Master's degree thesis

INF951

Mobile cross-platform development in fragmentized environments

Daniel Andreas Ravnestad Huus

Number of pages including this page: 145

Sandane, 30.12.2015

Molde University College
Specialized University in Logistics

Mandatory statement

Each student is responsible for complying with rules and regulations that relate to examinations and to academic work in general. The purpose of the mandatory statement is to make students aware of their responsibility and the consequences of cheating. Failure to complete the statement does not excuse students from their responsibility.

*Please complete the mandatory statement by placing a mark **in each box** for statements 1-6 below.*		
1.	**I/we hereby declare that my/our paper/assignment is my/our own work, and that I/we have not used other sources or received other help than mentioned in the paper/assignment.**	☒
2.	**I/we hereby declare that this paper** 1. Has not been used in any other exam at another department/university/university college 2. Is not referring to the work of others without acknowledgement 3. Is not referring to my/our previous work without acknowledgement 4. Has acknowledged all sources of literature in the text and in the list of references 5. Is not a copy, duplicate or transcript of other work	Mark each box: 1. ☒ 2. ☒ 3. ☒ 4. ☒ 5. ☒
3.	I am/we are aware **that any breach of the above will be considered as cheating, and may result in annulment of the examination and exclusion from all universities and university colleges in Norway for up to one year, according to the Act relating to Norwegian Universities and University Colleges, section 4-7 and 4-8 and Examination regulations section 14 and 15.**	☒
4.	**I am/we are aware that all papers/assignments may be checked for plagiarism by a software assisted plagiarism check**	☒

5.	I am/we are aware that Molde University College will handle all cases of suspected cheating according to prevailing guidelines.	☒
6.	I/we are aware of the University College's rules and regulation for using sources	☒

Publication agreement

ECTS credits: 45

Supervisor: Ketil Danielsen

Agreement on electronic publication of master thesis

Author(s) have copyright to the thesis, including the exclusive right to publish the document (The Copyright Act §2).

All theses fulfilling the requirements will be registered and published in Brage HiM, with the approval of the author(s).

Theses with a confidentiality agreement will not be published.

I/we hereby give Molde University College the right to, free of charge, make the thesis available for electronic publication: ☒**yes** ☐**no**

Is there an agreement of confidentiality? ☐**yes** ☒**no**

(A supplementary confidentiality agreement must be filled in)

- If yes: **Can the thesis be online published when the period of confidentiality is expired?** ☐**yes** ☐**no**

Date: 30.12.2015

Preface

This thesis is a part of the Applied Informatics master program in Molde University College. A prerequisite for students attending the Applied Informatics master program is that the student works for a business while studying. I have been working as a software developer in Adcom Molde and the work presented in this thesis have also been a collaboration together with Adcom Molde.

With a great interest in new technology, writing this thesis about mobile application development was a great opportunity for me to learn more about new technology in my field of work and at the same time, research a topic that would greatly benefit my employer and their customers. I started working for Adcom Molde in 2009, where I have been in charge of the software engineering of a work order system used by Norwegian power plant companies. During my work in Adcom Molde I have faced many tasks and challenges on various issues, but the need for mobility was a recurring topic related to the ELA software. This thesis have been an effort to create a mobile application solution that would fit the size of Adcom Molde and at the same time fit their customer's needs.

Writing the master thesis has been a truly great endeavour and one that I could not have overcome without the continued support from my loving family and wife. I would also like to thank my colleagues in Adcom Molde and especially the development department for their support in this challenging task, I could not have asked for better colleagues and I look forward to our next projects.

I am also very grateful for the valuable feedback from my supervisor Ketil Danielsen, who have lead me through the thesis with much needed help and support. Finally, I would like to use this opportunity to honour the memory of one of the best teacher I had at Molde University College, Ola Bø. He started as my supervisor for this thesis, but sadly passed away due to sickness. He was an excellent teacher and a good man.

Daniel Andreas Ravnestad Huus

Summary

The mobile landscape have gone through major changes since the introduction of iOS and Android smartphones. This newly found industry have become a multi-billion dollar industry. With the increased possibilities and extended use of mobile devices for both personal and professional use, software companies and developers alike have to keep up with these technological advances if they expect to be able to create relevant software solutions. This thesis focused on the inability of creating mobile applications that can work on all devices with regard to stakeholders and developers in a small company. Choosing an incorrect development method can become a very costly affair and can be crippling for smaller companies. One of the causes for this inability is a result of various types of fragmentation in mobile devices which can add a great deal of complexity to mobile software development.

This thesis sought out to find a mobile development strategy that would allow Adcom Molde, a relatively small Norwegian company, to create a mobile application for their customers.

To achieve this goal the thesis first investigated various aspects of fragmentation to see in what extent they existed in the global mobile market and in the company's customer's market segment. Mobile development methods where reviewed and based on the needs and prerequisites of the company together with the fragmentation findings a development strategy was defined to eliminate potential pitfalls and costly project expenses. The development strategy was then implemented and a mobile web application was created. The finding of this thesis supports the choice of development method, based on the prerequisite of the company and its target audience. The thesis found that the alternative methods would have been more costly and time consuming or would not fit with the resources that were available in the company.

Contents

This thesis follows the action research template similar to the one described by Santos and Travassos (2011, 67).

Abbreviations

The following table describes the meaning of the abbreviations and acronyms found in this thesis.

OC	Operating context
DPI	Dots per inch
RIM	Research in motion
API	Application interface
HTML	Hypertext Markup Language
OS	Operating System
SVG	Scalable vector graphics
2D/3D	Two dimensional, three dimensional
DOM	Document object model
GPS	Global positioning system
IE	Internet explorer
MVC	Model view controller
CSS3	Cascading style sheet version 3
SQL	Structured query language
MSSQL	Microsoft SQL server
VPN	Virtual private network
PHP	Hypertext pre-processor
CMS	Content management system
JSON	JavaScript Object Notation
XML	Extensible mark-up language
REST	Representational state transfer
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
QOS	Quality of service
CRUD	Create, read, update, delete
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
CPN	Coloured petri net
WCF	Windows communication foundation
ERP	Enterprise resource planning (software)

CRM	Customer relation management (software)
BYOD	Bring your own device
CPT	Cross-platform tools
ADT	Android Developer Tools
SDK	Software development kit
IDE	Integrated development environment
EHS	Environment, Health and Safety
WOFF	Web Open Font Format
ASF	Apache Software Foundation
XHR	XMLHttpRequest
AIR	Adobe Integrated Runtime
RIA	Rich Internet Application
GB	Gigabyte
SE	Software Engineering
NIS	Network Information System
UI	User Interface
SSB	Statistics Norway
DMZ	Demilitarized zone
TLS	Transport Layer Security
CA	Certificate Authority
WSDL	Web Services Description Language
EDM	Entity Data Model
REN	Rasjonell Elektrisk Nettvirksomhet
PC	Personal Computer
NIS	Network Information System

List of figures

The following list shows the various figures found in this thesis.

List of tables

The following list shows the various tables found in this thesis.

1. Introduction

This chapter will introduce the thesis topic and explain the background and motivation behind the research. First a description of the problem is introduced followed by the context and background of the thesis. Finally the research problem is explained.

1.1 Problem description

With the advancements and power from new Smartphone's and pads, a completely new application market has arisen. There are now several million applications available for these new devices and new mobile "apps" are released every day. The mobile platforms have surpassed the personal computer (PC) market and it is here that much of the future applications will be used. However, not all applications are available for every device. One of the most common problems with developing for mobile platforms is the high level of diversity in the mobile market, also called fragmentation. With so many different nuances in mobile devices and mobile platforms, it is difficult for developers to create applications that work on all devices without adaption and customization of the application code. Developers need to know what possibilities and limitations that exist in the different technologies that are available, in order to apply correct methods when developing mobile apps. This is important for both application users and the companies that develops the application. Choosing the wrong method can become a costly and time-consuming effort.

1.2 Project context

IT Data AS[1] is a company located in Molde city in Norway. IT Data AS is a part of a part of Adcom[2], a nation-wide franchise chain in Norway. The various companies in the chain delivers hardware, software and business services to companies in Norway.

IT Data AS, also known as Adcom Molde, is the only company in the Adcom chain that have its own development department. The development department is the smallest in the company and only consists of four people. There are two developers, one project manager and one department head. The developers work full time with programming and development tasks, while the project manager and department head also have other non-development tasks. For more information about the employees in the development department please see Appendix A. The development department have been delivering their own business software

---

[1] IT Data AS (refer to: https://w2.brreg.no/enhet/sok/detalj.jsp?orgnr=966946873)

[2] Adcom (refer to: http://adcom.no/)

to small and medium power plants in Norway since 1999, when Adcom Molde released the first version of the ELA[3] (Elektronisk Arbeidsordre/Electronic work order) software. The start-up and initial development of the application was a cooperation between Adcom Molde and Stranda Energi AS[4]. The first versions used Microsoft Access[5] with Visual Basic[6]. In 2009, Adcom Molde rewrote the application to the newer Visual Basic .NET[7] platform and changed the database type to Microsoft SQL server[8] (MSSQL). With continuous development over several years, ELA now keeps track of all work orders, hours, material, basic data (customers, cost carriers, articles, net information etc.), risk assessments, calendar (planning tool) and has multiple integrations with other systems to export and import business data. Twenty different power plant companies in Norway use ELA and several hundred users in total. For more information about ELA please see Appendix B.

1.3 Research theme

Since ELA uses Microsoft technology aimed to run on the Windows operating system, there is limited possibilities to use the application on a mobile device. The only way users could register data in ELA, while out of office, was to bring a laptop connected to a cellular network. The users could then either use remote desktop to a terminal server or install ELA on the laptop and access the data through a virtual private network (VPN) connection. The drawbacks with this is that the laptop need a stable internet connection with a large capacity. Since there are multiple users in ELA and the application data is stored on a MSSQL database, there is no support for offline usage. Some ELA customers use a printed copy of the data, but with a pen and paper, users have to type any new information into ELA when they get back to the office.

Adcom Molde tried to find a mobile solution that would fit their customers' needs for several years. Multiple third party companies where contacted in order to try to find a solution that would fit ELA and meet the customers' needs. With the rising demand and popularity of smartphones and mobile apps, Adcom Molde had to find a solution or as a result they could

---

[3] ELA (Elektronisk Arbeidsordre) (refer to: http://ela.no/)

[4] Stranda Energi AS (refer to: http://www.strandaenergi.no/)

[5] Microsoft Access (refer to: https://products.office.com/nb-no/access)

[6] Visual Basic (refer to: https://msdn.microsoft.com/en-us/library/2x7h1hfk.aspx)

[7] .NET Platform (refer to: http://www.microsoft.com/net)

[8] Microsoft SQL server (refer to: www.microsoft.com/nb-no/server-cloud/products/sql-server/default.aspx)

start losing ELA customers. In the past ELA development have been customer-centred, but have become more market-driven and product-centred in recent years. The main reason for this is that it reduces the number of customizations in the software. However, the main challenges in a market-driven environment is a larger number of requirements from multiple sources (Gorschek et al. 2006). With a larger number of requirements for each task, there must also be an increase in consideration concerning its design. A positive effect of this is that developers only have one design to maintain, but in some cases it is not possible to fit all requirements in the same design. This means discarding customers' requirements or changing a customer's methods and processes to be able to use the software. This is also true for mobile application development. However, since the mobile application in this case would be an extension of the ELA application where this design was already defined. Knowing there are multiple types of mobile operating systems and both old and new mobile devices on the mobile market the challenge lay in the development and implementation of the software. Adcom Molde only had two developers and none of them had any prior experience with mobile application development which made this task a major challenge. Since ELA is shifting towards a market-driven approach, Adcom Molde needed a mobile solution that would fit the size of the company and at the same time target all of their customers and their existing mobile devices. The developers needed more knowledge about how to create mobile applications and what types of methods could potentially be a viable solution for a mobile ELA application. The next chapter contains a survey of existing literature covering various topics related to mobile application development.

1.4 Thesis outline

The arrangement of the chapters in this thesis is set up to guide the reader through the various steps that where necessary to evaluate from start to finish, in order to reach the conclusion.

- *Chapter 2: Mobile technology and fragmentation* will provide definitions for topics discussed in the thesis and mobile application types, as well as general solutions to counter fragmentation.
- *Chapter 3: Research goals* presents the research questions and hypotheses of the thesis and the reasons for why they need to be answered, together with definitions on the scope of the research questions.
- *Chapter 4: Methodology* presents the research design of the thesis. In addition a review of the data collection methods is presented together with explanations on validity and reliability of the data that has been collected.
- *Chapter 5: Analysis of fragmentation in the mobile market* provides a detailed review of the current situation in the global mobile market. The largest mobile platforms are presented and issues concerning fragmentation are highlighted for the most relevant platforms.
- *Chapter 6: Development Methods and Software Tools* provide a detailed review of the methods and software tools that are available for mobile development within each mobile application type.
- *Chapter 7: Development strategy planning* analyse the findings from chapter five and chapter six as well as the findings from a survey. The summary of this analysis determines the development method that is used in chapter 8.
- *Chapter 8: Adcom Molde mobile application development* present a detailed summary of the development method and the process that that was created by Adcom Molde.
- *Chapter 9: Evaluation and analysis* will present the result of the development method that was used in chapter 8.
- *Chapter 10: Reflections and learning* analyse the result from chapter nine and tries to compare it with other research findings to specify the outcome of the research. Finally a review of the learning and practical outcome is reviewed.

At the end of the thesis document there is attached a comprehensive Appendix. Only a summary of the collected data is available here. If raw data is needed please send an e-mail to: daniel.huus@gmail.com

2. Mobile technology and fragmentation

This chapter contains a review of existing literature on mobile technology and fragmentation. The chapter will provide the definitions and theoretical framework behind the research questions in this thesis.

2.1 *Fragmentation definition*

Fragmentation in a mobile context refer to the heterogeneity and diversity of the mobile phones available on the market. This diversity includes both hardware and software specifications on the mobile phones. Mobile fragmentation for developers is the diversity and lack of unification in a given context. Mobile fragmentation is “the inability to develop an application against a reference operating context (OC) and achieve the intended behaviour in all OCs suitable for the application.” It means that when developing mobile applications, diversities in the many OC’s will affect how an application works from one OC to another. An OC for an application is the environment that influences the way the application work and behave (Rajapakse 2008b).

2.2 *Mobile devices definition*

When referring to mobile devices the thesis refers to smartphones and tablets. A smartphone is a cell-phone with capabilities similar to that of a computer. Web browsing, e-mail and the possibility to run many different applications on top of a standardized operating system (PC Magazine 2012). A tablet is very similar to a phone and is a general-purpose computer contained in a large touchscreen panel. However, a tablet usually don’t have the capability to use SMS or make phone calls (PC Magazine 2015).

2.3 *Fragmentation types on mobile devices*

There are several types of fragmentation that affect mobile development. The various fragmentation types have been discussed in an article by Damith C. Rajapakse (Rajapakse 2008b) from the National University of Singapore, School of Computing and is the main source for fragmentation definitions in this thesis.

When developing a mobile application it is normally towards many different OC’s. As seen on Figure 2.1 the target market can contain a lot of different factors that all results in different OC’s and which, in turn, also results in an fragmented application in order to function correctly in all targeted OC’s.

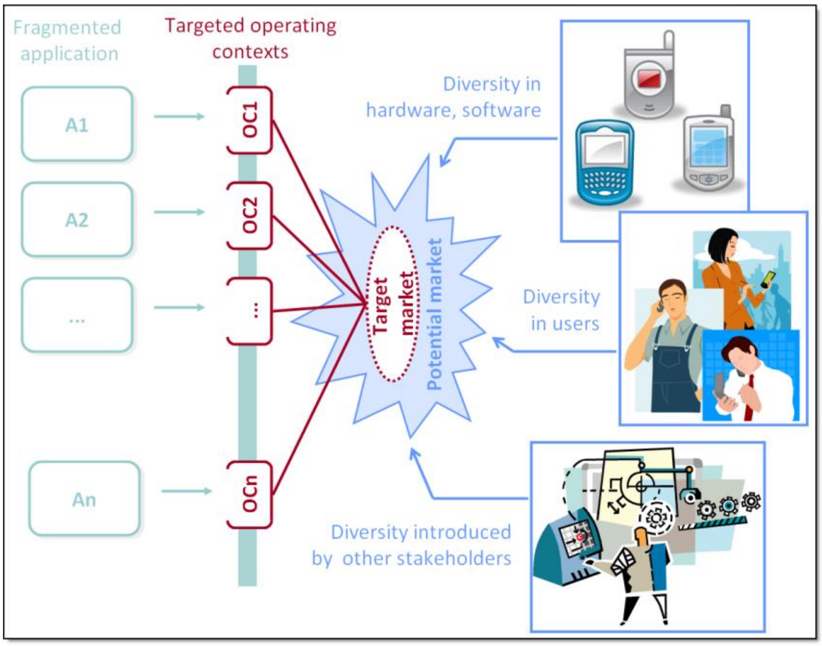


FIGURE 2.1: SHORT OVERVIEW OF MOBILE APPLICATION FRAGMENTATION (RAJAPAKSE 2008B).

New mobile phone releases are constantly changing, with every mobile vendor. This makes development on mobile devices a great challenge for developers. When developing mobile application there are multiple types of fragmentation developers have to be aware off. The next section discusses these types more closely.

2.3.1 Software fragmentation

Software fragmentation divides in to a subset of three fragmentation factors.

1. Platform fragmentation

   Mobile phones run many different types of platforms/operating systems that differs in compilation language, available API's and hardware support.

2. Implementation of standards

   Mobile vendors and operating systems may implement standards in different ways, making the variations in the way they behave.

3. Feature variations

   The software and OS may just include a subset of features depending if it is a full version or a light version.

2.3.2 Hardware fragmentation

One level of fragmentation is the physical aspect of the mobile device itself, the hardware. This includes for instance the screen. The screen consists of many different factors:

- The physical size (measured from one corner to the other corner, diagonally), usually measured in inches.
- Resolution: The total number of pixels on the screen gives us the screen resolution. A pixel is a small point of light that can change colour. Each point represents a part of the image the whole screen is displaying and together they form an image. The more pixels, the more details the screen is able to display. Referencing the screen resolution means referencing the number of pixels in the width and height of the screen, for instance 800 x 600. To get the total amount of pixels, multiply the number of pixels in the width and height. A width of 800 pixels and a height of 600 pixels will give us a total resolution of 480 000 pixels.
- Screen density, also referred to as DPI (dots per inch), is the amount of pixels within a physical area of the screen and usually measured within one inch of the screen. The larger the amount of pixels measured in one given area, the closer the pixels are to each other – the higher the image quality is.
- The aspect ratio of the screen is how wide the screen is compared to its height. Most large screens are wider than they are in height, but this may vary too, for instance the screen orientation.
- On modern mobile phones and tablets, the screen orientation can change at runtime to the way the users hold/view the screen. The orientation is the perspective the user views the screen. Either landscape (wide) or portrait (tall). For instance, a screen with an aspect ratio of 16:9 (in landscape) can, when turned to portrait, become 9:16.
- The colour depth gives us information about how many colours the screen is able to display. Colour depth is measurement in bits. The higher the bit count, the more colours can the screen reproduce.

Other hardware diversities are how the phone receives input (keyboard, touch screen, buttons etc.). Phone features such as camera, GPS (global positioning system), Bluetooth, connectivity (2G, 3G, 4G, Wi-Fi etc.) and other features such as processing power (CPU) and memory size (Rajapakse 2008b).

2.3.3 User preference fragmentation

Each user or user group may have different preferences in language, style and accessibility requirements. For instance, poor eyesight or other types of disabilities that makes it hard to use the application in a way that others would find easy.

2.3.4 Environmental fragmentation

The physical locations and places where the applications are used can be very varied. Connection signals, restrictions in network access, opened ports. Etc. all add to the level of fragmentation.

2.4 *Mobile application types*

Currently, there are three types of mobile applications:

1. Native application.
2. Web application.
3. Hybrid application.

2.4.1 Native applications

A native application is an application specifically designed to run on a device's specific operating system and machine firmware (Global Intelligence Alliance 2010) (Appcelerator 2012b).

> Native apps can come preinstalled on smartphones or tablets, but can also be downloaded from a public and internal/enterprise app stores. Native apps are developed to leverage mobile device capabilities directly from the operating system, such as the camera, geo-location, animation and more. These device-specific functionalities add to the richness of the user experience and are a prime differentiator between native and HTML5 applications (Appcelerator 2012b).

These types of applications can access all of the device software and hardware features such as calendar, phonebook, camera, accelerometer etc. The application run as a standalone and independent application on the device.

Deploying native software on several of the OS's, means that developers have to write the same application multiple times, one for each programming language that is involved. Table 1 shows an overview over the different programming languages a developer needs to know when developing multiple native applications. For a small development team it will be hard to support multiple platforms, quality assurance and development costs will be much higher for each additionally supported platform. In addition, varieties in implementations of Application programming interfaces (API) and standards can occur (for instance HTML interpretations between browsers) and according to practitioners, this will be one of the most tiresome types of fragmentation (Rajapakse 2008a).

Creating an application in a native programming language, like Java for Android, will limit the use of the application to the Android platform. Therefore, it is only usable by users that have a phone with the Android operating system. When writing native applications the choice of platform is very important. It requires a larger capacity from the development companies to develop native applications for multiple platforms at the same time. The most common thing will be to develop towards one platform at a time or just support one platform (Mominis 2011).

To support multiple platforms, developers are required to have different skill sets. Table 1 gives an overview of the different skill sets required to develop on multiple platforms. Support and maintenance will also be more labour intensive and the cost of development will become higher for each additional platform.

