreifen
        - Für native Eigenschaften, verwenden wir Plugin (Eigenentwicklung/Opensource), die es ermöglichen gerätespezifische Features zu verwenden
    - Welche Probleme traten bei vergangenen Projekten auf?
        - Frameworkupdates von Herstellern bremsten das Release
        - Entwicklungsumgebung oft noch in einer Betaphase
        - Hilfestellungen durch Foren (zB stackoverflow) nur eingeschränkt vorhanden

Die Testperson konnte bereits sehr viel Erfahrung in diesem Bereich sammeln, da sich das Unternehmen mittlerweile auf diese Technologie spezialisiert hat und ausschließlich Cross-Plattform-Applikationen entwickelt, auch wenn es vom Kunden nicht erwünscht ist. Dies hat den Hintergrund, dass bereits eine passende Infrastruktur geschaffen wurde, und das meiste Know-How in diesem Bereich vorhanden ist.

5. Welche Erfahrungen konnten Sie bereits mit Xamarin sammeln?

Die fünfte und letzte Testperson hat schon sehr viel Erfahrung mit Xamarin sammeln können und es wurden auch schon viele Projekte damit umgesetzt. Sie ist von der Idee, die Microsoft verfolgt, und glaub das in den nächsten Jahren der Großteil an Applikationen mit Technologien wie Xamarin umgesetzt wird, da dadurch einfach sehr viele Ressourcen eingespart werden können und so sehr einfach eine große Anzahl an BenutzerInnen angesprochen werden kann. Sie hat auch schon erste Erfahrungen mit der Entwicklung für MacOS sammeln können und findet die Idee sehr gut, dass eine Anwendung entwickelt wird und dann für alle mobilen Endgeräte, Laptops und klassischen Desktop-Clients zu Verfügung gestellt werden kann.

# ABKÜRZUNGSVERZEICHNIS

| | |
|---|---|
| GPS | geografisches Informationssystem |
| UWP | Universal Windows Plattform |
| PDA | Personal Digital Assistent |
| SDK | Software Development Kit |
| API | Application Programming Interface |
| iOS | iPhone Operating System |
| HTML | Hypertext Markup Language |
| CSS | Cascading Style Sheets |
| XAML | Extensible Application Markup Language |
| SPA | Single Page Application |
| JS | JavaScript |
| OS | Operating System |
| UI | User Interface |
| MVVM | Model View ViewModel |
| WPF | Windows Presentation Foundation |
| FAB | Floating Action Button |

# ABBILDUNGSVERZEICHNIS

Abbildung 1-1: Zuwachs in Seiersberg (bevoelerkung.at, 2017) ..... 7
Abbildung 1-2: Aufbau und Gliederung der Arbeit ..... 10
Abbildung 2-1 Dimensionsmodell (Durlacher, 1999) ..... 12
Abbildung 2-2: Architektur einer nativen App (Schönberger, 2014) ..... 14
Abbildung 2-3: Architektur einer Web-App (Peters, 2017) ..... 15
Abbildung 2-4: Architektur von einer Xamarin-Applikation (Petzhold, 2017) ..... 16
Abbildung 2-5: Architektur von einer Web-App (Schönberger, 2014) ..... 16
Abbildung 3-1: Xamarin iOS, UWP und Android ..... 25
Abbildung 3-2: iOS ..... 27
Abbildung 3-3: Android ..... 27
Abbildung 3-4:Android, iOS und Windows Phone ..... 28
Abbildung 4-1: Entscheidungsprozess ..... 31
Abbildung 4-2: UWP neuer Alarm ..... 32
Abbildung 4-3: iOS neuer Alarm ..... 32
Abbildung 4-4: Android neuer Alarm ..... 32
Abbildung 4-5: UWP neuer Alarm 2 ..... 33
Abbildung 4-6: iOS neuer Alarm 2 ..... 33
Abbildung 4-7: Android neuer Alarm 2 ..... 33
Abbildung 4-8: iOS Detailansicht ..... 34
Abbildung 4-9: UWP Detailansicht ..... 34
Abbildung 4-10: Android Detailansicht ..... 34
Abbildung 4-11: iOS Detailansicht 2 ..... 34
Abbildung 4-12: Android Detailansicht 2 ..... 34
Abbildung 4-13: Android Detailansicht horizontal ..... 35
Abbildung 4-14: UWP Detailansicht horizontal ..... 35
Abbildung 4-15: iOS Detailansicht horizontal ..... 35
Abbildung 4-16: iOS Einsatz beenden ..... 36
Abbildung 4-17: UWP Einsatz beenden ..... 36
Abbildung 4-18: Android Einsatz beenden ..... 36
Abbildung 4-19: iOS Navigation ..... 37
Abbildung 4-20: UWP Navigation ..... 37
Abbildung 4-21: Android Navigation ..... 37
Abbildung 4-22: UWP Einsatzliste ..... 38
Abbildung 4-23: iOS Einsatzliste ..... 38
Abbildung 4-24: Android Einsatzliste ..... 38
Abbildung 4-25: iOS Einstellungen ..... 39
Abbildung 4-26: UWP Einstellungen ..... 39
Abbildung 4-27: Android Einstellungen ..... 39