Mobile OS Type	Skill Set Required
Apple iOS	C, Objective C
Google Android	Java (Harmony flavored, Dalvik VM)
RIM BlackBerry	Java (J2ME flavored)
Symbian	C, C++, Python, HTML/CSS/JS
Windows Mobile	.NET
Window 7 Phone	.NET
HP Palm webOS	HTML/CSS/JS
MeeGo	C, C++, HTML/CSS/JS
Samsung bada	C++

TABLE 1: SKILL SETS REQUIRED TO DEVELOPING NATIVE APPLICATIONS ON DIFFERENT PLATFORMS (CHARLAND AND LEROUX 2011).

In addition to the need of knowing different programming languages, developers also need to handle different programming tools and environments, even hardware. Developing for the Apple platform requires a MAC (OS X) while developing for the Android and Windows Phone platform requires a PC (Windows).

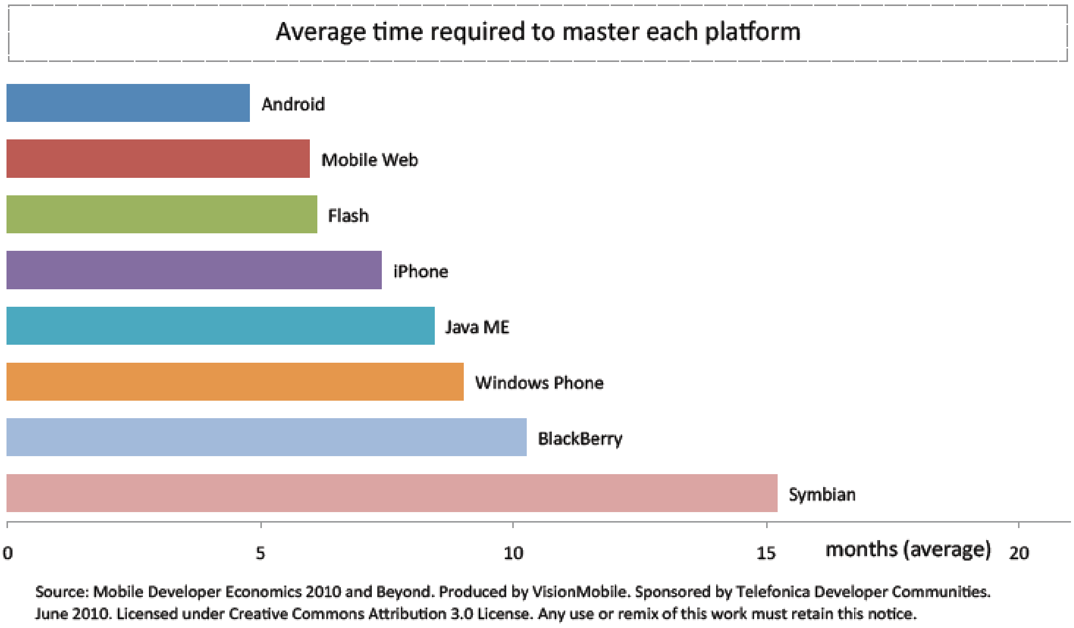


FIGURE 2.2: AVERAGE TIME IT TAKES TO LEARN TO DEVELOP FOR VARIOUS PLATFORMS (VISIONMOBILE 2010).

A study on the impact of programming language fragmentation on developer productivity by Jonathan L. Krein et al. (2010), found that developers using a single programming language is more productive than a developer that develop with multiple programming languages. Developers that write evenly with multiple languages impacts the size of code contributions. The study included 500 randomly selected developers that worked on open-source projects on SourceForge[9] and compared how many code lines they produced with the number of programming languages they used. The study makes a note on the fact that the developers in this study includes minimally active developers and that the number of contributed code lines is likely to low for professional full-time developers. Thus the study cannot make a conclusion about full-time developers. The study state that they believe knowing the causality in language fragmentation will lead to better-informed decisions in adopting new programming languages, frameworks and when assigning developers to projects. They also expect that programming language fragmentation will have a higher impact on developers that works with multiple paradigms (such as object-oriented vs. languages that aren't object oriented).

[9] SourceForge (refer to: http://sourceforge.net/)

2.4.2 Web Applications

A web application is an application made especially for the internet and runs in a web browser. Web applications typically uses HTML (Hypertext Markup Language) and CSS (Cascading Style Sheet). HTML is a very simple language and easy to understand. It is not a script or programming language, but a Markup Language. A Markup Language describes the content, like text, images and tables. CSS changes the visual layout and appearance of the HTML with images, colours, positioning of elements, fonts, backgrounds etc. (W3C 2012a).

2.4.3 Hybrid applications

A hybrid application is a combination of a native application and a web application.

> With a hybrid app, much or all of the user interface appears in a browser window, with a native app wrapped around it to provide access to device functionality not available via the browser (Lionbridge 2012, 5).

2.4.3.1 Native web wrapper

A web wrapper is a framework that inserts the mobile web application in a native "shell". This shell runs the web application as a native application that runs on the phone instead of a browser. This wrapper offers similar functionality as a browser, but the only thing the application does is run your web application. The wrapper also allows access to APIs in the framework. With the use of a wrapper, it is also possible to submit the compiled application to the various application stores.

Some of the frameworks that use the web wrapper method is PhoneGap, Intel XDK, Cocoon, Icenium and Marmalade (VisionMobile 2013c).

2.4.3.2 Web-to-native converter

The web-to-native converter is similar to the native web wrapper method. Both application types uses HTML5/CSS with JavaScript to create the code. The web-to-native approach uses JavaScript code to generate native code for each platform. The application code is JavaScript and a compiler/pre-processor maps the JavaScript code to the native programming language. This gives the application access to hardware APIs and services. These applications are be distributed through the application stores. Some of the frameworks that uses this approach is Appcelerator Titanium, Game closure and Cocos2D (VisionMobile 2013c).

2.4.3.3 Native JavaScript API

Some platforms support native access through JavaScript APIs by using HTML5/CSS with JavaScript to create the application. This approach allows native compilation and the application can be installed on the device. This allows for distribution through the application stores. Some of the platforms that support this approach is BlackBerry, Google Chrome, Firefox OS and Windows 8. (VisionMobile 2013c).

2.5 *Fragmentation solutions*

A part of the reason there are fragmentation issues on the mobile platform is that there are a limited amount of resources available. With smaller screens and fewer input methods than in a traditional computer environment, the software has to be adapted to each OC, while as on a computer developers can more easily fit the OC to the software. An example of this is the screen size. In a desktop environment, developers can optimize the application to fit the smallest screen resolutions and the application will still be very usable on a computer with a larger resolution. However, if developers adapt the mobile application to fit on a very small screen, a user with a larger screen will not get the same user experience and usability. With so many variations, it is hard to create just one application that will fit all OC's.

When developing for multiple OCs there are some approaches that developers can use in order to manage fragmentation. Figure 2.3 shows a flow chart with the complete ontology of the different approaches to handle multiple OC's. The next section will describe the details in each approach.

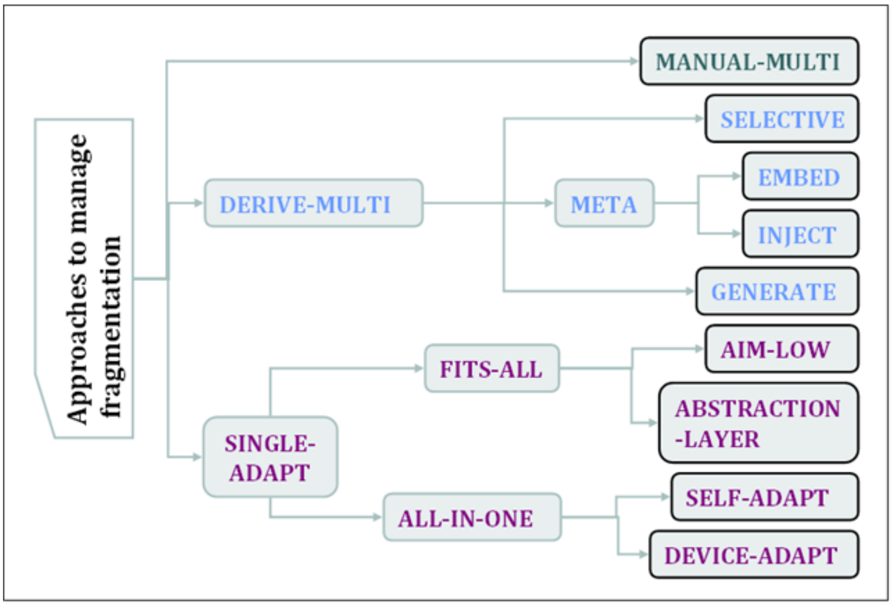


FIGURE 2.3: APPROACHES TO MANAGE FRAGMENTATION (RAJAPAKSE 2008B).

2.5.1 MANUAL-MULTI

The simplest way to develop an application with multiple OC in mind is to develop a new version for each OC (Manual-multi). This approach duplicates the code base for each OC the application will support. Each codebase contains OC specific changes to fit the OC. This approach is perhaps best suited if there is a low amount of OC's to support. By using this method, it is important to understand that each codebase duplication will multiply the workload needed to make any changes in the application.

Number of OC = Number of application versions

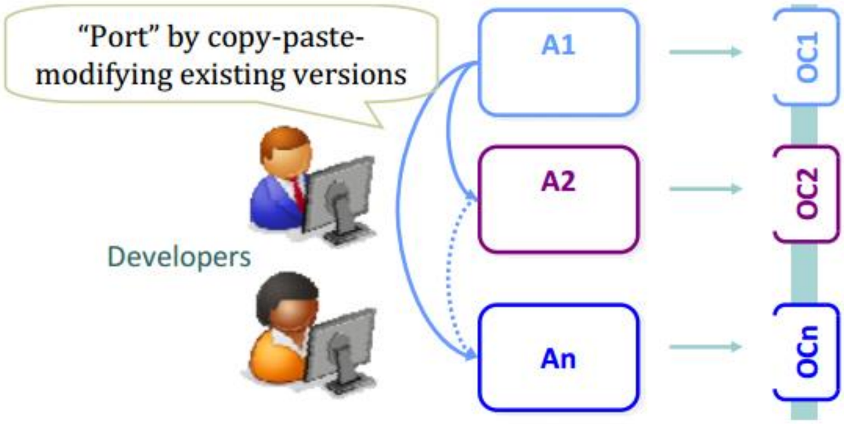


FIGURE 2.4: MANUAL-MULTI DE-FRAGMENTATION TECHNIQUE (RAJAPAKSE 2008B).

2.5.2 DERIVE-MULTI

Other approaches can also minimize the workload when developing for multiple OCs. One approach is "Derive-multi" which uses a single code base to fit all required OCs. There are three ways to achieve this.

2.5.2.1 SELECTIVE

With the selective method developer's work on a single code base, and use different interchangeable components for each OC that they want to support. A build script then compiles the application with instructions to import the required components for each OC.

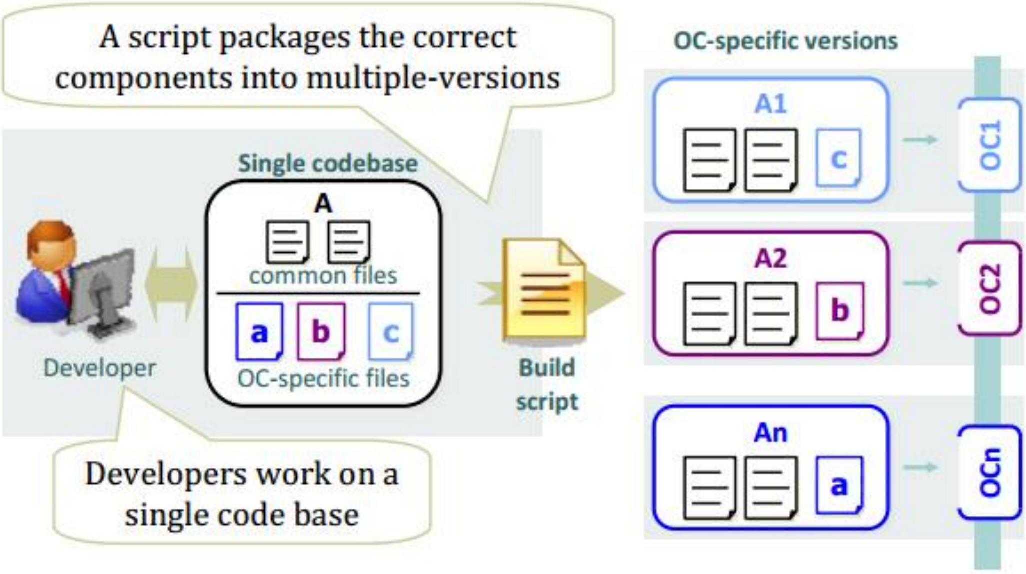


FIGURE 2.5: SELECTIVE METHOD DE-FRAGMENTATION TECHNIQUE (RAJAPAKSE 2008B).

2.5.2.2 META (Meta-programming)

Meta-programming dynamically inserts code into the compiled application. There are two ways to achieve this, embed and inject.

2.5.2.2.1 EMBED

With the embed method the developer's works on one codebase that contains all OC variations. When compiling the application, tags in the code will tell the compiler what to compile for each version of the software.

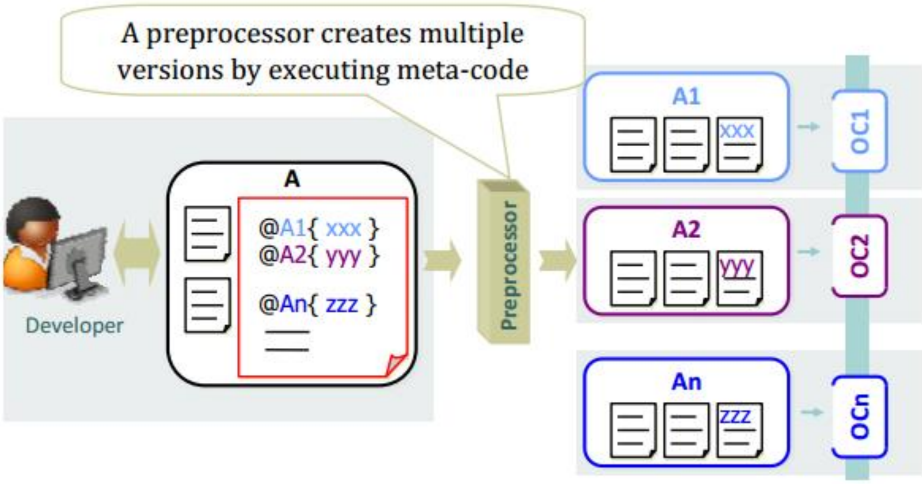


FIGURE 2.6: DE-FRAGMENTATION TECHNIQUE. EMBED METHOD (RAJAPAKSE 2008B).

2.5.2.2.2 INJECT

With the inject method, OC specific code is written separately from the application code, and a pre-processor retrieves and merges the OC code with the generic application code for each OC.

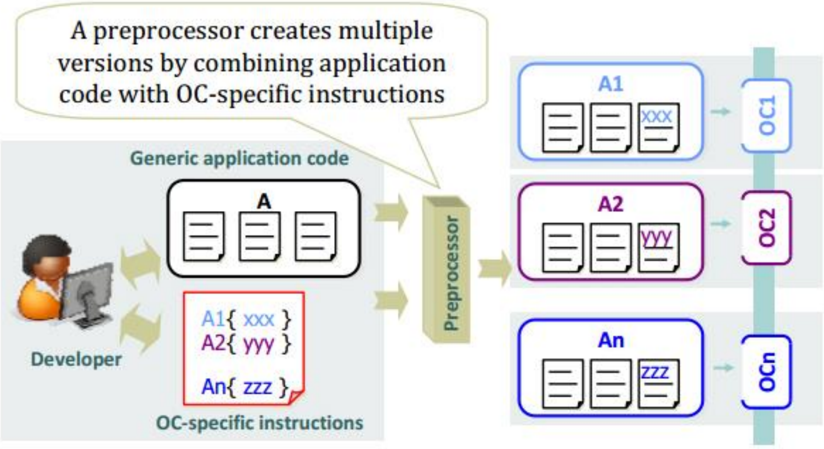


FIGURE 2.7: DE-FRAGMENTATION TECHNIQUE. INJECT METHOD (RAJAPAKSE 2008B).

2.5.2.3 Generate

The generate method uses a generator that adapts the software (written in a generic way) and creates a version for each OC. The generator has built-in knowledge on how to make the software fit the needed OC's. This requires less coding for the developer, but it the OC adaption is limited to the generators built-in knowledge of various OC's.

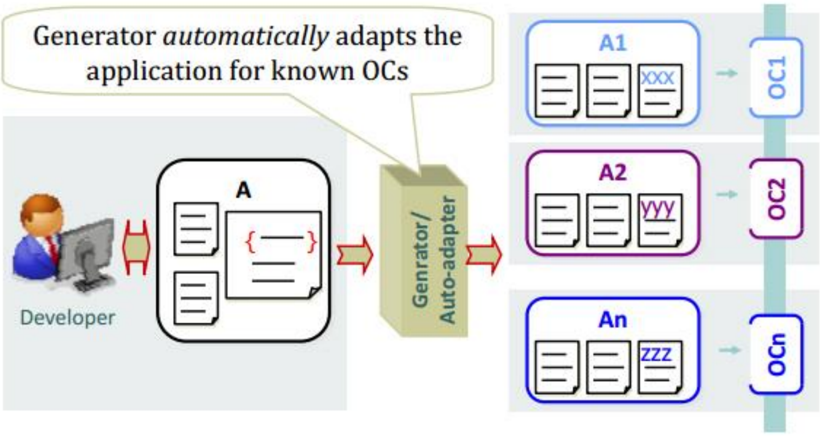


FIGURE 2.8: DE-FRAGMENTATION TECHNIQUE. GENERATE METHOD (RAJAPAKSE 2008B).

2.5.3 SINGLE-ADAPT

Another method to minimize workload is the SINGLE-ADAPT approach. The application is compiled from one codebase specifically adapted to the OCs. SINGLE-ADAPT divides into two sub categories: FITS-ALL and ALL-IN-ONE.

2.5.3.1 FITS-ALL

With the FITS-ALL method, the application fits all OCs without any adaption in the code. There are two ways to accomplish this: AIM-LOW and ABSTRACTION-LAYER.

2.5.3.1.1 AIM-LOW

The aim-low approach the application only uses methods and features that all OC's supports. If for instance if one of the file systems the developer wants to use does not support files above 1 GB (Gigabyte), then the application will be designed to contain only files smaller than 1GB. The lower the scope of the requirements the more OCs the application support.

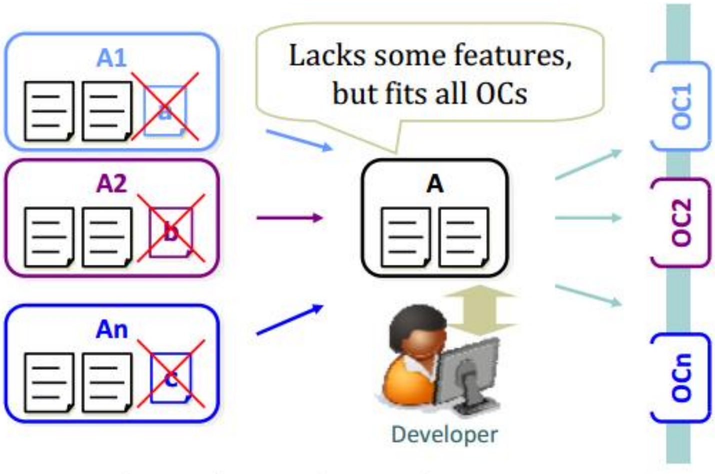


FIGURE 2.9: DE-FRAGMENTATION TECHNIQUE, AIM-LOW METHOD (RAJAPAKSE 2008B).

2.5.3.1.2 ABSTRACTION-LAYER

With the abstraction approach the application will be based on an abstraction layers (layers/interfaces that does something on a lower level) using the API's of the abstraction layers. One way to accomplish this is by using web-services or DLL files. These layers send and retrieve data or execute functions in a language that the application can interpret and that fits the various OCs.

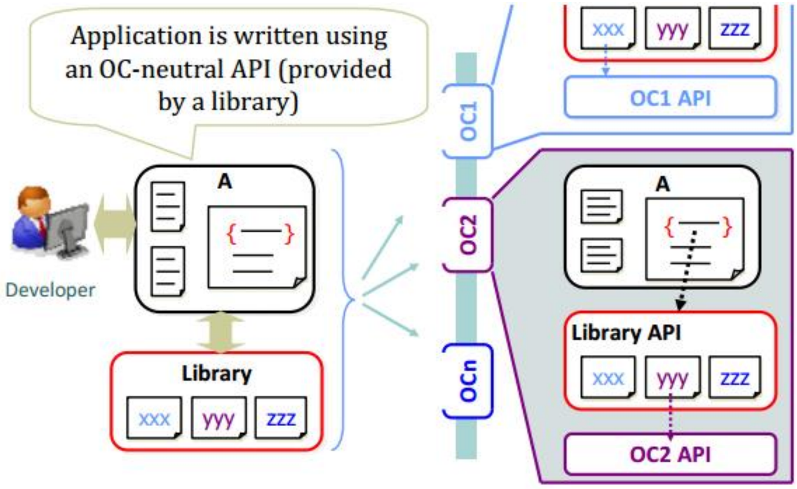


FIGURE 2.10: DE-FRAGMENTATION TECHNIQUE, ABSTRACTION-LAYER METHOD (RAJAPAKSE 2008B).

2.5.3.2 ALL-IN-ONE

With the ALL-IN-ONE approach, the application is also running from one code base, but the code in the application is adaptable to the OC. There are two ways to achieve this:

2.5.3.2.1 SELF-ADAPT

With the self-adapt method the application gathers information about the environment at run-time, and changes accordingly to fit the current OC.

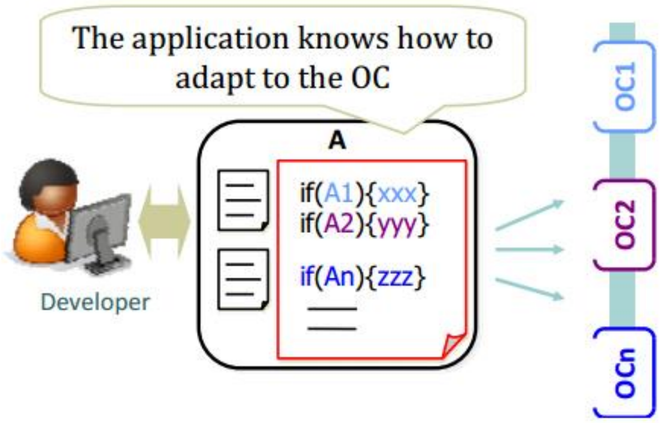


FIGURE 2.11: DE-FRAGMENTATION TECHNIQUE, SELF-ADAPT METHOD (RAJAPAKSE 2008B).

2.5.3.2.2 DEVICE-ADAPT

With the device-adapt method the application code is abstract and the device that is running the application adapts the application to the current OC. Commonly applied when dealing with fragmentation in the UI (user interface) part of an application (Rajapakse 2008b).

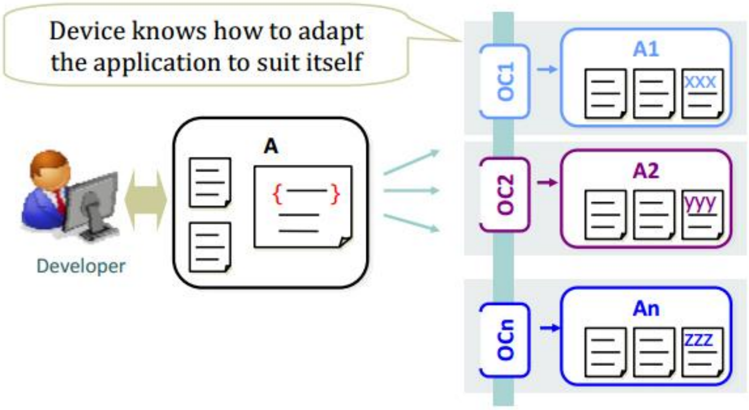


FIGURE 2.12: DE-FRAGMENTATION TECHNIQUE, DEVICE-ADAPT METHOD (RAJAPAKSE 2008B).

3. Research goals

Rajapakse (2008b), defines the various types of fragmentation that can exist on mobile platforms and what general methods that can be used to deal with fragmented OCs. However, Rajapakse does not state the types of devices or mobile platforms where these methods are valid. Since there has been many changes with mobile devices and platforms since the first release of Apples iPhone in January 2007[10] and Android in October 2008[11], there are no information in the methods Rajapakse propose that say if it is possible to use them in mobile application development for these new platforms. Furthermore, Rajapakse does not provide any details on the current state of mobile development nor what tools are available for developing applications with the methods that he describes. To be able to select a proper development strategy it is not enough to know what general approaches are available to develop mobile applications. Rajapakse states that he wish to continue with research on mobile fragmentation and making a comprehensive evaluation of tools and techniques currently used to manage fragmentation. This type of information is very valuable for all developers that want to create mobile applications, not just for developers in Adcom Molde. This leads us to the research questions of this thesis.

3.1 *Research questions*

Sjøberg, Dybå, and Jørgensen (2007), state that software engineering (SE) "is about developing, maintaining and managing high-quality software systems in a cost-effective and predictable way". Having to deal with fragmentation in SE is in direct conflict with this argument and the possible implications of fragmentation is what makes it important and interesting for developers and stakeholders. Sufficient knowledge about fragmentation is an important foundation developers and managers need before selecting development methods. This knowledge helps to define areas that must be addressed in order to be able to develop applications that have to work across multiple OC's. First after exploring the notion that fragmentation is an issue the thesis can move on to find solutions that can compensate for the negative side-effects in the given context. In case fragmentation would be minimal or non-

---

[10] Press release from Apple January 9, 2007. (refer to: https://www.apple.com/pr/library/2007/01/09Apple-Reinvents-the-Phone-with-iPhone.html)

[11] Android hacker's handbook. Page 2, section 3 (refer to: https://books.google.no/books?id=2qo6AwAAQBAJ&pg=PA2&dq=android+history+alpha+version+1.0&hl=no&sa=X&ei=Pj3vVIX0OOL7ywPlhoCgBA&ved=0CDQQ6AEwAA)

existent the basis for implementing de-fragmentation methods would be superfluous. In order to make a comprehensive and well-contemplated development strategy it is vital to have knowledge about the current OCs that exist in the mobile market and which of these OCs that the mobile application should support. In other words, developers are required to have knowledge about what fragmentation exist in the mobile market before developing a mobile application. The first research question of this thesis will try to answer this.

RQ 1. What types of fragmentation exist within the ELA customer group and the largest mobile platforms on the global mobile market?

<u>Hypotheses (suppositions):</u>

(H0) The ELA customer group does not contain any fragmentation.

(HA1) The ELA customer group contain multiple types of fragmentation.

(HA2) The fragmentation types found in the ELA customer group is similar to the fragmentation types found in the global mobile market.

As stated earlier in section 1.3 it is a known factor that there are multiple operating systems and phones being sold and used in the mobile market, it is thus a likely assumption to state that H0 is not true. In the likely event that the result of RQ1 is either HA1 or HA2, developers will have to select the OC's that the mobile application will need to support. After this, the next step will be to select the development method and software tools that are available for the selected OC's. If possible and perhaps more importantly, what method and tools will be able to target all of the selected OC's at the same time. This leads to research question two.

RQ 2. What development methods and software tools exists to minimize fragmentation issues for Adcom developers?

Studies by Rainer et al. 2005 and Kitchenham et al. 2007 (quoted in Santos and Travassos 2011, 4) suggest that professionals consider expert opinions rather than scientific evidence in most decision making. This leads to the adoption of new software technologies without scientific basis or well-grounded criteria's. In the event that an incorrect development method is selected, the implications can be serious for the developer and/or the company. Changing a development method late in a development phase will in many cases mean starting from scratch which can be time consuming and costly process. To be able to select a proper development method developers need to compare the possible OC's found in RQ1 with the available methods and tools found in RQ2. Only after reviewing information about these two factors will developers be able to choose the best development approach available. Section 3.2 to 3.4 will provide a definition for the terms in the research questions.

3.2 *ELA customer group*

The ELA customer group is companies that purchases or renew their license within a given year. The license includes updates to the newest version and features of the ELA software. Companies that have purchased ELA, but have not renewed their license are limited to use the version they used when their license expired. Optional modules such as risk management and the planning module can also be added to the standard version for an additional cost. Twenty-five companies paid for renewal of their license in 2012. It is unknown if there are any companies that use ELA without paying a yearly license.

3.3 *Largest mobile platforms*

This thesis defines the largest mobile platforms as the platforms that have the highest sales numbers. The thesis does not consider the lifespan of mobile devices in this definition, only the platforms that have the largest market share in the sales statistics presented in section 5.1.

3.4 *Mobile market*

The mobile market in this thesis is every operational smartphone in the world that can run a mobile application on one of the largest mobile platforms. There are limitations that narrow the mobile market such as language barriers and available distribution channels. However, the scope of this thesis does not consider these factors.

4. Methodology

This chapter will introduce the research methods used in this thesis and how they apply to the research questions. Kothari (2004), state that "research methodology is a way to systematically solve the research problem". Methodology is the study of how research is executed and the various methods researchers use to try and find a solution to the research questions.

4.1 *Research design*

Sjøberg, Dybå, and Jørgensen (2007) argue that SE research is conducted in order to develop new or modify existing technologies such as process models, methods, tools or languages that can support SE activities and the evaluation of using the new technology in the interaction with individuals, teams, projects, organizations and various tasks and software systems. Since SE is a real-world phenomenon, the SE research must also use real-world studies such as empirical methods by systematically gathering information based on observation and experiment, rather than deductive logic or mathematics. Runeson and Höst (2008, 1), state that a purely analytical paradigm in SE research is not sufficient for investigating complex real life issues and that acceptance of empirical studies in SE is continuously growing. Sjøberg, Dybå, and Jørgensen (2007), also state that SE research needs more studies that are empirical and to achieve this goal they propose to improve the connection between academia and industry. This thesis is a part of the Applied Informatics Master degree program at Molde University College where one of the purposes of the program is that the candidate is in an employment situation at the same time as writing the thesis. With this in mind, Molde University College encourages that the thesis is related to the candidate's job situation and thus supporting a stronger link between academia and industry.

The empirical method applied in this thesis is Action Research together with a mixed method approach using different quantitative and qualitative research techniques. Baskerville and Wood-Harper (1998), state that many researchers state their results a case-based, even though their research is action-based. Runeson and Höst (2008, 4), argue that a case study is purely observational while action research is focused on and involved in the change process. Action Research was a suitable empirical method because it "focuses particularly on combining theory and practice" (Greenwood, D.J. and Levin, M. quoted in Sjøberg, Dybå, and Jørgensen 2007, 4) and "provide practical value to the client organization while simultaneously contributing to the acquisition of new theoretical knowledge" (Avison, D., Lau, F., Myers, M. and Nielsen, P.A. quoted in Sjøberg, Dybå, and Jørgensen 2007, 4).

> The practical knowledge embedded in the action (represented by the identification of causal factors that can be manipulated to get the desired consequences within a set of circumstances) is the hypothesis being tested. If the intended consequences occur, then the hypothesis is confirmed. Otherwise it is rejected (or the alternative hypotheses based on the supposed environmental conditions can be accepted) (Argyris et. al. 1985 quoted in Santos and Travassos 2011, 11-12).

Sjøberg, Dybå, and Jørgensen (2007, 10), also states that Action Research is the most realistic research setting since "the setting of the study is the same as the scenario the results will be applied in for a given organization, apart from the presence of the researcher(s)". Easterbrook et al. (2008), state that in some cases the researcher and problem owner is the same person. In this thesis, the problem owner is the researcher's employer IT Data AS (Adcom Molde), thus making it a good choice given the background in which the thesis is written.

As seen in Figure 4.1, software development is a process that involves multiple steps and factors. Planning software engineering activities combines engineering knowledge with organizational culture to select a set of resources, procedures and artefacts. If a new situation emerges the engineering knowledge might be insufficient to select an appropriate software design for the activity. This is where action research comes in. The first step in action research involves planning the solution and can consist of searching for a solution provided by the scientific community or developing it within the organization. Then, simultaneously as the selected activity is preformed its execution is evaluated to create organizational learning and build new theories (Santos and Travassos 2011).

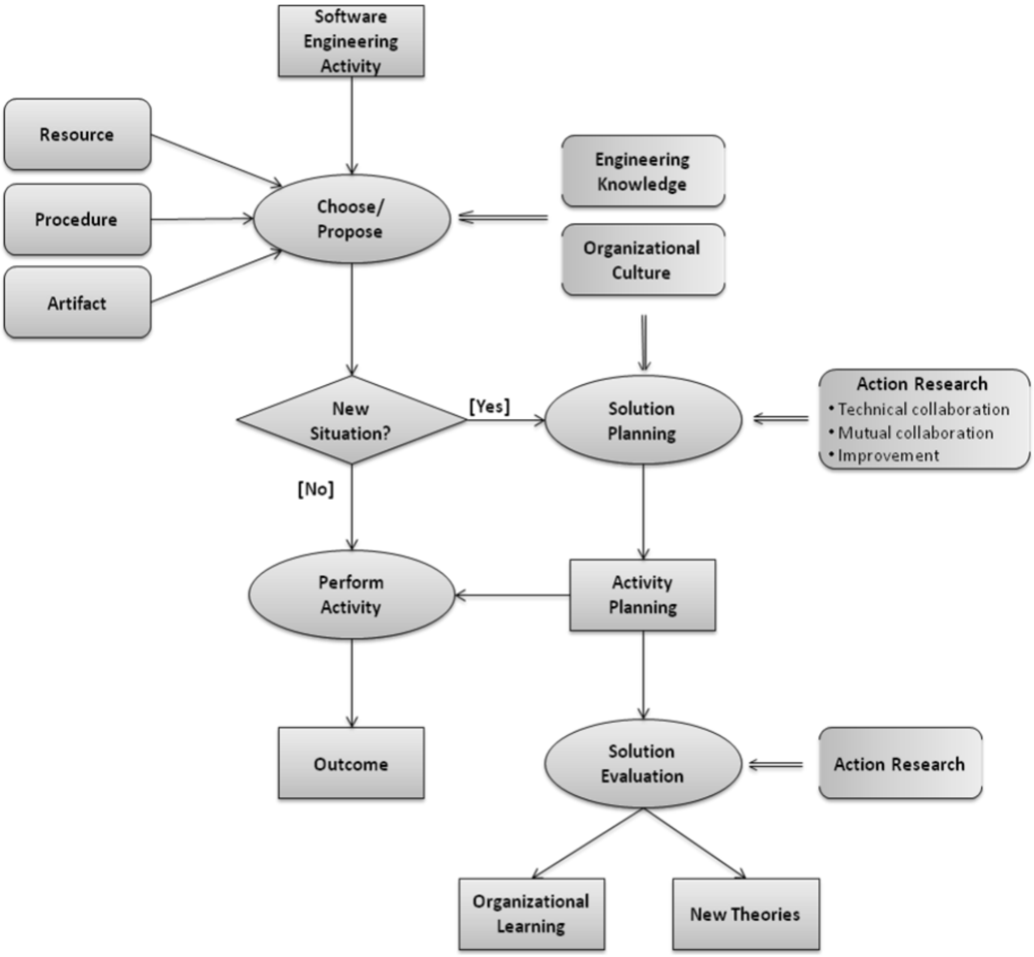


FIGURE 4.1: SOFTWARE DEVELOPMENT PROCESS WITH ACTION RESEARCH (SANTOS AND TRAVASSOS 2011)

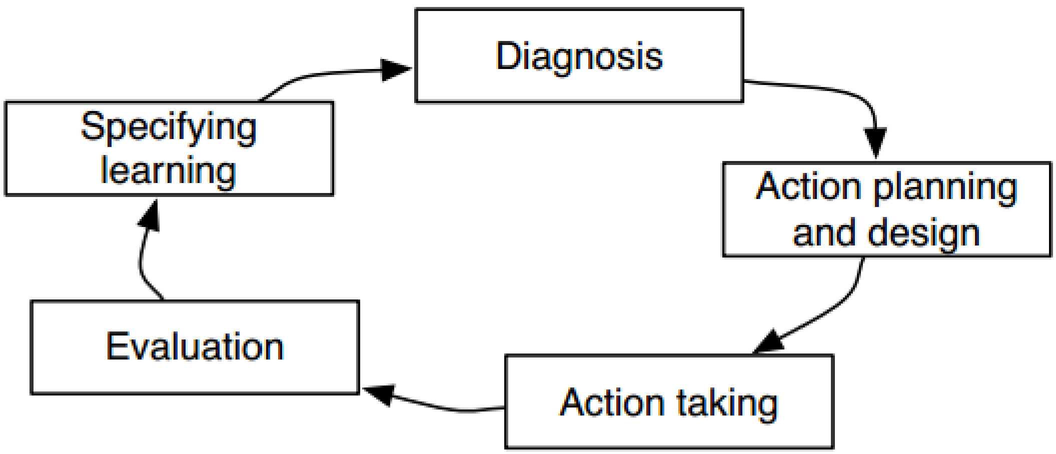


FIGURE 4.2: ACTION RESEARCH CYCLE (KAI PETERSEN ET AL. 2014)

Susman and Evered (1978), state that Action Research can consists of five phases. These phases are diagnosis, action planning, action taking, evaluating and specifying learning. It is however possible that research projects can have a different amount of phases. For instance if the researchers are conducting a “diagnostic” study. Figure 4.2 illustrates the normal Action Research cycle and how the method can be repeated if needed. This thesis consist of five phases and the execution of each phase is explained in the section below. The definition for each phase is explained by Kai Petersen et al. (2014).

The diagnosis in phase one focuses on describing and understanding the problem, it also includes defining the research questions. Gorschek et al. (2006)[12], state that it is critical to observe the real world before formulating a research question and that a close connection between the researchers and practitioners is required to be successful. Furthermore, Kai Petersen et al. (2014) state that the division of labour should be removed in Action Research and the researcher should work with the practitioner as a team. In this case, the researcher was already one of the practitioners. The researcher and the other practitioners had been trying to find a way to create a mobile application for a diverse environment (with very limited resources) for several years. Section 1.1 to 1.3 describes the background for the problem and give an explanation to how the problem arose in the organization. Chapter 2 provides a definition of the problem as well as generic solutions, while chapter 3 defines the research questions.

Part two “Action planning and design” consists of an elaborate investigation and a study of the “state of the art” to see what solutions are available. The different alternatives are reviewed and a suitable method to solve the problem is selected A review of the global fragmentation state is presented and reviewed in chapter 5 “Analysis of fragmentation in the mobile market”, while various development solutions are reviewed and discussed in Chapter 6, “Development Methods and Software Tools”. Finally, chapter 7 “Development strategy planning” will analyse the fragmentation state in the ELA customer group and how the fragmentation in this group compares to the global fragmentation state analysed in chapter 5. After this analysis a suitable development method from chapter 6 will be selected based on the needs and requirements of Adcom Molde and their ELA customers.

Part three “Action taking” explains the implementation of the action plan that was defined in part two. With Action Research the implementation of the solution proposed by the researcher is conducted in a real environment and can directly affect the organization, the object/solution that is being observed can also change during the research. To be able to conform to the organizations business needs and at the same time generate scientific

---

[12] Gorschek et al. (2006) review a technology transfer study (in a collaboration between academia and industry), however Sjøberg, Dybå, and Jørgensen (2007) state that this type of research is to be considered as Action Research.

knowledge, the researcher needs to have a deep knowledge of the organizations processes and organizational structure. The researcher also need knowledge about collecting data and be capable to interpret and understand the field that is under observation (Santos and Travassos 2011). Chapter 8, “Adcom Molde mobile application development” provides a detailed summary of how the implementation was executed in the Adcom Molde organization and explain the technical features and challenges.

Part four is “Evaluation” and is a review of the implementation from part three. Here different methods can be applied to evaluate how well the implementation was executed. These methods can consist of measurements, observation, questionnaires, interviews or focus groups. The evaluation of the Action Research can be found in chapter 9, “Evaluation and analysis” and uses interview, measurement and observation as evaluation methods.

Part five “Specifying learning”, is the final part of the Action Research cycle and consist of concluding what the learning outcome has been based on the evaluation from part four. The conclusion is used to decide how to proceed and to review if several cycles of Action Research is needed to solve the problem defined in the “Diagnosis” from part one. More information about part five can be found in chapter 10, “Reflections and learning”.

4.2 *Data collection*

The data sources in this thesis consist of both primary and secondary data.

> In software engineering, the blend of technical and human behavioral aspects lends itself to combining qualitative and quantitative methods, in order to take advantage of the strengths of both (Seaman 1999, 557).

Sekaran (2003, 219) define primary data as “information obtained first-hand by the researcher on the variables of interest for the specific purpose of the study” and data collected for research from the “actual site of occurrence of events” (Sekaran 2003, 59). On the opposite side secondary data is data gathered by other researchers and already exist. Secondary data can come from journals, books, case studies, archival records government publications, statistical and census data etc. Secondary data can either be published or unpublished (Sekaran 2003). The primary data in this thesis consist of quantitative data from a survey, hour lists and web traffic data in addition to qualitative data in the form of customer feedback, an interview and observations through Action Research.

Interviews are conducted in order to gather historical data, opinions and impressions. Interviews can also be used to elaborate about observations made by the researchers or collect information about observations and events that the researcher did not observe themselves (Seaman 1999). Since the researcher have not been a part of every step in the implementation and development, an interview with the co-developer in addition to department meetings (usually by videoconference) have been conducted to document and clarify aspects of the process in order to gain greater insight in the implementation process.

Surveys can be a good tool in order to find answers to research questions by data collection and subsequent analysis (Sekaran 2003). As a part of this thesis a survey was conducted in cooperation with IT Data AS and their ELA customers. The survey was created to get more information about the market where the research would be implemented. In addition to confirming or possibly disconfirm the initial finding in the literature review, it also creates a foundation for evaluating what OCs the implementation had to consider. More information about the survey and the specific results can be found in section 7.3.

The secondary data comes from a wide range of different sources such as books, journal articles, web articles, surveys, statistics, websites, reports and white-papers.

4.3 *Validity and reliability*

To prove that research have scientific value and to be accepted as a contribution to existing knowledge, researchers have to convince the readers that the conclusion is valid. The criteria's that define validity depends on the researcher's philosophical stance (Easterbrook et al. 2008).