Abbildung 4-28: hermeneutischer Zirkel (Gadamer, 2010) ..... 49
Abbildung 4-29: Xamarin.Forms-Previewer ..... 52
Abbildung 4-30: Xamarin Live Player ..... 53
Abbildung 4-31: Zugangsdaten MacinCloud ..... 54
Abbildung 4-32: Add Server in Visual Studio ..... 54
Abbildung 4-33: erfolgreiche Verbindung ..... 54
Abbildung 4-34: FAB-Beispiel ..... 55
Abbildung 4-35: Architektur der einzelnen Frameworks (Landwert, 2016) ..... 58

# TABELLENVERZEICHNIS

Tabelle 2-1: Gegenüberstellung Android, iOS und UWP ..... 14
Tabelle 2-2: Vergleich der Technologien ..... 17
Tabelle 2-3: Vergleich der Betriebssysteme (Gschweidl, 2017) ..... 21
Tabelle 4-1: definierte Anforderungen ..... 30
Tabelle 4-2: Antworten zur Frage 1 ..... 46
Tabelle 4-3: Antworten zur Frage 2 ..... 47
Tabelle 4-4: Antworten zur Frage 3 ..... 47
Tabelle 4-5: Antworten zur Frage 4 ..... 48

# LISTINGS

Listing 4-1: FAB Xaml Code ..... 56
Listing 4-2: FAB Klasse ..... 57

# LITERATURVERZEICHNIS

Achal, R. (17. Juni 2016). *quora.com*. Von https://www.quora.com/What-is-the-advantage-of-Xamarin abgerufen

Aichele, C., & Schönberger, M. (2016). *App-Entwicklung – effizient und erfolgreich.* Springer.

Asim, H. (2017). *Angular 4: From Theory To Practice: Build the web applications of tomorrow using the new Angular web framework from Google.* CodeCraft.

Baidachnyi, S. (2016). *Developing Windows 10 Applications with C#.* CreateSpace Independent .

Benisch, S., & Müller, C. (2016). *Möglichkeiten und Grenzen der plattformübergreifenden App-Entwicklung.* SpringerLink.

*bevoelerkung.at*. (01. Jänner 2017). Von http://bevoelkerung.at/gemeinde/seiersberg-pirka abgerufen

Boggan, P. (10. November 2017). *Xamarin.com*. Von Xamarin: https://blog.xamarin.com/net-standard-comes-xamarin-forms-project-templates/ abgerufen

Bogner, A., Littig, B., & Menz, W. (2013). *Das Experteninterview.* Springer.

Borycki, D. (2016). *Programming for the Internet of Things: Using Windows 10 IoT Core and Azure IoT Suite.* Developer Reference .

Budde, R., Kautz, K., Kuhlenkamp, K., & Züllighoven, H. (1992). *Prototyping.* Springer.

Buettner, K., & Simmons, A. (2011). *Mobile Web and Native Apps: How One Team Found a Happy Medium.* SprinkerLink.

Cordts, S. (2016). *Apps für Windows 10 in C#.* manaBuch.

Daubney, B. (Juni 2014). *vansonbourne.com*. Von https://www.vansonbourne.com/research-report/byod) abgerufen

Elgin, B. (17. August 2005). *web.archive.org*. Von WayBackMachine: https://web.archive.org/web/20111021061828/http://www.businessweek.com/technology/content/aug2005/tc20050817_0949_tc024.htm abgerufen

Eschenbach, A. (2016). *Plattformunabhängige Softwareentwicklung für mobile Endgeräte: Hybrid-Apps mit Cross-Platform Toolkits .* Akademiker Verlag.

Franke, F. (2012). *Apps mit HTML5 und CSS3: für iPad, iPhone und Android .* Galileo Computing.