Seaman (1999, 569), state that "software engineers are apt to attribute more significance to a single statistically significant finding in support of a hypothesis than is appropriate". This is because empirical findings are scarce in SE research and that the best researchers can hope for is to build a convincing body to support the proposition that the researcher is trying to confirm. The best way to do this is with a combination of qualitative and quantitative techniques and ensure that the techniques used are valid. An important tool to ensure validity is "triangulation". By using a number of different types of techniques (both qualitative and quantitative) and sources (interviews, observation, documents etc.) the support of the proposition increases (Seaman 1999). Triangulation can be used even within a single method (Easterbrook et al. 2008). This thesis builds its validity partially on using triangulation since the collected data consist of both various sources and methods. Criticism towards the thesis must be made concerning the fact that data can have been overlooked, misinterpreted or considered as more significant than what is actually applicable.

Software requirements and the tasks software is required to perform are usually very unique. The action implemented in Action Research is specific to a social setting that makes transferability and external validity difficult to obtain. Hence, it is important to recognise that the implementation of the research is done for IT Data AS and the results of the implementation might not be true for an implementation in another organization or project. Transferability and inner validity can be high within the same social setting if the context is similar (Kai Petersen et al. 2014). Repeatability in Action Research is usually not relevant since the problems are context sensitive. Knowledge outcome for the participants is the most important in Action Research and the practical outcome is at least as important as the knowledge gained. However, there are little consensus about how to balance the practical outcome versus the gained knowledge. Lau (1999, quoted in Easterbrook et al. 2008, 19) define some criteria's that the research should consider; the problem should be authentic and include authentic participants, with an appropriate level of access to the organization and with a planned exit point. The intended change should also be appropriate and adequate and most importantly "there should be clear knowledge outcomes for the participant". As mentioned in chapter 4.1 the researcher works in the company where the research is implemented and thus have good access to the organization.

> If properly planned and executed, the Action Research methodology with its dual objective of improving organizational problems and generating scientific knowledge leads to a 'win-win' scenario for both professionals (organization) and researchers (Santos and Travassos 2011, 51).

The research topic was very relevant to the participants and the planned exit point was to implement a viable solution on the problem in the organization and at the same time introduce new knowledge on the topic for all participants. To the experience of this research paper, both researchers and participants need to focus on both tasks at the same time. This part of the thesis is somewhat critique worthy since in this case the implementation/development part received too much attention and other participants have been poorly informed about some parts of the research and findings. This is something that the researcher needs to inform the other participants about, since it is the researcher's job to manage the implementation and research process. There should be more time spent on discussions around findings during the execution of the project, this could help other participants gain greater insight and feel more involved in the process. More reflective discussions with practitioners about the research could also have led to more scientific knowledge and findings. Informing other participant on the result from the “academic point-of-view” is also something that is pointed out by Santos and Travassos (2011, 58).

Since the researcher that is conducting the Action Research is a part of the organization the researcher may be biased and might not provide the objective and external view that is needed. In addition, since the researcher is an inventor of the action applied in the research the researcher might be subjective to selective bias and interpret the results positively. To ensure validity and transferability it is important to be aware of the context where the action is implemented and provide explanations to why something works in one context, but not in another. Researcher bias can be reduced by involving multiple researchers, steering groups etc. Another validity threat is changes in learning and context over time, a method of countering this can be to involve different people over time and be aware of major changes (Kai Petersen et al. 2014). In this research, with the exception of context awareness, none of these steps have been implemented to ensure validity. The reason for this was the lack of available resources and other researchers in the master thesis program.

4.3.1 Interviews

The interview in Appendix J was conducted in a semi-structured way. A series of questions was prepared before the interview was conducted to create a template for what information needed to be gathered, while the execution of the interview is structured more like a conversation using the questions to drive the conversation forward. If needed new questions was added as new information was learned throughout the interview, after the interview the report was looked over together with the interviewee to clarify issues and improve the quality of the response (Shull, Singer, and Sjøberg 2008). Bias can be introduced by the interviewer, interviewee and situation. The interviewer can misinterpret a response and unintentionally encourage/discourage a response through gestures and facial expressions. Bias can also be introduced if respondents only provide information they think the interviewer wants or expect to hear. If respondents don’t understand a question they may be hesitant to seek clarification and provide an answer that isn’t fully informed. Some interviewees might not feel comfortable being interviewed at the workplace and might not disclose their true opinion on the subject. Interview questions and the way that the questions (such as tone, emphasis on words, tone and voice inflections) are asked also need to be considered. If a question is leading or is “loaded” the answers might become biased. Furthermore, bias can be introduced if an interview is only recorded from memory and not audio/video. Memory is imprecise and often likely to be incorrect. At least interviewer should take notes during the interview or as soon as the interview is finished. Interviewees can introduce bias knowing that what they answer is being recorded on audio/video as they might not be comfortable knowing they are not fully anonymous (Sekaran 2003). The interview conducted in this research was done by videoconference between colleagues.

Since the researcher have been working remotely from a home office since late December 2012, videoconferencing is with some colleagues done on a daily basis. During the entire research period any questions that the researcher have had related to the project was quickly able to be clarified by using either meetings, e-mail, videoconference or chat. The social work-place environment where the research have been implemented is friendly and laid back. This is especially true for colleagues working within development department. It is a small group of people where confidentiality is respected and transparency of opinions are encouraged.

4.3.2 **Survey**

Sekaran (2003), state that in order to create a good survey it is important to get answers from the correct population (the people, events and things of interest the researcher wish to investigate). This means that the researcher query the people or objects that have the correct answers to solve the questions that the researcher is asking. The population frame of this survey is people working in a company that uses the ELA software. Since the survey only retrieve an answer from a subset (a "sample") and not everyone in the population the researcher have to draw conclusions from the data collected from the sample that is generalizable to the whole population. A link with information about the survey was sent to ELA customer contacts by e-mail and via the ELA application built-in news feed. Thus, it was the easiest and most accessible population that received the survey. Criticism of the sample selection in this survey must be made towards the fact that people who answered the survey have done it voluntarily, also known as a "convenience sampling" or "self-selecting sampling". This method of collecting data is often used during the exploratory phase of a research project. It is perhaps the best way of getting some basic information since it is a quick, convenient and cost-effective method. A big drawback with the method is that it is not generalizable to the rest of the population. Here there is also a potential for some bias in the results since some respondents might have a personal interest in answering the survey. Respondents comes from all parts of the organization, and possibly people that will not use all parts of the finished mobile application. For instance is management and engineers the two largest employee groups that responded to the survey. These two user groups are often involved in decision making in companies, thus maybe having other views and opinions than mobile application users from the technical/installer group. In retrospect, a quota sampling could have been implemented to get more replies from other layers in the organizations. In addition, a more random selection of respondents should have been selected to minimize bias in the response.

The survey does not ask exactly how many employees there are in each company and thus it is not possible to know exactly how large the population is. However, using information found on PROFF[13] about how many employees each company has it is possible to get an estimate of the population. PROFF[13] state that there are 531 employees in the companies that responded to the survey. With 39 responses (from 17 different companies) this gives us a confidence interval of 15.12 at 50% in the 95 percentile. A note on this is that the confidence interval for the population is only valid (as mentioned before the survey method is not generalizable to the rest of the population) for the companies that actually responded to the survey and that the information about the number of employees are from 2014. More information about the survey and its findings can be found in chapter 7.3.

4.3.3 Goal, question, metric (GQM)

Santos and Travassos (2011), argue that there are at least two fundamental aspects that gives a study rigour. The first part is the control level applied to the research which consist of minimizing researcher bias and the influence variables have over the outcome. The second part is keeping the theoretical knowledge explicit during research actions. To ensure this the Goal Question Metric (GQM) method can be applied. The GQM method was created by Basili, Caldiera, and Rombach (1994, 2) and is "based on the assumption that for an organization to measure itself in a purposeful way it must first specify the goals for itself and its projects, then it must define those goals operationally, and finally provide a framework for interpreting the data with respect to the stated goals". A bottom-up approach for measurements will not work for software because there are too many observable characteristics (complexity, time, defects, productivity etc.). A top-down approach based on goals and models is needed to define the context where the measurement (metric) should be interpreted. It is also important to make clear what informational needs the organization has

[13] PROFF "The business finder" is the official distributor of public company information found through Brønnøysundregistrene (refer to: http://innsikt.proff.no/ofte-stilte-sporsmal/)

so that it can be quantified when possible. In the end the quantified information can be analysed see if the goals have been achieved.

At the conceptual level (goal) a goal is defined for an object from a point of view in a given context. Objects of measurements can be products, processes and resources. After defining the goal the next step is at the operational level (question). In the operational level a set of research questions is defined to characterize the object of the study, in regard to the selected quality and to determine its quality from the selected point of view. Finally a set of metrics is selected to determine what data should be collected to answer the research questions defined in the operational level. Santos and Travassos (2011), state that "metric" should be called "operational questions" when GQM is used in Action Research since both quantitative and qualitative data can be used. The same term will be used in this thesis.

The GQM is a hierarchical structure, and the GQM for this thesis is presented in Figure 4.3.

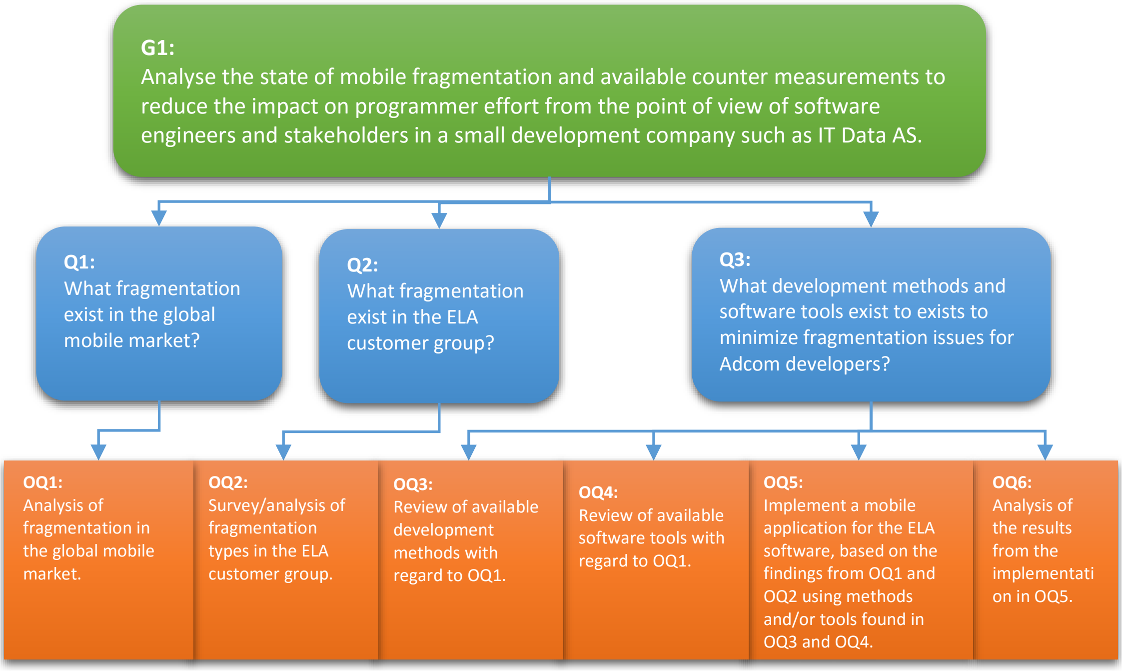


FIGURE 4.3: GQM MODEL FOR THIS THESIS.

As seen in Figure 4.3 the goal of this thesis is to find methods that can be used to reduce the impact fragmentation has on a company like IT Data AS. The first research questions have been broken down into two parts and is presented in Q1 and Q2 while RQ 2 is presented in Q3. After inserting the research questions in to the GQM we can define operational questions that will try and answer each questions and ultimately try to answer the research questions and achieve the thesis goal.

5. Analysis of fragmentation in the mobile market

To create a development strategy and choose which OCs to support, developers need to know what OCs there are in the mobile market. This leads to RQ 1 and what fragmentation exist within the largest mobile platforms and the ELA customer group. This chapter will define the operational question OQ1 from the GQM and try to answer Q1 by analysing fragmentation in the global mobile market. Only software and hardware fragmentation will be discussed. In addition an analysis of developer platform adoption, monetization and trends is included to prepare for possible changes in fragmentation that might come in the future. Q2 from the GQM about fragmentation in the ELA market will be tried answered as a part of the discussion on selecting a development method in section 7.2.

5.1 *Software fragmentation*

One of the biggest challenges in the mobile market is the fragmentation of mobile platforms. This section will analyse operating system first rather than vendor because it is the operating system that needs to be supported when developing an application. Choosing the correct platforms to develop for is key in order to reach a larger amount of users.

Table C.1 lists a summary of smartphone market shares based on sales of operating system from 2008 to Q2 2013. The same data is shown as a graph in Figure 5.1. These figures shows that there are many different mobile operating systems competing in the mobile market. It is important to keep in mind that these figures do not represent the total mobile phone shares that end users are using, but instead represents the state of the mobile market and future target market. A Supreme Court ruling in Norway from 2007 states that a mobile phone should last three to four years. The industry then later decided that the norm would be four years (Elektronikkbransjen 2007). It is therefore reasonable to believe that phones sold in 2010 would still be used in 2013 and 2014.

Another important factor to take into account is the size of the smart phones market share compared to feature phones. A survey from MEDIENORGE show that the percentage of people that use smartphones compared to feature phones in Norway increased from 46% in Q1 2011 to 71% in Q4 2012. In Q1 2013, the percentage of smartphones was 78%, but the requirements of a phone to classify as a smartphone changed in 2013 and now just requires the ability to connect to the internet so the high percentage from 2013 will not be comparable with pre 2013 statistics (MEDIENORGE 2013). In year 2011, 65% of the total mobile phone sales were smart phones, an increase of 15% from 2010 (Steffens and Andersen 2012).

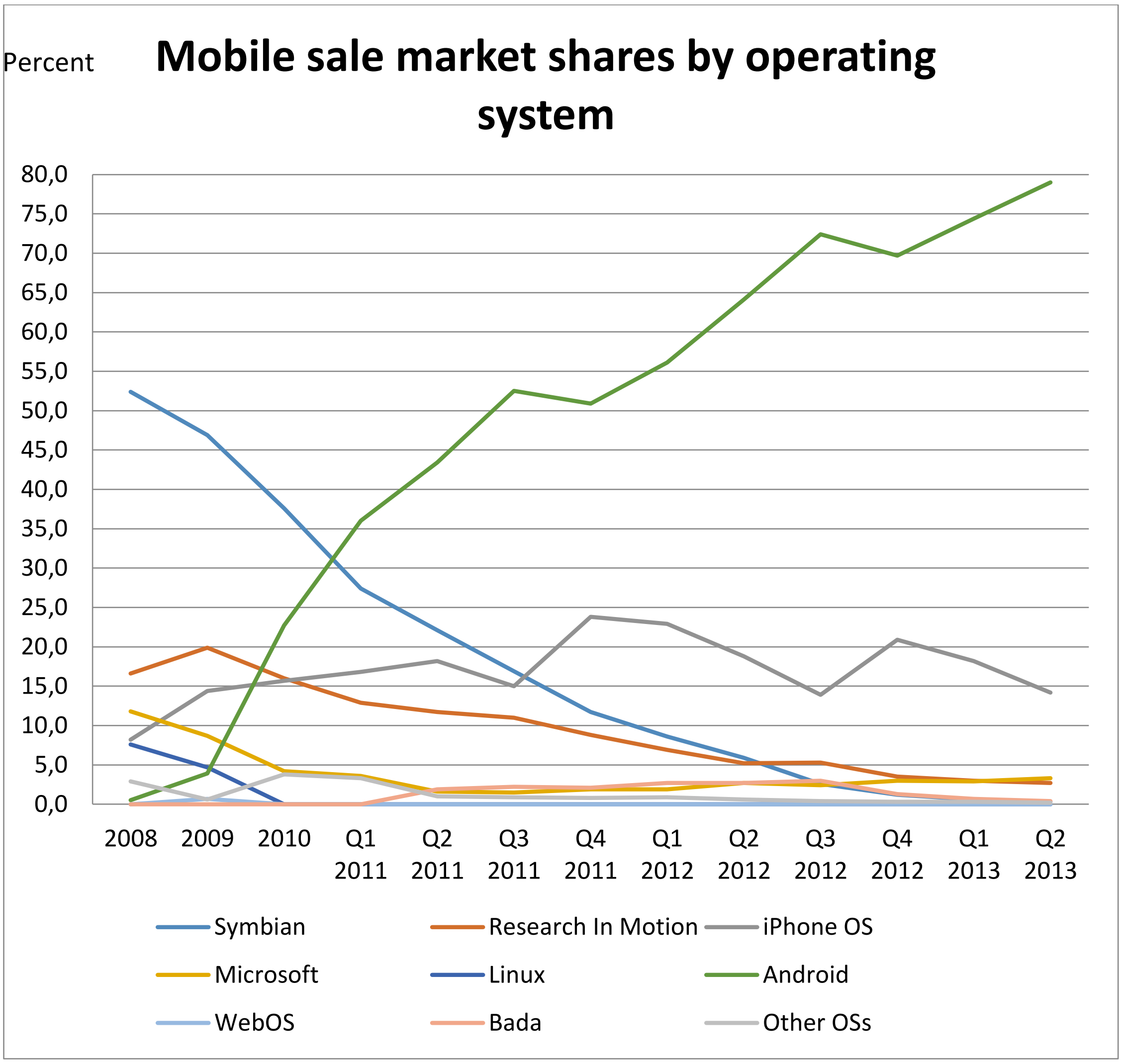


FIGURE 5.1: MOBILE SALE SHARES BASED ON OPERATING SYSTEM FROM 2008 TO Q2 2013 (GARTNER 2010, 2011D, A, B, C, 2012D, A, B, C, 2013A, B).

A summary of the market share developments are listed in Appendix C. The statistics from Figure 5.1 state that in Q2 2013 there were four major operating systems on mobile phones:

1. Android (miscellaneous vendors).
2. iOS (iPhone).
3. Microsoft (Nokia and other miscellaneous vendors).
4. Research In Motion (BlackBerry).

Android and iOS are the largest mobile platforms in the market with their competitors quite far behind, especially in Androids case. Looking at the statistics it does not look good for both Symbian and Research In Motion who has had a negative development in market shares since 2009. More information about the reasons for this are discussed in the platform reviews in section 5.2.3 and 5.2.4.

FIGURE 5.2: MOBILE MANUFACTURERS. REVENUES, PROFITS AND VOLUMES. Q1 2011 TO Q1 2012 (VISIONMOBILE 2012B).

Figure 5.2 (even though there are no exact numbers, it gives an indication of the mobile manufacturer landscape) shows that the manufacturers with the largest volumes are Nokia, Samsung and LG. However, the largest share is in the “other” category. This category includes several hundred different mobile producers that sell mobile phones in the developing world. This large share of other producers shows how commoditized the market has become.

Manufacturers sell very similar products/hardware, the devices are very similar, and consumers by the devices because of price, not the brand of the devices. This creates a lot of competition, driving the prices down making very small margins for profit. Even though the producers are selling a large number of devices, it gives the manufacturers very little revenue and almost no profit as shown in Figure 5.2 (VisionMobile 2012b).

Looking at the profit from 2011 in Figure 5.2 many manufacturers went out with a surplus, with Apple being the one with the largest profit and Sony Ericsson and LG barely showing on the chart. In 2012, the landscape changed and Nokia, RIM, LG and Sony Ericsson disappeared from the chart, leaving only Apple, Samsung, HTC and Motorola. It is interesting to see how much revenue Apple is able to create even though they are not selling as many mobile devices as the rest of the manufacturers do. Much of the reason for this is that they have a top brand with a good industrial structure and they are in charge of their own operating system development.

5.2 *Mobile platforms review*

The statistics in section 5.1 found that there is a significant fragmentation of mobile platforms and vendors in the global mobile market. In addition to knowing what OS's exist in the mobile market it is also important to know more about fragmentation within these platforms such as hardware. Based on the findings from chapter 5.1, this chapter will review the largest and most common platforms and analyse the fragmentation within the three largest platforms.

5.2.1 **Google Android**

Much of the reasons for Androids success are the fact that it is an open-source platform. In February 2012 had an average of 850,000 daily device activations on their platform and Google reported that there were 300 million Android powered units in use around the world and 450,000 mobile application available on their platform (PC World 2012). In April 2013, they had 1.5 million activations daily and almost 700,000 applications (Nelson 2013). In May 2013 Android reached a total of 900 million devices, more than 500 million just one year earlier (Bort 2013).

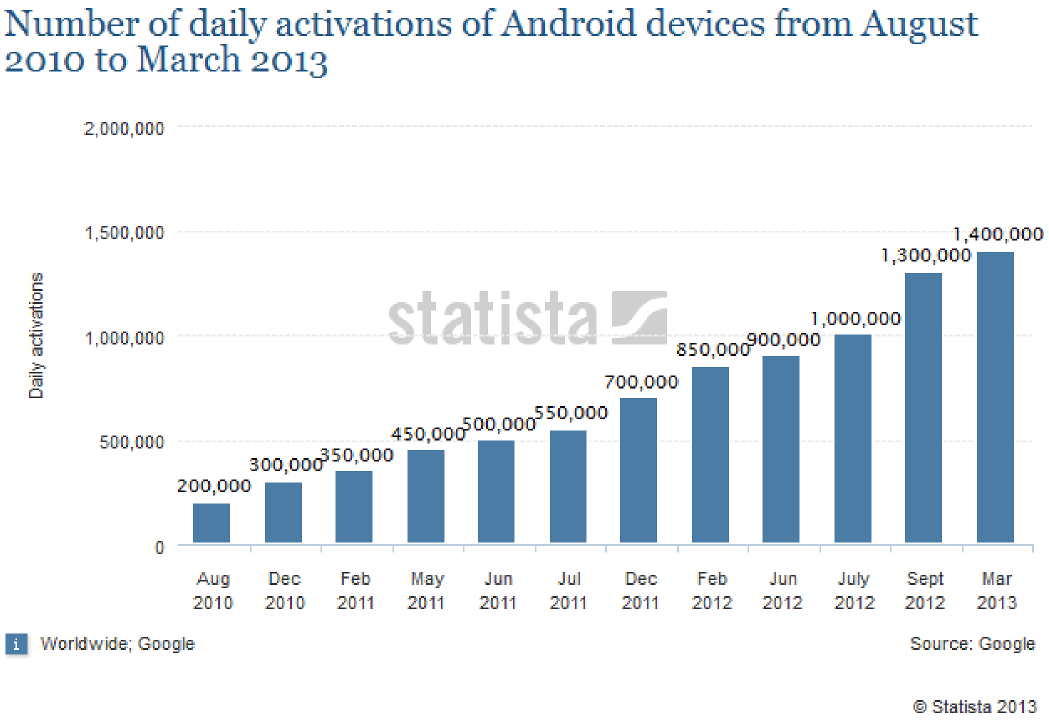
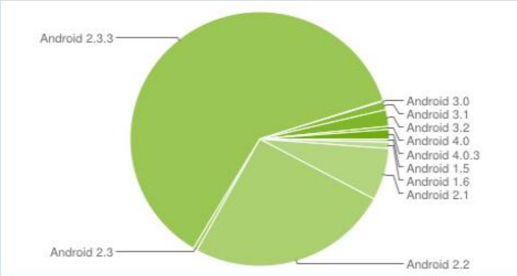


FIGURE 5.3: DAILY ANDROID ACTIVATIONS BETWEEN AUGUST 2010 AND MARCH 2013 (STATISTA 2013).

Since Android is open-source they allow their OS to run on many different devices from various manufacturers. With this approach, they are less dependent on a single manufacturer, but the various manufacturers compete with each other to sell devices.

Google has taken some steps in order to try to minimize the fragmentation of their Android platform with a non-fragmentation agreement clause for all official vendors on their platform. This states that all changes in their open-source OS needs approval by Google before release. This will ensure that there is only one version for each release of the Android OS (Burrows 2011).

There are however, some producers that has “forked” the Android codebase and created their own version of Android. These versions do not follow the same course as the main Android code base that Google control. Some of these include, but not limited to, oPhone from China Mobile, Wophone from China Unicom, Cyanogen and MiuiAndroid (VisionMobile 2011a).



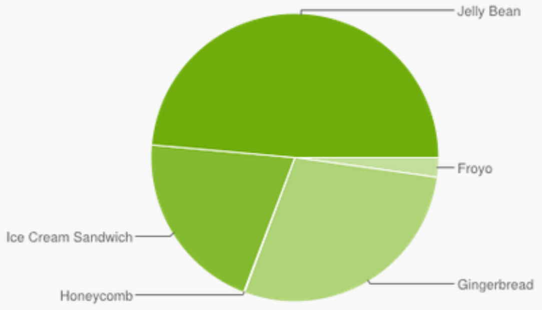
Platform	Codename	API Level	Distribution
Android 1.5	Cupcake	3	0.4%
Android 1.6	Donut	4	0.8%
Android 2.1	Eclair	7	6.6%
Android 2.2	Froyo	8	25.3%
Android 2.3 - Android 2.3.2	Gingerbread	9	0.5%
Android 2.3.3 - Android 2.3.7		10	61.5%
Android 3.0	Honeycomb	11	0.1%
Android 3.1		12	1.1%
Android 3.2		13	2.1%
Android 4.0 - Android 4.0.2	Ice Cream Sandwich	14	0.4%
Android 4.0.3		15	1.2%

Data collected during a 14-day period ending on March 5, 2012

FIGURE 5.4: ANDROID OPERATING SYSTEM DISTRIBUTION 05.04.2012 (GOOGLE 2012A).

Figure 5.4 shows the distribution of devices with Android 05.04.2012. The majority of mobile devices is running on version 2.3.3 to 2.3.7 with 61.5% and Android 2.2 with 25.3%. The rest is scattered on newer and older versions of the Android platform. Also important to notice is the API level of these OS versions. When developing an application the minimum API level the application requires to run, will determine the amount of phones it can run on. For instance, if the application requires API level 11, the application will not be able to install and run on more than 2.9% of the mobile phones with Android (Google 2012b).

Version	Codename	API	Distribution
2.2	Froyo	8	2.2%
2.3.3 - 2.3.7	Gingerbread	10	28.5%
3.2	Honeycomb	13	0.1%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	20.6%
4.1.x	Jelly Bean	16	36.5%
4.2.x		17	10.6%
4.3		18	1.5%



Data collected during a 7-day period ending on October 2, 2013.
Any versions with less than 0.1% distribution are not shown.

Note: Because this data is gathered from the new Google Play Store app, which supports Android 2.2 and above, devices running older versions are not included. However, in August, 2013, versions older than Android 2.2 accounted for about 1% of devices that *checked in* to Google servers (not those that actually visited Google Play Store).

FIGURE 5.5: ANDROID OPERATING SYSTEM DISTRIBUTION 02.10.2013 (GOOGLE 2013).

Figure 5.5 shows the distribution of Android platforms from 02.10.2013. Compared to Figure 5.4 the percentage of devices that are running version 4.0.3 have increased from 1.2%

to 20.6% (including version 4.0.4), while version 2.3.3-2.3.7 has decreased from 61.5% to 20.6%. So over a period of 1 ½ year Android have ported most of the devices from a point where most devices had an API level of 10 or less (95.1%) to a state where most devices have an API level of 15 or more (69,2%). Google also states that there are devices that are running other versions of Android, but Google does not list them since they only account for less than 0.1% of the total number of devices. In some sense, the Android platform is become less fragmented, since there are now fewer API levels in use on the majority of the distribution, but there is a larger scattering of the amount of devices on each API level, with still 30.7% of the devices on an API level of 10 or less.

One of the reasons for this fragmentized platform distribution comes from the OEM's (original equipment manufacturer) that produce the mobile phones for Android phones. When a new version of Android is released the OEM's have to tweak their HAL (hardware abstraction layer). Zeman (2012), state that fragmentation is inevitable because of the speed of improvements of hardware capabilities. This in turn creates a high release speed between major Android versions (VisionMobile 2011a, 32). The HAL software layer gives applications access to the hardware in the device. For instance, Android 4.0 (Ice Cream Sandwich) uses a new processor from Texas Instruments (TI OMAP) and this uses another HAL than older phone models. Therefore, in order for older phones to work with Android 4.0, they have to change the HAL. This means a lot of testing and in some cases requires new certificates. All of which is a time-consuming processes (Whitwam 2011).

This also affects the older Android devices. According to Figure 5.4 the majority of devices where the Gingerbread version (05.04.2012) and there are still 28.5% on the version 1 ½ years later, see Figure 5.5. The reason for this is the lack of incentives to update handsets that manufacturers already have sold to the end users. The manufacturers have a smaller profit margin on their devices since there is more competition on the Android platform (see Figure 5.2). OEM's producing handsets for the Android platform are more interested in developing new mobile phones that consumers will want to buy than to use money upgrading old devices free of charge (VisionMobile 2011a).

Some OEM's also create their own user interface to distinguish them from other producers. HTC has "Sense", Sony Ericsson has "Rachel", Motorola has "MotoBLUR", Samsung has "TouchWiz" and LG has "S-Class". All these nuances in interface design also add to the level of fragmentation for developers (VisionMobile 2011a).

On the Android platform, the number of different handsets (mobile phones) is enormous. A wide range of vendors distributes Android on their devices. On Androids main website[14] there is a list of hundreds of different mobile phones supported by the Android platform. All of these phones have very different hardware configuration and features. Figure 5.6 shows the menu button layout of some of Android models. The icons, arrangement and number of buttons vary greatly between phone models, even if the phones are using the same platform.

FIGURE 5.6: HARDWARE FRAGMENTATION OF MENU BUTTONS ON ANDROID PHONES (ENGADGET 2010).

The collection of Android phones also have a large range of different screen sizes as seen in Figure D.1, Figure D.6 and Figure D.7, which all have different screen sizes. The variations from one device to another can vary greatly as shown in the mobile phones in, a more detailed review of some of these hardware variations can be found in Appendix D. Android's developer guide, supply a guide for handling the available screen sizes on their platform (Google 2012c). From Android 1.6 (API level 4) Android grouped their screen sizes in four segments (small, normal, large and xlarge). However, from version 3.2 (API level 13) they deprecated these groups and instead uses four densities ldpi (low), mdpi (medium), hdpi (high), and xhdpi (extra high).

[14] Android device list (refer to: http://www.android.com/devices)

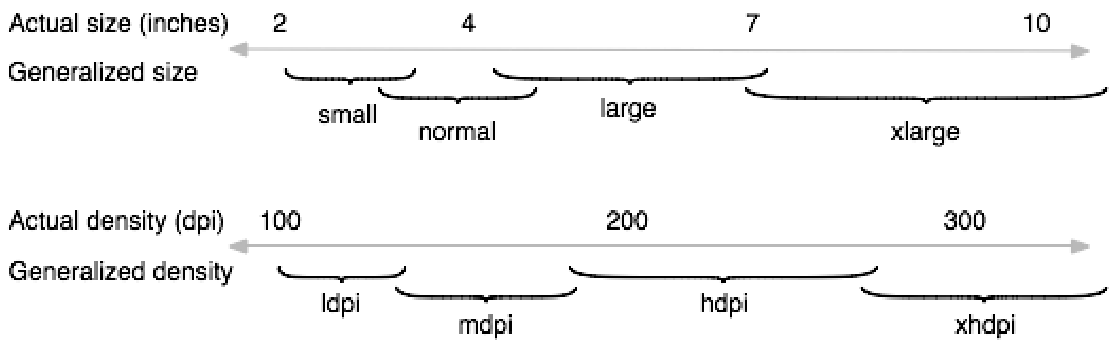


FIGURE 5.7: ILLUSTRATION OF HOW ANDROID ROUGHLY MAPS ACTUAL SIZES AND DENSITIES TO GENERALIZED SIZES AND DENSITIES (FIGURE IS NOT EXACT) (GOOGLE 2012C).

This does not mean that there are only four different screen sizes available on the Android platform. However, when developing screen layouts the phone will report that it fits one of these sizes. This means that two devices that report they have a certain mdpi resolution might have a physically different screen size and density. Nevertheless, the Android system will adjust your layout to fit the screen automatically. Although Android will scale your application to fit the current screen, Android advise developers to create layouts for different screen sizes (Google 2012c).

Zeman (2012), state that the possibility to customize the Android platform with so many different hardware combinations the fragmentation is not by design or by purpose, it simply is fragmented.

5.2.2 Apple iOS

The counterweight to Androids fragmented platform is the Apple iOS platform. Apple have also had great success considering that their platform is proprietary. They manufacture their own devices and their own operating system. In addition to this Apple have chosen only to manufacture a small amount of different devices. This is the main reason that Apple does not have the same level of fragmentation issues that the Android platform have. This strategy gives a higher level of device uniformity and improves standardization. This gives Apple more control on the devices and helps developers to ensure that their application will work as intended. The main drawback with this is that they are likely to reach a smaller portion of the market. Apple was in 2010 the only platform that used this approach (Holzer and Ondrus 2010).

Since Apple is making their own phones and are developing their OS themselves, it is easier for them to fine-tune their OS to the devices available on their platform. There are however changes and variations to every release (Apple 2012a), but most of them has to do with the phone features, such as increased processing power, camera resolution etc. As seen in Figure 5.8 the phone layout has remained almost the same for each version.

FIGURE 5.8: DIFFERENT IPHONE MODELS. FROM VERSION 2 TO 4 (MOBILEVENUE 2010).

The latest iPhone has a larger screen than the older ones, as shown in Figure 5.9. Going from 3.5 inches to 4 inches, this adds to the level of fragmentation of iPhone.

Color	Black & Slate; White & Silver	Black; White	Black; White
Weight and Dimensions[1]	Height: 4.87 inches (123.8 mm) Width: 2.31 inches (58.6 mm) Depth: 0.30 inch (7.6 mm) Weight: 3.95 ounces (112 grams)	Height: 4.5 inches (115.2 mm) Width: 2.31 inches (58.6 mm) Depth: 0.37 inch (9.3 mm) Weight: 4.9 ounces (140 grams)	Height: 4.5 inches (115.2 mm) Width: 2.31 inches (58.6 mm) Depth: 0.37 inch (9.3 mm) Weight: 4.8 ounces (137 grams)
Display	4-inch (diagonal) Retina display 1136-by-640 resolution 326 ppi	3.5-inch (diagonal) Retina display 960-by-640 resolution 326 ppi	3.5-inch (diagonal) Retina display 960-by-640 resolution 326 ppi
Cellular and Wireless	GSM model: GSM/EDGE UMTS/HSPA+ DC-HSDPA CDMA model: CDMA EV-DO Rev. A and Rev. B LTE[3] Wi-Fi (802.11a/b/g/n; 802.11n on 2.4GHz and 5GHz) Bluetooth 4.0 GPS and GLONASS	GSM/EDGE UMTS/HSPA – CDMA EV-DO Rev. A[2] – Wi-Fi (802.11b/g/n; 802.11n on 2.4GHz) Bluetooth 4.0 GPS and GLONASS	GSM model: GSM/EDGE UMTS/HSPA – CDMA model: CDMA EV-DO Rev. A – Wi-Fi (802.11b/g/n; 802.11n on 2.4GHz) Bluetooth 2.1 + EDR GPS

FIGURE 5.9: COMPARISON OF SOME OF THE IPHONE MODELS (APPLE 2012C).

Apple does not release any official information about the current distribution of their platform. There are however user-data statistics available. The graph in Figure 5.10 shows us the distribution (based on data from various application developers) of iOS versions in the period July 22 (2010) to March 1 (2012). The graph in Figure 5.10 shows that as one version of iOS is released the share of the other versions decrease.

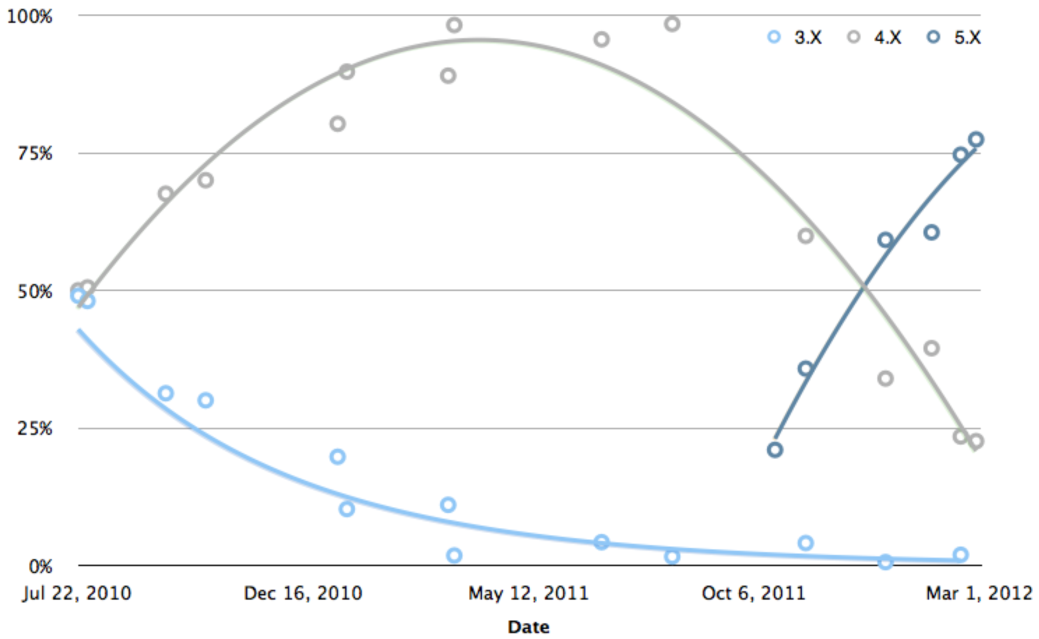


FIGURE 5.10: APPLE IOS DISTRIBUTION STATISTICS. JULY 22, 2010 TO MARCH 1, 2012 (PXLDOT 2012).

Figure 5.11 shows the iOS distribution for an application called "Audiobooks" between March 6 (2012) and April 15 (2012). The free version of this application receives around 100,000 downloads per week. Version 5.1.0 of iOS was released on March 7, 2012 (Apple 2012b) and in just 11 days the percentage of phones having version 5.1.0 increased from 0% to around 60%. An additional 10% - reaching 70%, one month later. One possible reason for the decrease in amount of upgrades and a peak of 70% may be because of backwards compatibility issues with older versions of the iPhone (iOS 5.1.0 is only supported on iPhone 3gs and above, in addition to 3rd generation iPod touch (and up) and iPad (Apple 2012b).

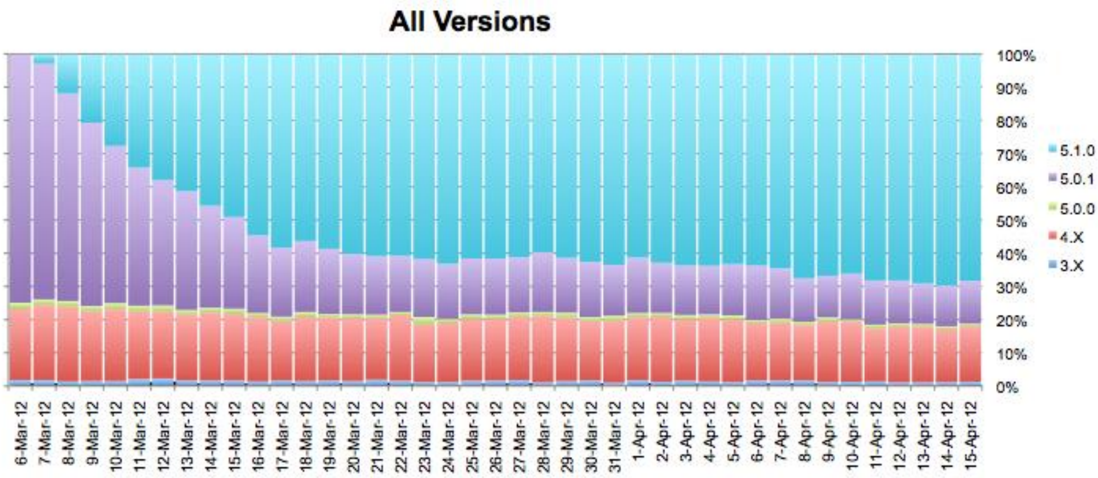


FIGURE 5.11: IOS DISTRIBUTION FROM MARCH 6 2012 TO 15 APRIL 2012 (SMITH 2012).

Zeman (2012) stated that iOS 6 features such as FaceTime[15] (over a cellular network), 3-D map Flyovers[16], and expanded Siri[17] capabilities would only be supported by iPhone 4S and iPad retina (and up), iPad2 would only get 3-D fly over and older devices such as iPhone 4 and 3S (which was still in sale at the time) would not get any of these new features. The original iPad would not get access to iOS 6 at all. According to Apple this was by design because the A5 processor was required for these features. However, fragmentation could be a business strategy since Apple always reserve some of the best features for its newest hardware. iPhone 4 did not get Siri[17] with iOS 5 like the iPhone 4S. Even if iPhone 4 was cheaper than the iPhone 4S the sales numbers for iPhone 4S was superior to the iPhone 4. The same happened with the iPad 2, it had a user-facing camera and got access to FaceTime[15], the original iPad did not.

5.2.3 Symbian / Windows Phone

Looking at the statistics from 2008 to Q4 2011 in table 1, Symbian has gone from a leading market share of over 52.4% to a third place with only 11.7%. One of the reasons for this decrease in market share is the introduction and popularity of the iOS and Android phones. And as a result of this Nokia decided to give up the Symbian platform and become partners with Microsoft in a new strategic alliance to be able to compete with iOS and Android (Nokia 2012). Nokia will however continue to support the platform until 2016, but have outsourced this to Accenture (Nokia 2011).

The cooperation will feature the best from both companies and will include Bing, Xbox and Office. Nokia has a global supply and distribution network and are able to deliver their mobile phones to consumers almost all over the world. With the Windows Phone OS being similar to the iOS and Android approach, with touch screen optimization and easier application development, and with the solid and well-known Nokia platform. This alliance can become a real competitor for iOS and Android.

Even though Windows Phone and Nokia still has a small percentage of the market today, their market shares will probably increase if they get developers interested in developing towards their platform. In a survey on Developereconomics.com (VisionMobile

---

[15] FaceTime (refer to: http://www.apple.com/no/ios/facetime/)

[16] Flyover (refer to: http://www.apple.com/ios/maps/)

[17] Siri (refer to: http://www.apple.com/ios/siri/)

2012b) developers are hopeful to Windows 8 and their cross-platform development capabilities between Windows 8 desktop computers and tablets.

Fragmentation of the Windows Mobile platform is also less than on its major competitors. Most of the mobile devices that was sold with Windows Phone 7 were upgraded to Windows Phone 7.5 with all its features. An exception is when Windows introduced the Windows Phone Tango that had less expensive hardware (with down to as little as 256 RAM being the major issue) and could not access all features because of its memory limitations. Much of the same issues are also found in BlackBerry OS and BlackBerry phones discussed in the next chapter (Zeman 2012).

5.2.4 BlackBerry (Research in Motion)

The BlackBerry platform is not analysed in detail due to its limited market in Norway and the scope of this thesis. There are however a few points that is important to notice about this platform for developers.