Gadamer, H. (2010). *Hermeneutik I: Wahrheit und Methode: Grundzüge einer philosophischen Hermeneutik.* Mohr Siebeck.

Gschweidl, H. (25. April 2017). *mediaforte*. Von https://www.mediaforte.eu/blogs/news/statistiken-rund-um-handy-und-smartphone abgerufen

Hann, T. (2013). *PhoneGap 3: Apps für iOS, Android, Windows Phone & Co. Entwickeln.* Franzis.

Hartley, A. (22. Februar 2017). *Xamarin*. Von https://blog.xamarin.com/building-your-first-macos-app/ abgerufen

Haseman, C., & Grant, K. (2016). *Beginning Android Programming: Develop and Design by Chris Haseman.* Peachpit Press.

Heckmann, F. (1992). *http://www.ssoar.info*. Von ssor: http://www.ssoar.info/ssoar/bitstream/handle/document/2570/ssoar-1992-heckmann-interpretationsregeln_zur_auswertung_qualitativer_interviews.pdf?sequence=1 abgerufen

*heise.de*. (09. Oktober 2017). Von heise: https://www.heise.de/newsticker/meldung/Microsoft-Manager-Keine-neuen-Funktionen-mehr-fuer-Windows-Phones-3852862.html abgerufen

Heitkötter, H., Hanschke, S., & Majchrzak, T. A. (2012). *Evaluating Cross-Platform Development Approaches for Mobile Applications.* SpringerLink.

Hermes, D. (2015). *Mobile Development Using Xamarin.* SpringerLink.

Hommes, J. (2016). *Mobile Device Management Strategien: Wie Unternehmen dem Consumerization-Trend begegnen können.* Paperpack.

Huber, T. C. (2015). *Windows Presentation Foundation: Das umfassende Handbuch zur WPF, aktuell zu .NET 4.6 und Visual Studio 2015.* Rheinwerk Computing.

ISO/IEC IS 13250-2:2006. (2006). *Information Technology - Document Description and Processing Languages - Topic Maps - Data Model.* (International Organization for Standardization, Hrsg.) Abgerufen am 04. 05 2012 von http://www.isotopicmaps.org/sam/sam-model/

Joseph, C. (2015). *How to Submit and Distribute Apps On the Google Play Store.* Correa Media Group.

Kersten, H. (2014). *Mobile IT-Infrastrukturen: Management, Sicherheit und Compliance.* Paperback.

Knoll, M., & Meinhardt, S. (2016). *Mobile Computing.* Springer Vieweg.

Kohn, F., Kutzmutz, O., & Larisch, P. (Hrsg.). (2010). *Destillate - Literatur Labor Wolfenbüttel 2010.* Wolfenbüttel: Bundesakademie für kulturelle Bildung.

Krajci, I., & Cummings, D. (2016). *Anrdoid on x86: An Introduction to Optimizing for Intel Architecture.* apress.

Krämer, A. (2018). *Cross-Plattform-Apps mit Xamarin entwickeln: Mit C# für Android und iOS programmieren.* Hanser.

Künneth, T. (2016). *Android 7: Das Praxisbuch für Entwickler. Inkl. Einstieg in Android Studio. 70 Projekte zu allen Android-Funktionen: Multimedia, Kamera, Organizer, Sensoren, Datenbanken, Android Wear u. v. m. .* Rheinwerk Computing.

Lackes, R. (29. 11 2017). *wirtschaftslexikon*. Von http://wirtschaftslexikon.gabler.de/Definition/expertenwissen.html#definition abgerufen

Landwerth, I. (26. September 2016). *blogs.microsoft.com*. Von https://blogs.msdn.microsoft.com/dotnet/2016/09/26/introducing-net-standard/ abgerufen

Lanzer, W. (2012). *Kontextsensitive Services für mobile Endgeräte.* Springer Gabler.

Marquardt, V. (2007). *Datenerhebungstechniken im Vergleiche - Befragung, Beobachtung, Inhaltsanalyse.* GRIN Verlag.

Maske, P. (2012). *Mobile Applikationen 1.* Springer Gabler.

Mayer, S., & Wichers, T. (2016). *Obective-C 2.0: Programmierung für Mac OS x und iPhone.* mitp.

Meine AG. (2012). *Projektdokumentation.* Abteilung IT. Graz: Mein Unternehmen.