The platform popularity graph from the last couple of years in Figure 5.1 shows that their platform reached almost a 20% market share in 2009. Since then is has decreased steadily every year and lost market shares to their competitors, with a 2.7 % market share remaining in Q2 2013 (Gartner 2013b).

Their developer platform facilitates developers to develop in a known environment. Either HTML applications with native capability support integrated in the BlackBerry platform or native C/C++ applications (even Adobe AIR ActionScript or Java applications). Developers that have developed applications for Android (java) can also port existing applications to BlackBerry with their Android Runtime application (for Android 2.3.3 applications) (Research In Motion 2012a).

In 2012 BlackBerry had over 75 million users (Research In Motion 2012b).

5.3 *Monetization*

With the introduction of new mobile platforms the value chain and market structure for mobile applications have evolved and changed (Holzer and Ondrus 2010). The distribution process usually consists of three main parts: developers, application portal and the consumers, as illustrated in Figure 5.12.

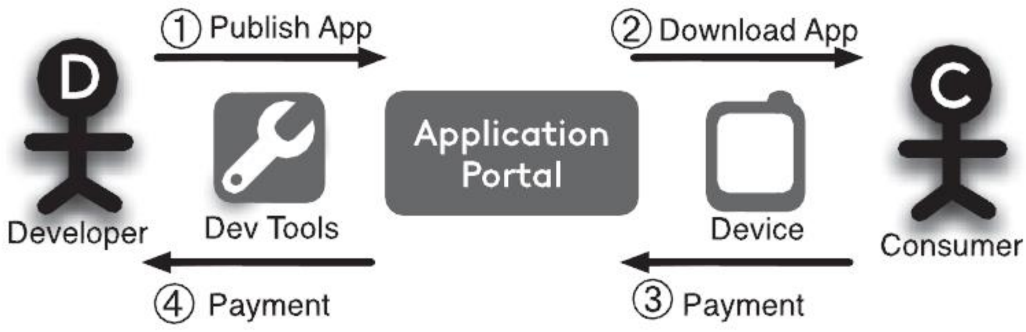


FIGURE 5.12: A MOBILE APPLICATION DISTRIBUTION PROCESS (HOLZER AND ONDRUS 2010).

When developers are ready to release their application, they can distribute their application through an application portal. The portal acts as an intermediate between the application provider and the end-user customers and acts like a typical two-sided market. There are many of these application stores available, both official from the different mobile platforms and unofficial portals. The easiest to use are the official portals, since they come pre-installed on the mobile phones.

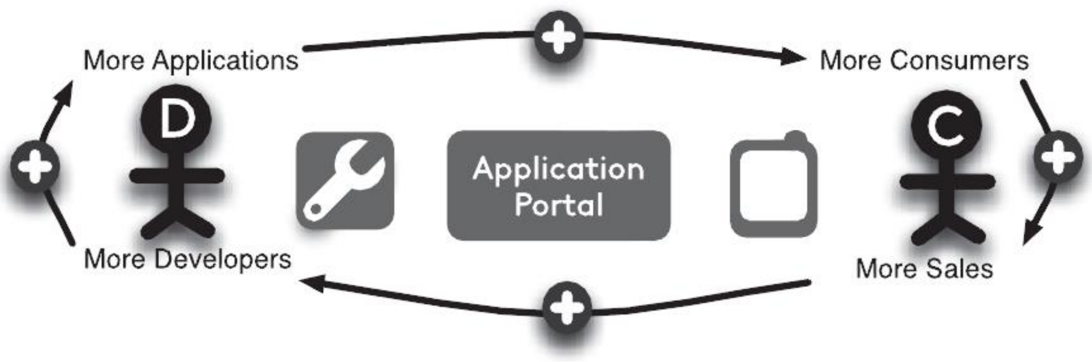


FIGURE 5.13: POSITIVE FEEDBACK LOOP PROCESS.(HOLZER AND ONDRUS 2010)

As the number of available end-users/consumers increase, the interest with the developers will also increase, since a larger customer base will increase their possibility for higher revenue. The same effect applies to the other side where the number of consumers/end-user will increase because of the increasing amount of available application on the platform, as illustrated in Figure 5.13.

FIGURE 5.14: AVERAGE MONTHLY DEVELOPER REVENUE POTENTIAL FOR EACH PLATFORM (VISIONMOBILE 2013B).

Figure 5.14 shows the average monthly revenue potential a developer has with each platform and what types of revenue models that are the most popular with each platform. iOS has the highest potential revenue for developers with 5200 USD. Android comes second with 500 USD less at 4700 USD. Windows phone ranks as number three with 3600 USD and HTML5 in fourth with 2900 USD, almost half the revenue potential that iOS developers has. VisionMobile (2015), state that 17% of developers that were interested in making money does not make anything at all and 18% make less than 100 UDS per month. A total of 52% make less than 1000 USD per month. This group also include full time developers that only have mobile apps as their income.

A survey by (Global Intelligence Alliance 2010, 16), found that “Twice as many publishers see higher user adoption and usage volume on native apps”, and “30% respondents with both interfaces see over double usage volume over the native application”. The same report also found that “Usage stickiness appears stronger for web apps, while native app use tends to peak at download” (Global Intelligence Alliance 2010, 17).

5.4 *Developer platform adoption*

This chapter will analyse developer’s interest for the different platforms and how many platforms each developer supports. Information about the developer community behind the platforms can also provide information about why developers choose some platforms over others. As seen in chapter 5.3, investigating these statistics is more important than ever since there is a closer connection between the mobile developers and the end-users than there have ever been before.

In a survey by VisionMobile (2012b), developers where asked what platform they were planning to stop developing for. The results are similar to the sales statistics listed earlier in the thesis. Figure 5.15 lists platforms where developers planned to stop application development. Similar to the sales statistics from Table C.1, Symbian and Blackberry are among the platforms that is losing the most interest with developers. Where about half of the developers currently developing on their platform are planning to stop. There are also smaller platforms that were losing developer's interest. Although Samsung's Bada had some increase in market share in 2012, 49% stated that they would stop developing for the platform. Bada ended up with a 1.3% market share in 2012 and was in Q2 down to 0.4%.

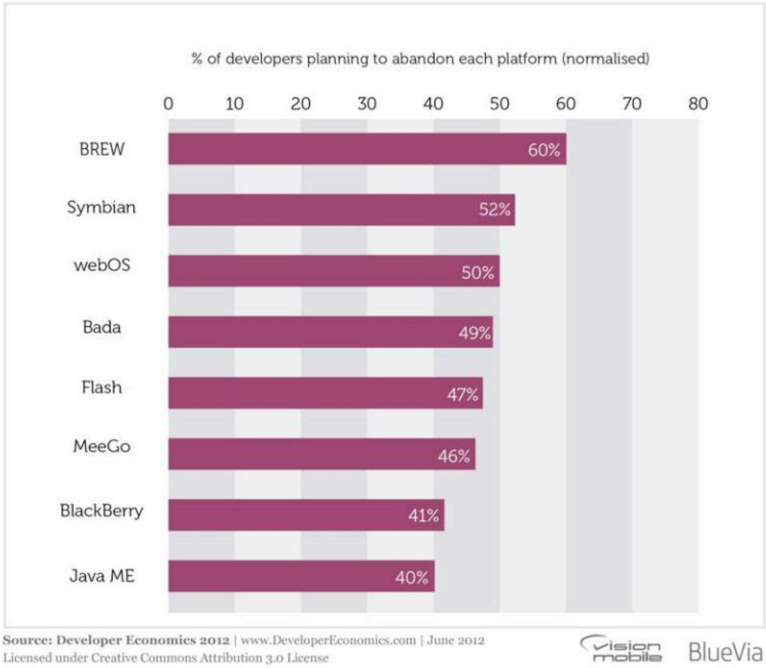


FIGURE 5.15: PLATFORMS WHERE DEVELOPERS PLANNED TO STOP DEVELOPMENT (VISIONMOBILE 2012B)

Figure 5.16 shows the result of an annual survey conducted by Appcelerator and International Data Corporation. The survey was conducted in August 2012 where around 5000 developers participated, making it the world's largest mobile developer survey at the time (Appcelerator 2012a).

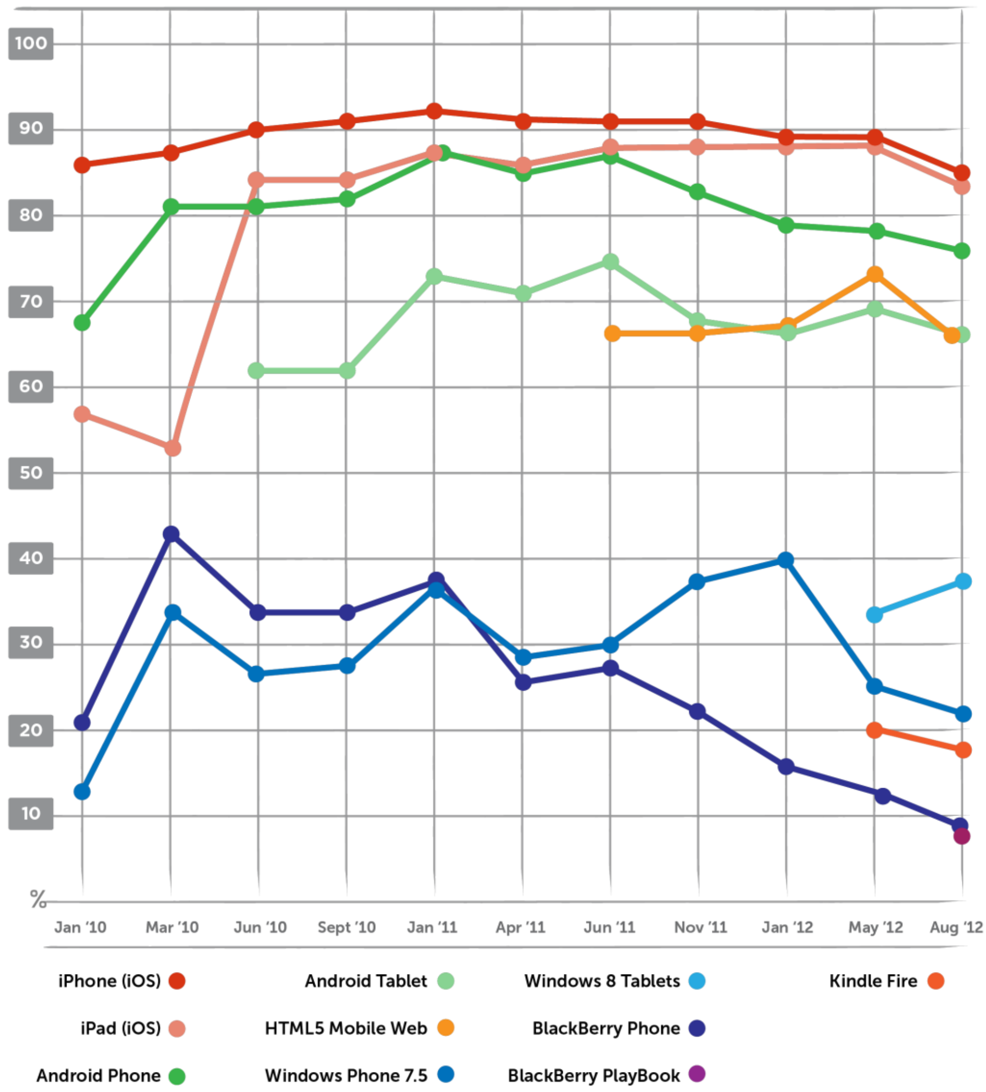


FIGURE 5.16: PLATFORM POPULARITY WITH DEVELOPERS (APPCELERATOR 2012A)

The most popular platform is definitely the iOS platform. Since 2010, they have been the top choice for developers with around 90% (with a peak of 92%) of the developers saying that they want to develop for iPhone. The iPad has become more and more popular with developers and since June 2010, it has become more popular than the Android platform. The iPad has also become almost as popular as the iPhone. The reason for this popularity is the massive fragmentation with Google's Android and that developers choose to develop native apps on the iOS platform first, before developing for Android.

The Android platform is right behind the iOS platform in popularity with a peak of around 80-86% of the developers saying that they want to develop for the Android platform. The Android tables are however quite behind the iPad with a peak of 75% and the iPad peaking at 88%. The iPad is also outselling Android tablets three to one.

All platforms except for Windows 8 tablets (going from 33% to 38%) have lost some interest with the developers. However, Windows phone 7.5 has lost some interest in 2012 going from a peak of 40% to almost half at 21%. BlackBerry has plummeted in popularity the last year going from 38% in January 2011 to an all-time low at just 9% in August 2012.

HTML5 started with 67% June 2011 and peaked May 2012 with 73%, but has fallen in popularity in August 2012 back to around 67%.

The most important criteria's for developers when choosing a platform to develop for is as follows:

1. A large installed base of devices (53%).
2. Low cost of devices (37%, 38% ranked this criterion as number three).
3. Revenue potential (43.2%, 34.3% ranked this criterion as number two).

This ranking gives the existing platforms a head start with developers. Android (79% market share, Q2 2013) and Apple (14.2% market share, Q2 2013) have the largest installed base and for others to catch up they have to sell a significant amount of devices. Apples mobile phones are in the high-end of the price range, but they are also marketing towards their competitor's high-end phones. An advantage for Android and other platforms is that they have manufacturers that also sell phones in the lower-end of the price range for consumers that do not wish to spend a large amount of money on their mobile phone.

The revenue potential is also greater where there are many consumers, but with many available applications on the Apple and Android platform, there is also difficult to attract attention in the myriad of available applications.

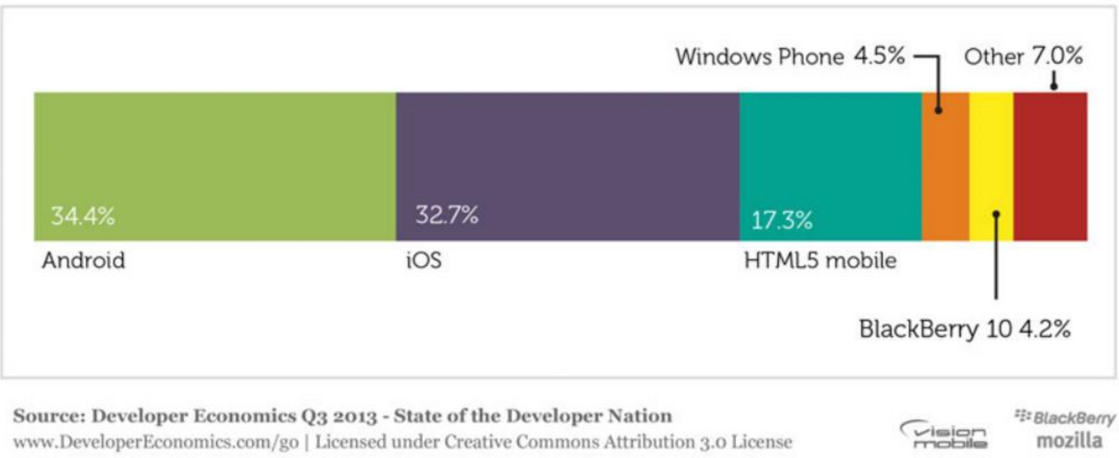


FIGURE 5.17: DEVELOPERS MAIN PLATFORM CHOICE (VISIONMOBILE 2013A, B).

A survey published by Vision Mobile in July 2013 (over 6000 developers participated, making it the largest and most global mobile developer survey up until 2013) found that 84%

of all developers use either iOS, Android or HTML5 as their primary platform. Android and iOS have the largest developer shares with 34.4% that prefers Android while 32.8% prefer iOS. 17.3% uses HTML5 as their primary platform (VisionMobile 2013a).

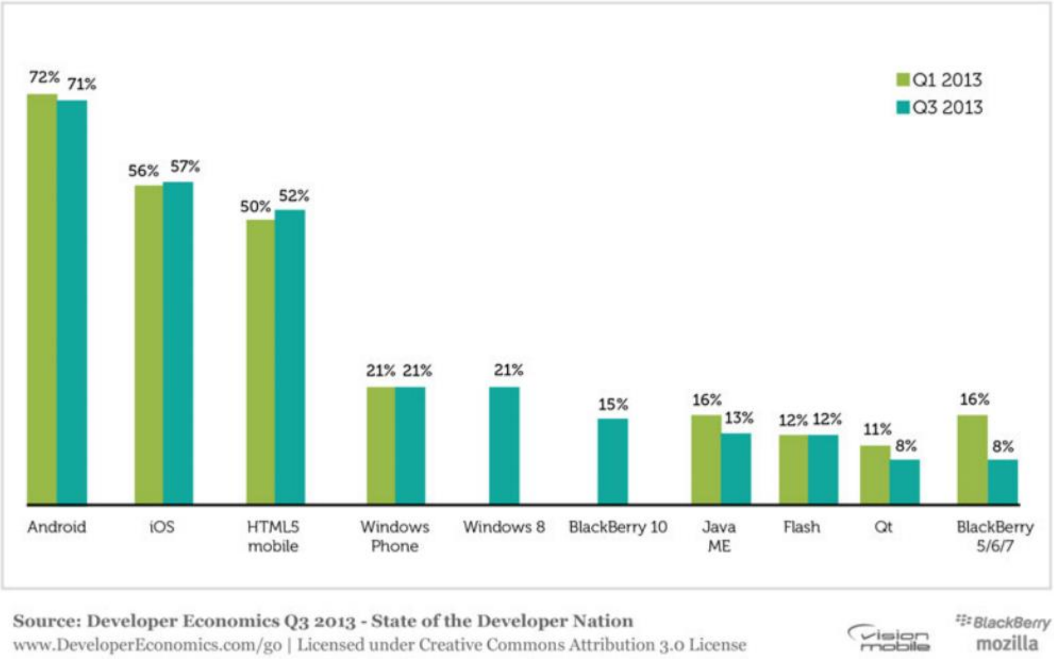


FIGURE 5.18: DEVELOPERS USING EACH PLATFORM Q3 2013 (VISIONMOBILE 2013B).

Figure 5.18 shows the percentage of developers developing for each platform. Most developers develop applications for more than one platform. In 2011, developers developed for an average of 3.2 platforms while in Q1 2013 the average number of platforms where 2.6. In Q3 2013, this has increased again to an average of 2.9 platforms. VisionMobile state that this is the first observation of a shift towards diversification since their earlier research in 2011 (VisionMobile 2013b).

The type of audience is also an important factor when selecting a main platform. In the consumer segment, iOS and Android are equally important, but when targeting enterprises iOS is preferred over Android while HTML5 counts for 25% of developer's main platform (VisionMobile 2013b).

The developer economics survey by VisionMobile (2013a) also found that 78% of developers that where interested in generating revenue from their mobile applications developed for two or more platforms concurrently. In 2013, mobile developers used 2.6 mobile platforms, 2.7 in 2012 and 3.2 in 2011. This shows that the interest for more than three platforms are narrowing downwards to only two platforms. "The Android-iOS duopoly in smartphone sales is gradually creating a concentration of developers around these two platforms" and that developers that do not develop for one of these two platforms, on average generate half the revenue as the ones who do. Eighty percent of the respondents develop for these two platforms, “making them the baseline in any platform mix” Forty-nine percent, uses one or two mobile platforms concurrently and 75% use up to three mobile platforms. VisionMobile’s research shows that revenue is higher when developers are using more platforms and that loyalty to only one platform does not pay off. However, VisionMobile also states that the higher revenue for developers that are working on four or more platforms is probably a result of extending a mobile application that has already become successful on the existing platforms. Developing for multiple platforms is not something that every developer is able to do since it requires a development team that is large enough to support several platforms VisionMobile (2013a, 22).

5.5 *Bring your own device (BYOD)*

Fragmentation of the mobile landscape is also a challenge for enterprises. Every user has a unique set of preferences for what phone they want to use, which ultimately add up to fragmentation of devices, that in turn affects developers. Many companies allow their employees to use their own device at work, hence the term “bring your own device” (BYOD). This method is popular with small companies and companies with a temporary staff. There are also similar terms such as choose your own device (CYOD), where pre-approved phones are selected by the company. These phones can either be paid for by the employee or the company provide a stipend, so the employee can keep the phone for the duration of their employment. The third option is called company-issued, personally-enabled (COPE) where the mobile device is paid for by the company, but is also allowed for personal use. This is the model closest to the traditional Corporate-owned business only (COBO) method (Sutton 2014).

These models also seem to be used in some of the ELA customer companies, as seen in the survey presented in section 7.3. The survey shows that some employees have multiple phones, possibly to differentiate between personal and corporate use, while others only have one phone. It seems like the majority of companies allow users to use one device for both private and business applications, and possibly, a mobile device that they prefer to use and have chosen themselves.

This demand was first created by iOS and was a straightforward task for businesses that allowed access to corporate apps without compromising company data. When Android was introduced they allowed changes to the OS that in term also cause fragmentation. Companies that allow BYOD also introduce degraded performance, makes application access more difficult, compromises security (Apperian 2012) and reliability and compatibility goes down (Sutton 2014).

5.5.1 Managing multiple devices in an enterprise

According to Sutton (2014), business traditionalists may try to counter the security and compatibility issues with the COPE method. However, this method raise some privacy issues since the company have visibility and ownership of the devices. CYOD solves many of the issues, and is especially good for dealing with Android fragmentation. However, this method is contingent on companies keeping an updated list of devices. The job of updating this list will consume IT resources and is something businesses will struggle with. CYOD will provide some level of satisfaction for employees to freely choose a device, however some employees may still try and use their own unapproved device. The method used in companies comes down to the need and requirements of each organization (Bender 2013).

Another way of tackling multiple devices in a company is to implement a Mobile Device Management (MDM). MDM allows companies to implement software policies for network access, application download and usage, service usage and device security. MDM manage the whole device or a virtual portion. MDM offers remote lock and wipe, policy enforcement and data tracking. MDM offer device control, but not security for the mobile user since the entire device is managed by the policy and the company's IT department. However, this type of management over time is a costly and complex process. It also reduces the benefit of BYOD as it restrict the usage of each device. Another method can be to implement a Mobile Application Management (MAM) platform. This is less complex than MDM. Usually users install apps via an e-mail link or through an enterprise application store (unique for the enterprise) where approved applications can be downloaded. All application data sent between the device and corporate resources are encrypted. MAM also allows for device scanning searching for any rogue applications- and policy violations (Apperian 2012) (Hess 2014). A third option can be Mobile Content Management (MCM), which focuses on authentication, authorization and access to document repositories where various participants can share documents and information. Some of the methods used are a username and password, IP addresses and device authentication. MCM deliver a secure application that access a single repository, while MAM delivers apps that performs a type of function. MCM is the least intrusive method of the three types of device management (Hess 2014).

These types of approaches will not be further reviewed. However, it is useful for enterprise application developers to know that some companies allow for BYOD and similar methods, since it can increase the level of fragmentation. It is also important to be aware that some companies might implement methods that can restrict user and app behaviour.

5.6 *Fragmentation summary*

This chapter have reviewed the current state of the art in the mobile market, with regard to topics that relate to mobile fragmentation and mobile development. There are many evidence that shows that the mobile market contains multiple types of fragmentation which developers have to consider when developing mobile platforms. Mobile apps have become a billion dollar industry. In 2014 mobile e-commerce generated significantly more revenue than all other revenue models combined, with App Store sales on second place (VisionMobile 2015). Developer support and application drive on the platform will give a competing edge against other platforms and creates a synergy effect for the developers since the potential user reach and monetization increases.

The review of the mobile platforms in section 5.2.1 to 5.2.4 show that there is fragmentation on each of the four largest mobile platforms. There are multiple devices, each with a unique combination of hardware for each platform. Each mobile platform also has multiple versions of their operating systems. All in which have to be considered by mobile developers.

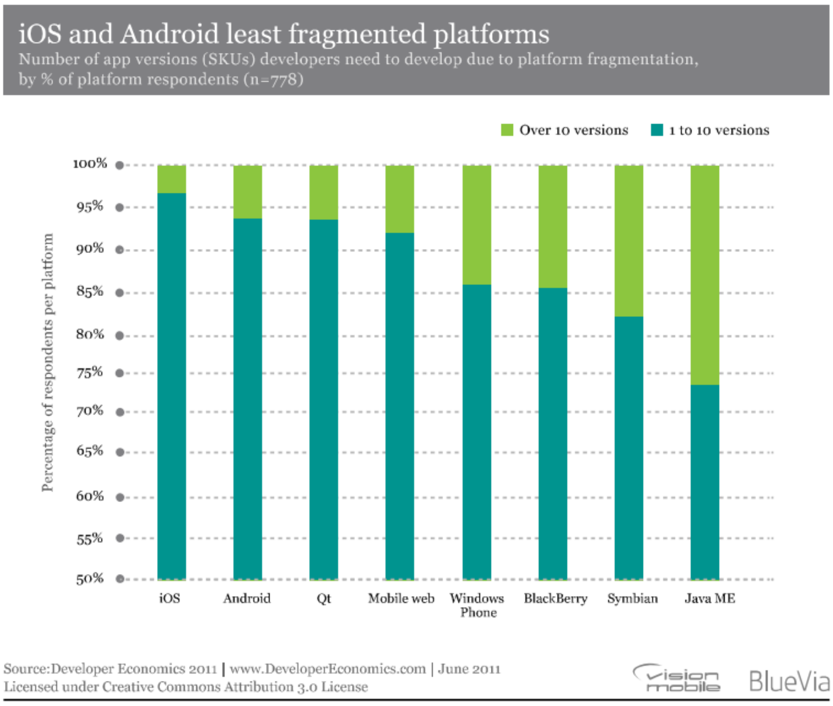


FIGURE 5.19: NUMBER OF APP VERSIONS DEVELOPERS MUST DEVELOP FOR EACH PLATFORM (VISIONMOBILE 2011B).

However, as seen in Figure 5.19 a survey by VisionMobile (2011b) found that the two largest platforms Android and iOS are also the least fragmented platforms compared to other platforms. Across all platforms, developers have an average of four different versions per application. Android have on average six versions of their application, which is the same as for mobile web applications. However, it is important to note that a web application will be able to support multiple platforms while native applications are still restricted to one platform. The Android platform who has a reputation of being fragmented is relatively un-fragmented compared to Windows Phone, Java ME, BlackBerry and Symbian that on average requires developers to create about twice the number of app versions for these platforms. In addition to having multiple versions of each application on one platform, mobile developers on average develop for 2.9 different platform (Q3 2013, see section 5.4). With the most popular platforms to develop for being iOS, Android, HTML5 and Windows Phone. With an exception of HTML5, which is not an operating system, this is also the most popular platforms with consumers in the global market.

6. Development Methods and Software Tools

As seen in chapter 5 there are multiple fragmentation types in the mobile market that developers have to consider. In addition to fragmentation it is also important to review available development options and what possibilities and limitations there are with each method, before a proper development strategy can be selected. This is especially true for developers that need to support more than one OC, for instance multiple operating systems. This leads us to RQ 2, what methods and software tools exists to minimize fragmentation issues in mobile application development? This chapter will define the operational questions OQ3 and OQ4 from the GQM by analysing available development methods and tools for the three mobile applications types found in section 2.4.

6.1 *Native applications*

With a native application, there is simply not an option to create one application that can run on multiple operating systems (OS). As an alternative an emulator can be used to run the application on other devices, but this option require third party software installed on the mobile device and the application would only available to those that had a suitable emulator installed. A native application will be running on the device itself and installs on the OS.

As reviewed in section 2.5, developing applications that have to run on multiple OS (multiple OC's) have to use MANUAL-MULTI and create a completely new codebase for each OS the application needs to support. In addition to the MANUAL-MULTI, each OS version of the application needs to support multiple OC's on the current operating system. This is possible with the FITS-ALL, AIM-LOW method, but this will lower the potential of our application to a minimum on all OC's. A better way is to use the ALL-IN-ONE, SELF-ADAPT or DEVICE-ADAPT methods. With the SELF-ADAPT, the application scales according to the device it is running on. For instance, the GUI scales up/down to fit the current screen. The Android and iOS platform does this automatically. In Android this is called "alternate resources" while in iOS it called "storyboard". With the DEVICE-ADAPT method, developers can create individual configurations/layouts for a specific resolution. In Android, developers provide alternative resources not only for screen size or pixel density, but also for any specific device configuration. Configuration qualifiers (such as language, region, layout direction, smallest width, screen orientation etc.) identify android device configurations. The number of different device configurations are mathematically huge and there is no practical way to supply all the alternative resources, so developers have to choose the configuration qualifiers they want to support in order to create the best agnostic user experience (Liao 2014). When running an application designed for mobile phones on a tablet, the application will look stretched and unnatural since tablets have a larger screen. If the application detects the current OC, it will instead load and use the configuration specifically designed for the OC. On the iOS platform there are less fragmentation challenges than on the Android platform, but when developing for the iPad developers also have to design the application specifically for this usage as well. In Apples manual, they do not recommend just using a scaled up version of the iPhone application on an iPad since the user experience greatly diminishes. Graphical elements need to be of higher quality. Apples manual also state that most individual UI elements are available on both devices, but that the overall layout differs dramatically (Apple 2013).

6.1.1 Native Android development environment

The Android Software Development Kit (SDK) is a collection of libraries, tools, documentation and samples that are required to run and develop Android applications. The SDK has a modular structure and the SDK manager allows developers to update and download additional components if the application that the developer is creating requires it (Wolfsen 2013).

Android Studio is the official integrated development environment (IDE) for the Android platform. The SDK download has Android Studio included. Android studio comes with a code editor, code templates, GitHub[18] integration, multi-screen support, virtual devices for debugging/testing and support for build automation with Gradle[19]. In addition to this package, it is required to install the Java Development Kit (JDK) from Oracle. Developers can use a number of different OS platforms when developing Android applications such as Windows, Mac OS X and Linux (Google 2015b).

Earlier development on the Android platform used the Eclipse IDE instead of the new Android Studio. This required the Android Development Tools (ADT) in addition to the SDK (Google 2015a).

Google manages the Android ADT and the main Android platform, but the group that manages the ADT is still different from the one that manages the main platform. The standard SDK is often linked to the platform release, but not always. Even though Android is open-

[18] GitHub (refer to: https://github.com/)

[19] Gradle (refer to: https://gradle.org/)

source, the most current codebase is kept behind closed doors and is first released to manufacturers and other insiders before it is released to the public. The ADT accepts outside contributions to the current codebase (Wolfsen 2013).

6.1.2 Native iOS development environment

To develop native iOS applications one need a computer that runs Mac OS X (version 10.9.4 or later). In addition, developers need the Xcode IDE. Xcode is the same as the Android Studio for Android developers and contains a source-code editor and a graphical user interface editor. Furthermore, the developers also need the iOS SDK. This will add tools, compilers and frameworks for iOS development to Xcode (Apple 2015).

6.1.3 Native Windows phone development environment

Windows phone developers requires a computer with Windows and the Visual Studio Community IDE. Visual Studio includes application templates, code editor, debugger and a windows phone emulator. Visual Studio Community also includes the Windows Phone SDK and the Windows SDK (Microsoft 2015).

6.2 *Cross-platform applications*

As stated in section 6.1 there are simply no way to develop a pure native application that can run on multiple OS. To achieve this, developers have to create either a web application or a type of hybrid application. Both of these application types have cross-platform capabilities. This section will review some of these methods and how they allow applications to run on multiple OS.

6.2.1 Web applications

Web applications run in browsers and not on the operating system of a device, this makes web applications platform and operating system independent. One of the main advantages of the web is that it is based on standards. These standards are implemented and supported by browsers that are available on all platforms and operating systems. Web browsers are an important feature on newer mobile devices and comes pre-installed on the operating system. With the introduction of high-speed internet (3G/4G) and affordable costs of using it combined with high availability, mobile users are now seem to be more online than they have ever been before.

HTML content is static until you change it. However, web pages used in conjunction with a server-side programming language is able to change the content dynamically. Before the server sends the web page to the end user, the server converts all server-side code into a "static" page so that the user only sees HTML, CSS and JavaScript. Server side code is more sophisticated that HTML and can be used to handle business logic. Server side code can for example, perform calculations and be used to retrieve information from databases (Lane 2012). Developers can also add more functionality to web applications by using JavaScript. JavaScript is a script language that add functionality to webpages that extends the basic HTML functionality, such as events caused by user interactions (Mozilla 2012). AJAX (Asynchronous JavaScript and XML) also allow HTML pages to become dynamic. JavaScript uses XML (extensible Markup Language) to send and receive information to the server without reloading or refreshing the entire page. The web application only query the server for the information it needs with AJAX and uses JavaScript to updates parts of a web page; this causes web pages to appear faster, more responsive and interactive than before and is an important part of new web applications (W3Schools 2012).

6.2.1.1 HTML5

With the advancements made on mobile devices there is also developed a new version of HTML. The new version "HTML5", aims to work better on mobile devices and thus making it easier for developers to create web applications that support mobile devices. HTML5 received final draft in May 2011 (W3C 2011) and was finished in October 2014 (W3C 2014). HTML5 includes standardized API's that enable browsers to access native features on mobile devices and create a completely new range of possibilities for web applications, making them more competitive with native applications.

One of the new features of HTML5 is an API that will provide device specific information, this will allow developers to tailor a web application to a device (Appendix F.1). Vector graphics and better font support is also added to improve the UI experience (Appendix F.2, Appendix F.3). Threading will allow operations that are computationally intensive to run in the background, which also will help to improve the UI experience (Appendix F.12). Multimedia playback is supported without the use of third party plug-ins and the media capture API's will allow web applications to capture photos, video and audio through hardware on the mobile device (Appendix F.4). HTML5's new API's will also allow access to the device calendar, address book (Appendix F.8) and sensors such as GPS, accelerometer, battery camera etc. (Appendix F.9). Furthermore, HTML forms will allow for specific data type input by showing native date and time controls for selecting values. Forms will also allow for built-in validation (Appendix F.5). User interactions such as touch, hand gestures, swipe etc. will allow users to interact with web application in new ways and web applications will be able to use vibration and sound notifications on the mobile device (Appendix F.6). In addition, will web applications be able to utilize new communication methods such as e-mail, SMS and MMS (Appendix F.10). Database functionality is also added in addition to long term and session's data storage (Appendix F.7). An "availability cache" and the "WC3 widgets" will allow a web application to be installed on the mobile device so it can be used offline (Appendix F.11).

In addition to these and several other API's the goal for the HTML5 standard is to keep evolving to what W3C calls an "Open Web Platform" and "Application Foundations"[20] to make HTML easier and more powerful to use for developers.

6.2.1.2 Mobile browser fragmentation

Since web applications runs in web browsers it is necessary to know what browsers exists on the mobile market. Browser statistic needs to be taken into account when developing a web application and developers should as a minimum create web applications that supports the major browser vendors. Most modern phones and tablets allow users to download and install new browsers with a step-by-step installer or through an application store. However, it is not a viable solution to expect users to download a specific browser to make an application run smoothly and will limit the number of potential users. Figure E.4 shows the development in market shares for various mobile browsers in the period December 2008 and January 2014 based on usage statistics from over 3 million web sites. A detailed review of these statistics can be found in Appendix E. The browser statistics show how the mobile market is developing, what platforms are the most popular and what browsers the majority of users uses. This data will provide additional information of what platforms to aim for when developing a mobile web application. According to the statistics, there were four major browsers in the market in January 2014: Android, iPhone, Opera and UC Browser. As the UC browser and Chrome gained a lot of momentum between January 2013 and January 2014, supporting these browsers will be increasingly important for web applications. Since both Opera and the UC Browser supports multiple operating systems, it is not possible to know

[20] Application Foundations (refer to: http://www.w3.org/appfoundations/)

what operating system lies underneath from these usage statistics. However, this is not relevant either since web applications have to support browsers and not mobile platforms.

In 2012, there were over 200 different interpretations of HTML5 in various browsers. There was also an increase in the number of updates released for the various browsers. Browsers manufacturers are continuously competing with each other in order to have the browser that scores best on HTML and CSS tests. In two years (pre 2012), the Android browser released 21 updates for their web browser (Appcelerator 2012b). This creates fragmentation in the browsers on the market. With potentially 21 different browsers in the market from only one vendor, in just two years, can create variations in the way web applications work and behave. It is also likely to assume that there can be variations in behaviour between platforms, for browsers that work on multiple operating systems. If a web application requires the use of a feature/API it is important to know if the targeted browsers support it, since the support for HTML5 varies between the different browsers. A detailed summary of statistics on the level of HTML5 support in mobile browsers can be found in Appendix F. There is also fragmentation in how much of the HTML5 web standard web browser have implemented. Some features can be partially implemented by some web browsers, but the extent of the implementations varies from one vendor to another. The same type of fragmentation exists on desktop browsers. Historically, it was particularly IE (Internet Explorer) who did not follow the web standards the same way other browsers did. This meant that web developers had to create hacks and tweaks in order to make web pages look the same way for all users. These hacks where mostly made for IE and web sites made especially for IE often did not work at all in other browsers. IE is now following the web standard much closer and this makes it easier for web developers to create web sites that look and behave the same way in all desktop browser.

iOS was the second most sold operating system in 2012 (Table C.1 or Figure 5.1), the second most used mobile browser from mid-2012 (Figure E.4) and had second to third best support for HTML5 (see Table G.1 and Table G.2). Opera was also available for the iOS platform, but HTML5 support was not available in that version (Opera 2012).

Android was in 2012 the most sold operating system (see Table C.1 or Figure 5.1), but their browser was ranked as sixth (version 4.0 of Android) and tenth place (version 2.3 of Android) among the browsers with the best HTML5 support (see Table G.1 and Table G.2). The 2012 Android version distribution in Figure 5.4 shows that version 2.3 – 3.0 accounted for 62% of all Android devices whereas version 4.0 and up only accounted for 1.6%. Since Android have such a large amount of the market, it would be important for developers to know that Android had an issue with a low browser score compared to their competitors. If an application required support for a particular HTML5 feature that was not available in the Android browser, they would not be able to target a very large part of the market. Opera is available for the Android platform with HTML5 support, but is not a standard application on the Android system (Opera 2012). Before releasing a web application, developers should make real life tests on the different browsers and platforms to ensure that the application is compatible.

6.2.1.3 HTML5 tools

The simplest tools available for HTML5 development are Notepad on Windows or TextEdit on a Macintosh. However, these tools will not provide syntax checking or color-coding. To get these features developers need to use tools such as Notepad++[21] (open-source), Adobe Dreamweaver[22] (proprietary), Eclipse[23] (open-source) or Aptana Studio[24], an open-source Eclipse-based IDE tailored for web development (Wargo 2012).

6.2.1.4 Web applications summary

VisionMobile (2013c)[25], stated that only 37% of the available Android applications could have been implemented by using HTML5.

VisionMobile (2013a), stated in a report that HTML5 is the third most popular choice among mobile developers. Since HTML is common and widely used, there are also a larger number of developers available for web applications than that of the native languages. 50% of all mobile developers use HTML5 either as a deployment platform (to create mobile web applications) or as a development platform (to create hybrid applications). HTML5 is becoming an alternative to developing native applications in various categories such as Business and productivity (42% of developers), Enterprise (32%) and media applications (28%). Game development is not a common use for HTML5 (12%), probably because the

---

[21] Notepad++ (refer to: https://notepad-plus-plus.org/

[22] Adobe DreamWeaver (refer to: http://www.adobe.com/no/products/dreamweaver.html)

[23] Eclipse (refer to: https://eclipse.org/)

[24] Aptana Studio (refer to: www.aptana.com)

[25] The survey reviewed 30,339 Android applications within different categories in the Google Play US application store.

level if interaction and performance that is often needed in games is not supported by HTML5 (Appendix F.12).

A drawback with web applications, as reviewed in section 6.2.1.2, is the fragmentation in the way browsers interprets internet standards.

> The main challenge with the web as a platform is fragmentation of both development and distribution. Unlike mobile platforms, in the mobile web there is no clear leader to push forward a single, coherent web development platform with a sufficiently large installed base of devices and a single, compelling distribution channel or app store. VisionMobile (2011a, 11).

There are no common distribution channel for web applications like with native applications, there are no ways to submit a web application to iOS or Android application stores since they only accept native applications. This makes it harder to reach users since they cannot download or search for the application in the integrated application stores on their platform. This in turn also makes monetization harder for the developers (Yeung 2012). There are however some marketplaces that offer web applications. Such as Mozilla Marketplace[26], Amazon Appstore[27] and AppFuel[28]. These marketplaces also support monetization, but each of them with their own API that needs to be implemented.

Two independent surveys listed the cross-development capabilities of HTML5/code portability as the best feature for HTML5. Other reasons that weighted high included low development costs and immediate updates. The dissatisfaction with HTML5 were as mentioned earlier Monetization, distribution and fragmentation, as well as timeliness of new updates/access to latest native API's (faster standardisation), performance, user experience and improved development environment. See Appendix F.13 for further details of these surveys. According to Glenn Stein[29], cross-platform tools for developing hybrid applications are not a quick fix for the fragmentation problem;

> Multi-platforms development is a major challenge. For a solo developer there is so much they need to look at. You can use cross-platform tools like

---

[26] Mozilla Marketplace (refer to: https://marketplace.firefox.com/developers/)

[27] Amazon Appstore (refer to: https://developer.amazon.com/public/solutions/platforms/webapps)

[28] AppFuel (refer to: http://appsfuel.com/about/)

[29] Glenn Stein (refer to: http://www.howwemadeitinafrica.com/app-tackles-south-africas-language-barrier-one-phrase-at-a-time/)

> PhoneGap but it's not that simple, there is a lot of tweaking to be done. Going native can also be hard, it takes a lot of time and patience. HTML5 will help things go in the right direction. -Glenn Stein, Java developer and maker of PhraZapp (VisionMobile 2013a).

With the cross-platform capability of HTML5 for multiple platforms, it is possible to save both time and money. A survey found that "Web applications hold clear cost advantages. The higher costs for native app development stem from OS platform porting and QA testing" (Global Intelligence Alliance 2010, 21).

According to VisionMobile (2013c), 61% of all developers go directly to the built-in browsers when developing mobile applications and not through third party software/compilers such as PhoneGap or Appcelerator. Section 6.2.2 will review hybrid applications more closely and examines the extent they can help to reduce issues caused by mobile fragmentation for mobile application developers.

6.2.2 **Hybrid**

A hybrid application is a cross between a native application and a web application. A hybrid application is a web application that compiles as a native application with the help of a framework. There are several frameworks available for this purpose. This chapter will discuss and review the most used frameworks with more details about how the frameworks work, what features they have and how they aid developers with issues caused by fragmentation.