Montemagno, J. (1. May 2016). *Montemagno.com*. Von https://montemagno.com/xamarinforms-xaml-previewer-design-time-data/ abgerufen

Montemagno, J. (29. Juni 2017). *Montemagno.com*. Von https://montemagno.com/plugins-for-xamarin-go-dotnet-standard/ abgerufen

Montemagno, J. (11 2017). *SettingsPlugin*. Von Github: https://github.com/jamesmontemagno/SettingsPlugin abgerufen

Morgillo, I., & Viola, S. (2016). *Learning Embedded Android N Programming.* PacktLib.

Mounaim , L., Lakhrissi, Y., & Nfaoui, E. H. (2017). *Review of mobile cross platform and research orientations.* Wireless Technologies, Embedded and Intelligent Systems (WITS), 2017 International Conference on.

Negm-Awad, A. (2012). *Objective-C und Cocoa.* SmartBooks.

Ortinau, D. (18. May 2017). *Xamarin*. Von https://blog.xamarin.com/glimpse-future-xamarin-forms-3-0/ abgerufen

*ÖWA*. (September 2017). Von http://www.oewa.at/basic/browserstatistik abgerufen

Peters, M. (2017). *Progressive Web Apps: Der Sprung ins nächste App-Zeitalter.* Independently.

Petzhold, C. (2017). *Creating Mobile Apps with Xamarin Forms.* Microsoft.

Rauber, M. (2017). *Cross-Plattform: Entwicklung und Anwendung.* entwickler.press.

RFC 2616. (1999). *Hypertext Transfer Protocol -- HTTP/1.1.* (IETF, Hrsg.) Von http://www.ietf.org/rfc/rfc2616.txt abgerufen

Richter, J. M., & Bospoort, M. (2013). *Windows Runtime via C sharp.* Microsoft Press Corp.

Ross, M. (2013). *PhoneGap: Mobile Cross-Plattform-Entwicklung mit Apache Cordova & Co.* dpunkt.

Schnauder, V., & Jarosch, H. (2001). *Praxis der Software-Entwicklung: Techniken – Instrumente – Methoden.* Paperback.

Schreiber, D. V., & Leser, A. (Hrsg.). (2008). *Wichtiges Werk.* Graz: Wissensverlag.

Seifert, C. (2014). *Spiele entwickeln mit Unity: 3D-Games mit Unity und C# für Desktop, Web & Mobile.* Hanser.

*Spektrum.de*. (2000). Von Spektrum: http://www.spektrum.de/lexikon/neurowissenschaft/expertenwissen/3790 abgerufen

*statista*. (Mai 2017). Von Das Statistik-Portal: https://de.statista.com/statistik/daten/studie/184335/umfrage/marktanteil-der-mobilen-betriebssysteme-weltweit-seit-2009/ abgerufen

Steyer, M., & Softic, V. (2015). *AngurarJS Moderne Webanwendungen und Single Page Applications mit Javascript.* O'Reilly.

Tendulkar, M. (20. Oktober 2017). *Xamarin.com*. Von Xamarin: https://blog.xamarin.com/five-star-apps-with-visual-studio-mobile-center-test/ abgerufen

Theis, T. (2017). *Einstieg in Unity: 2D- und 3D-Spiele entwickeln. Ideal für Programmieranfänger ohne Vorwissen. Mit 15 kompletten Games aus allen Spielegenres.* Rheinwerk Computing.

Turner, J. (2011). *Developing Enterprise IOS Applications: Iphone and Ipad Apps for Companies and Organizations).* O'Reilly.

Verclas, S., & Linnhoff-Popien, C. (2011). *Smart Mobile Apps: Mit Business-Apps ins Zeitalter mobiler Geschäftsprozesse.* Springer.

Viswanathan, P. (05. May 2017). *lifewire*. Von https://www.lifewire.com/native-apps-vs-web-apps-2373133 abgerufen

Wartburg, R. v., Steinbacher, S., Wittmer, R., & Schütze, S. (2011). *Zitieren und Referenzieren nach APA.* Abgerufen am 20. 4 2012 von Wissenschaftliches Schreiben in der Psychologie: http://etools.fernuni.ch/wiss-schreiben/apa/de/html/index.html

Woiwode, G., Malcher, F., Koppenhagen, D., & Hoppe, J. (2017). *Angular: Grundlagen, fortgeschrittene Techniken und Best Practices mit TypeScript – ab Angular 4.* dpunkt.

Wurm, M. (2017). *How to publish iOS apps with no programming skills: Step by step guide to publishing mobile apps.* CreateSpace Independent Publishing Platform.

*Xamarin.com*. (Oktober 2017). Von Xamarin: https://developer.xamarin.com/guides/cross-platform/azure/ abgerufen