In a large Cross-platform tools (CPTs) report from VisionMobile (2012a) in February 2012 they found over 100 different CPTs. CPTs allows developers and software companies to target multiple platforms, reuse developer skills, share codebases, synchronise releases and reduce support costs using the same design tool. Developers can create native, hybrid and web applications with several different methods such as JavaScript frameworks, app factories, web-to-native wrappers, runtimes and source code translation. CPTs can create applications for mobile, tablets and TV screens. VisionMobile also states that developers on average use 1.91 CPTs concurrently and that one in four developers use more than three CPTs. The reason for this is the lack of maturity and niche nature of CPTs. This lack of "one-size-fits-all and immaturity in the CPT landscape is what is stalling cross platform tools from shifting the balance of power in the iOS/Android duopoly towards alternative platforms." (VisionMobile 2013a, 44).

Thirty-eight percent of developers that focuses on HTML development uses CPTs. Particularly JavaScript frameworks and web-to-native wrappers. Sixty percent of developers using CPTs have over 5 years' experience with web development.

According to VisionMobile the most popular CPTs are:

1. PhoneGap (used by 34%)
2. Appcelerator (used by 21%)
3. Adobe Air (used by 19%)

The main reason for choosing a CPT is mainly its cross-platform capabilities (68%). Furthermore, 38% of developers choose a CPT because of the development speed and 33% because of their learning curve (VisionMobile 2013a).

According to research by VisionMobile (2012a), the learning curve is one of the top reasons for choosing and not choosing a cross-platform tool. The average time used to learn a new cross-platform tool is significant; the median time is three to four weeks. Thirty-eight percent of developers use one to four weeks and 28% use more than three months to learn a new tool. This is however four to five times faster than it takes to master native Android or iOS development. VisionMobile also argue that since many of the CPTs use HTML5 and web technologies and their ability to compile as a native application, they are effectively shifting HTML5 from a potential application platform to becoming a mainstream development technology instead.

Choosing a framework that will be developed and have continued support for many years is crucial for developers and stakeholders. Much like the mobile platforms there is a lot of competition between CPTs and there is a lot of acquisitions and divestments of these companies. In the case of "Open-Plug" the company was acquired by Alcatel Lucent in 2010 and later discontinued due to lack of revenues. This meant that 22,000 registered developers (with a minor amount actively developing applications) had to discontinue their work and find another development platform for their application.

6.2.2.1 PhoneGap

The first versions of PhoneGap where developed by a company called Nitobi. In October 2011, Adobe Systems Inc. acquired PhoneGap and to ensure that the project remained open source the PhoneGap project was at the same time donated to the Apache

Software Foundation (ASF) under the name Apache Cordova[30]. Adobe's vice president and general manager Danny Winokur said in an interview, "PhoneGap has proven to be an industry-defining app solution for HTML5 developers". He also stated that PhoneGap was a perfect complement to Adobe's broad family of developer solutions, including Adobe Air (Adobe 2011). In addition to donating PhoneGap, Adobe also donated the Flex framework to Apache (VisionMobile 2012a). More information about Adobe Air and Flex in chapter 6.2.2.3.

PhoneGap is a framework that creates a native container around a regular web application consisting of HTML, CSS and JavaScript. The container is a web view that fills the whole screen and allows the user to interact with the web applications inside the view.

> A web view is a native application component that is used to render web content (typically HTML pages) within a native application window or screen. It's essentially a programmatically accessible wrapper around the built-in web browser included with the mobile device (Wargo 2012, 3).

This means that the behaviour and quality of the application will vary based on the quality of the web view and rendering engine on the targeted platform. It is also subject to variations in the implementation of web standards as discussed on page 77 (Whinnery 2012). The web application that runs in the web view is a HTML5 application (Wargo 2012). A drawback by using this method is that the performance of the UI may not feel as smooth to the end-user as a native application due to latency's in processing (Charland and Leroux 2011). The PhoneGap framework is an implementation of Apache Cordova. The container and API plugins are from Cordova. In addition, PhoneGap delivers a set of command-line tools and the PhoneGap services (Wargo 2014).

Server-side code such as PHP/ASP/JSF/Java and .NET will not work inside the PhoneGap container and have to be located on a server. The application can use links or JavaScript to pull down server-side processed content from a web or application server (Wargo 2012). Many application stores will also refuse applications that use native containers to view a remote website. Communication with a server can be achieved, via a XMLHttpRequest (XHR) (Shotts 2014a). "PhoneGap is agnostic of back-end technologies and can work with any application server using standard web protocols" (Trice 2012). Application stores are often very strict about how applications should behave and the

[30] Apache Cordova (refer to: https://cordova.apache.org/)

applications should reflect the conventions used on the target platform (Shotts 2014a). The look and feel of the user interface for applications created with PhoneGap is entirely up to the developer themselves. PhoneGap does not provide any methods or APIs to create native looking applications. To achieve this, developers can use third party frameworks such as jQuery mobile, Dojo Mobile and Sencha Touch (Wargo 2012).

The web application inside the wrapper uses web technology and is inherently cross-platform. The only thing that is native is the wrapper and plugins for each specific platform. The code instantiates the container including core plugins developed by the Apache Cordova team, third-party plugins from the developer community or the developers own custom plugins. Custom plugins are required if the application needs a feature that is not yet supported by the framework. The native code creates a bridge/interface between the web code and the native plugins and platforms. This bridge provides the web code to access native device features with a simple and consistent API and allows web code to perform native device functions (Shotts 2014b).

In version 1.x of PhoneGap, developers were required to compile the application on their own machine. This meant that they needed access to a developer environment for each of the platforms the application would support, the same way a native developer needs as discussed in chapter 6.1. In version 2.x of PhoneGap, Adobe made it possible for developers to build and compile their projects in the cloud with the “PhoneGap build environment”. This method only requires a text editor to edit the web code. Drawbacks with this method is less control of the build process and code, files and other assets/resources are exported to an external environment, making the process less secure. Version 2.x of PhoneGap and the “PhoneGap Developer App” also made it possible to test the application on the devices without compiling the application first. The application connects to the development machine and loads the HTML, CSS and JavaScript code. Any changes in the code on the development machine will automatically be loaded on the device and thus eliminates the build and deployment cycle. The application is pre-compiled and only loads the web code the debugging only has support for the core plugins. The “PhoneGap Developer App” is available on various application stores. The “PhoneGap build environment” and the “PhoneGap Developer App” method requires no local SDK installations and supports Weinre[31] debugging. PhoneGap still supports a local build environment. Many developers prefer this

[31] Weinre (refer to: http://people.apache.org/~pmuellr/weinre/)

method even though it requires a lot of setup and maintenance. Since the entire build process takes place on the local machine developers have more control over the build process and better debugging possibilities. Developers can also deploy to a mobile device or an emulator (Shotts 2014b). However, setting up a local environment requires knowledge about all development environments. Each mobile platform uses different tools, as discussed in chapter 6.1. Each platform also use different configuration files and different folder structures. In addition, the PhoneGap JavaScript libraries used by the native container application are also different for each mobile platform (Wargo 2012).

The PhoneGap framework does not come with its own editor. Instead, the developers use the same editors they use for creating regular web applications (Wargo 2012). PhoneGap is integrated with Dreamweaver (Adobe 2011). See page 80 for more information about web application tools.

PhoneGap allow developers that have already created a web application to deploy their application through one or more of the application stores. It will also give developers access to device features that mobile browser does not support. Developers can access device features with a single interface, the framework translates the interface within the container application into the corresponding native API for the current operating system. PhoneGap allows the compiled application to run on various operating systems. See Table 2 for a list of supported platforms and features. Most applications developed with the PhoneGap framework targets the consumer market. However, more enterprises are starting to use PhoneGap for their employee-facing applications as well (Wargo 2012).

	iPhone / iPhone 3G	iPhone 3GS and newer	Android	Blackberry OS 6.0+	Blackberry 10	Windows Phone 8	Ubuntu	Firefox OS
Accelerometer	✓	✓	✓	✓	✓	✓	✓	✓
Camera	✓	✓	✓	✓	✓	✓	✓	✓
Compass	x	✓	✓	x	✓	✓	✓	✓
Contacts	✓	✓	✓	✓	✓	✓	✓	✓
File	✓	✓	✓	✓	✓	✓	✓	x
Geolocation	✓	✓	✓	✓	✓	✓	✓	✓
Media	✓	✓	✓	x	✓	✓	✓	x
Network	✓	✓	✓	✓	✓	✓	✓	✓
Notification (Alert)	✓	✓	✓	✓	✓	✓	✓	✓
Notification (Sound)	✓	✓	✓	✓	✓	✓	✓	✓
Notification (Vibration)	✓	✓	✓	✓	✓	✓	✓	✓
Storage	✓	✓	✓	✓	✓	✓	✓	✓

✓ - supported feature
X - unsupported feature due to hardware or software restrictions

TABLE 2: PHONEGAP FEATURES (PHONEGAP 2015B).

Table 2 lists the various operating systems and features that the PhoneGap framework supports and the features that are available on each platform. Since PhoneGap is open source, volunteering developers do most of the development of new features and bug fixes. The project can only complete the road map if there are enough resources available. Popular platforms such as iOS and Android get more attention than less popular platforms. Not all features are available on all platforms and developers need to take this into account and adjust the application accordingly (Wargo 2012).

Wargo (2012), state that a concern for companies is that support is available for the software they use and argue that open source products like Open Office would not be as popular if there were no support options available. PhoneGap have commercial support for enterprises ranging from basic to corporate and enterprise.

6.2.2.2 Appcelerator Titanium

Appcelerator, Inc. was founded in 2006 (Lardinois 2014) and introduced Titanium in December 2008 (Sarah Allen 2010). Titanium is an open source, hybrid application framework. Appcelerator launched Titanium when PhoneGap started to gain popularity with mobile developers. Titanium is similar to PhoneGap. PhoneGap uses HTML, CSS and JavaScript, but Titanium only uses JavaScript (Wargo 2012). When writing JavaScript code with PhoneGap or within a web page developers work with DOM objects and other object exposed by the web browser. With Titanium developers work with objects that are exposed through the Titanium API (Anderson 2013).

Appcelerator consists of an SDK with tools, compilers and APIs for building native applications for the target platform. It also has a visual environment called Titanium Developer for managing and building projects. Titanium has no code-editor so developers must choose a suitable editor themselves. Titanium is available on Windows, Macintosh and Linux (Sarah Allen 2010). Appcelerator states that they have over 726,000 developers from 185 countries and over 322 million devices are running applications that have been created by their framework (Appcelerator 2015b).

Operating system	Supported
Android	✔
iOS	✔
Windows Phone	✔
BlackBerry	✔
Web application (browser)	✔

TABLE 3: APPCELERATOR SUPPORTED OPERATING SYSTEMS (SAUNDERS 2015).

The user interface and application logic is created using JavaScript and the applications native container executes the JavaScript code (Wargo 2012). Unlike PhoneGap, Titanium provides a platform-independent API to access native UI components (navigation bars, menus, dialog boxes and alerts). Titanium also proved APIs for native device functionality. See Table 3 for supported features and platforms. During the build process, Titanium compiles the applications JavaScript code into native counterparts for the targeted platforms. The applications created by Titanium is therefore a pure native application with native UI controls (Anderson 2013). Similar to other developer environments Titanium requires the development computer to have the SDK from the targeted platform installed on the development computer for compilation (Sarah Allen 2010). See chapter 6.1 about native environment requirements.

In addition to creating pure native applications, Titanium also has the possibility to create hybrid applications and web applications. The hybrid application from Titanium works much in the same way as PhoneGap with a native web view container and a HTML5/CSS/JavaScript application inside it. The web application contains no native features other than those provided by browsers. There are however not all browsers that are compatible with the generated application and Appcelerator recommends that only mobile browser are use the applications and that desktop browser are only used for previewing and debugging Mobile web applications. In addition, some browser version have limited functionality and requires workaround to work properly (Appcelerator 2015c). Appcelerator states that with their methods developers are able to reuse 60-90 percent of the code base (Appcelerator 2015a).

Similar to PhoneGap, Titanium has a free community edition that can build and distribute applications. The professional and enterprise editions offer additional support and services (Sarah Allen 2010).

Appcelerator does not seem to provide a simple overview of the supported features for each platform, like PhoneGap do as seen in Table 2. To find more information about the currently supported features please refer to the Appcelerator online documentation on http://docs.appcelerator.com/.

6.2.2.3 Adobe Air

Adobe, founded in 1982, is a company based in San Jose, California. Adobe has almost 10,000 employees and annual revenues of over US$4 billion in 2011 (VisionMobile 2012a). In 2007, Adobe started a project code-named "Apollo", later renamed to AIR (Adobe Integrated Runtime). The purpose of the project was to create an environment where developers could create Rich Internet Applications (RIAs) (Brossier 2011).

The Adobe AIR FAQ section state that "Adobe AIR is a cross-operating-system runtime that lets developers combine HTML, JavaScript, Adobe Flash® and Flex technologies, and ActionScript®" (Adobe 2015b). To be able to run applications that use Adobe AIR on a supported platform, it is required to install the AIR runtime on the system. Developers program against the AIR runtime and not the targeted platforms. This eliminates cross-browser testing and ensures consistent functionality and interactions across each platform supported by AIR. Each application on iOS is a stand-alone application and does not require installation of a separate runtime (Adobe 2015a). The AIR runtime embeds within native applications and allow developers to target iOS, Android and BlackBerry tablets (VisionMobile 2012a).

According to VisionMobile, AIR and Flash was originally positioned to target creatives and designers with ActionScript (similar to JavaScript) handling the business logic while Flex (similar to PHP and C++) is a UI framework designed for back-end developers to build enterprise database-connected applications (VisionMobile 2012a). The AIR runtime is independent of the technologies that are used to developer the application (Wagner 2009). Developers can create applications using a combination of Flash, Flex and ActionScript. Or a combination of HTML, JavaScript, CSS and AJAX (Adobe 2015a). Adobe AIR uses WebKit to render HTML, CSS and JavaScript (Adobe 2015b). WebKit is known for supporting the W3C standard (Wagner 2009).

Adobe state that "AIR allows developers to use familiar tools such as Adobe Dreamweaver®, Flash Builder®, Flash Catalyst®, Flash Professional, or any text editor to build their applications and easily deliver a single application installer that works across operating systems." (Adobe 2015b). Developers design the application in Adobe's drag and drop IDEs, such as Flash Builder, Flash Professional and Dreamweaver (VisionMobile 2012a).

Adobe AIR FAQ sections further states that "Adobe Flash Player is a browser plug-in that provides advantages for users and content creators in the browser, including the ability to deliver RIAs in the browser. AIR incorporates technologies originally developed in Flash Player and enables RIAs on the desktop. AIR and Flash Player provide complementary deployment methods for RIAs." (Adobe 2015b).

Initially, AIR developers could write desktop software with ActionScript, HTML or JavaScript for Windows, Macintosh and Linux. Version 2.5 of AIR focused on mobile development and targeted smartphones, tablet computers and netbooks. Versions 2.5 and later supports the Android platform (Brossier 2011). Table 4 shows what platforms Adobe AIR support in 2015. A big drawback with Adobe AIR is that it does not yet support Windows Phone. The third largest mobile platform Q2 2013, see section Appendix C.6 for more information.

Operating system	Supported
Android	✔
iOS	✔
BlackBerry Tablet OS	✔
Web application (browser)	✔
Windows (not Windows Phone)	✔
Mac OS	✔

TABLE 4: ADOBE AIR SUPPORTED OPERATING SYSTEMS (ADOBE 2015B).

Adobe also has a project named the "Open Screen Project" together with technology and industry leaders. The purpose of the project is to create a consistent runtime environment. They seek to take advantage of the possibilities in Adobe Flash and Adobe AIR and state that the project "will address potential technology fragmentation by enabling the runtime technology to be updated seamlessly over the air on mobile devices. The consistent runtime environment will provide optimal performance across a variety of operating systems and devices, and ultimately provide the best experience to consumers." (Adobe 2015c). Brossier (2011), states that the premise of the "Open Screen Project" is to "offer a uniform tool for a fragmented mobile world. One could hope that AIR will be to mobile development what Flash Player is to the Internet. It enables ease of development and distribution of applications across platforms and browsers."

7. Development strategy planning

This chapter will analyse the ELA customer survey and define the operational question OQ2 from the GQM. OQ2 is analysed in this chapter and will provide the final details that are necessary to define a plan for implementing an action. OQ2 will give an overview of the fragmentation found in the ELA customer group and will together with the general findings from OQ1 (chapter 5), OQ3 and OQ4 (chapter 6) define the operational question OQ5, where a development method suitable for the organization Adcom Molde is selected and implemented.

7.1 *The ELA Mobile application*

As discussed in section 1.3, Adcom Molde had discussed developing a mobile application for a long time. With a mobile light-version of ELA, several new features are possible. The main feature is of course that users can access and register information without the need for paper when working out of office. Users could sign risk assessments, register hours, materials and work orders, and not have to go back to the office in the end of the day just to register this information.

7.1.1 User forum

Adcom Molde invites their ELA customers to join in a user forum that is arranged every other year (2007, 2009, 2010, 2012, and 2014). Here customers engage in discussions with Adcom Molde on how to make ELA a better tool and what features should be implemented in the years to come. Newly developed features are previewed, usually along with a couple of mini-courses on using the various features.

FIGURE 7.1: PHOTOS FROM THE ELA USER FORUM 12.06.2014 (ADCOM MOLDE).

The topic of a mobile application was discussed already in the first user forum in 2007. Adcom demonstrated how users could use ELA on a laptop while out of office. Using the 3G network with a VPN connection the responsiveness and functionality would be similar to sitting in the office. Some of the main drawbacks with this was that the mobile network was not available everywhere, it required a continuous connection, limited battery capacity, laptops that can handle the weather elements (often very expensive). A PDA-solution was also discussed and a cooperation with SpeedyCraft[32] had been initiated, however there was some concern from Adcom that their ELA clients was not interested in a PDA-solution. After discussing the topic in the user forum the feedback from the customers where that a PDA is small, and it was hard to imagine that it would be able to fit the needs of the user out of office. Stryn Energi AS stated that every one of their employees had a laptop to register information in ELA in the field and at home, and that a PDA was too small and less functional. At the user forum Adcom Molde made a suggestion that they could create a webpage solution with limited access to ELA as an alternative to a PDA application.

A mobile solution was not a topic in the user forum that was held in 2009 and 2010. The work with ELA was focused around converted to the .NET platform along with defining and releasing new software features.

---

[32] SpeedyCraft (refer to: https://www.devinco.com/)

This master thesis was presented to the participants in the user forum in March 27-28th 2012. Fourteen different companies with a total of 24 representatives attended the user forum in 2012. The participants were presented with the proposal that Adcom would start working on a mobile application and that they needed feedback through a survey (see section 7.3). The purpose of the survey was to get information about what environment the mobile application would be used in, such as user's preferences and their mobile devices. The survey was unfortunately not ready at the time the user forum was held, if it had it is possible it could have received some more responses. The survey along with its findings and how the collected data was used is presented in the next section.

7.2 *Choosing a method for mobile application development*

Before developing applications, it is important to have a good understanding of what the software needs to do. Getting a good overview of the user's needs together with selecting the proper method for developing applications is key in a process to create good software. Thorough discussion and a review of the state of the art had to be made before a mobile development strategy could be selected. The Adcom development department had several meetings where various solutions and technologies where discussed. The results from the ELA customer survey and feedback from customer meetings was also of assistance in this assessment.

As discussed in chapter 6.1 and 6.2 mobile developers today have several methods to develop applications. This chapter will review some of the pros and cons with each method compared to what the different methods can achieve compared to what OC the ELA Mobile application needs to support. Anderson (2013, 2), says each cross-platform tool have pros and cons and that tools such as Titanium are not always appropriate for a mobile application. Anderson argue that "it's important to know why you're using that tool versus something else" and that "the key to making an informed decision about what tool to use is knowing the pros and cons of each particular tool and using that as a guide for which one to use for a particular problem". In order to figure this out, Anderson (2013) says a list of four questions have to be answered before choosing a tool to create an application. These questions are presented below, together with a summary of the considerations and assessments done in relation to the ELA Mobile application with information from the survey.

7.3 *Application specifications and analysis of the customer survey*

The analysis of the mobile market in chapter 5 gives a broad overview of the fragmentation state of the mobile market. These statistics are valid on a global scale, but to answer research question one about the ELA customer group more specific information was needed.

Adcom sent a survey to its ELA customers to retrieve more information about the level of mobile fragmentation and other aspects concerning development of a mobile application. The survey was sent to sixty-six customer e-mail contacts (see Appendix IAppendix JAppendix I). The survey was also broadcasted it to all users of ELA as a news-feed pop-up message within the ELA application. In addition to details about their mobile devices, the survey tried to find out more about user preferences and environmental fragmentation. This section contains an analysis and review of the survey results. The majority of the survey data was collected between 17.08-2012 and 12.09-2012. Two responses were registered 05.07-2013 and one on 04.11-2013.

Note: The percentages are round upwards to one decimal and some questions are multiple choice, so the total can add up to be over 100%. The survey received 39 responses. 92% from men and 8% from women. The average age of the respondents was 42.1 years where the youngest was 21 years old and the oldest was 62 years old.

7.3.1 **Company related information**

First, the respondents were asked to give some information about their company. In total, 17 different companies participated in the survey and are listed below (some companies had more than one response).

1. Røros Elektrisitetsverk AS
2. Selbu Energiverk AS
3. Rauma Energi AS
4. Skjåk Energi KF
5. Stranda Energi Nett
6. Fosenkraft AS
7. VOKKS Nett
8. Kvikne-Rennebu Kraftlag
9. Orkdal Energi AS
10. Stryn Energi AS

11. Årdal Energi KF
12. Oppdal Everk AS
13. Svorka Energi
14. Lærdal Energi AS
15. Notodden Energi AS
16. Andøy Energi AS
17. Sykkylven Energi

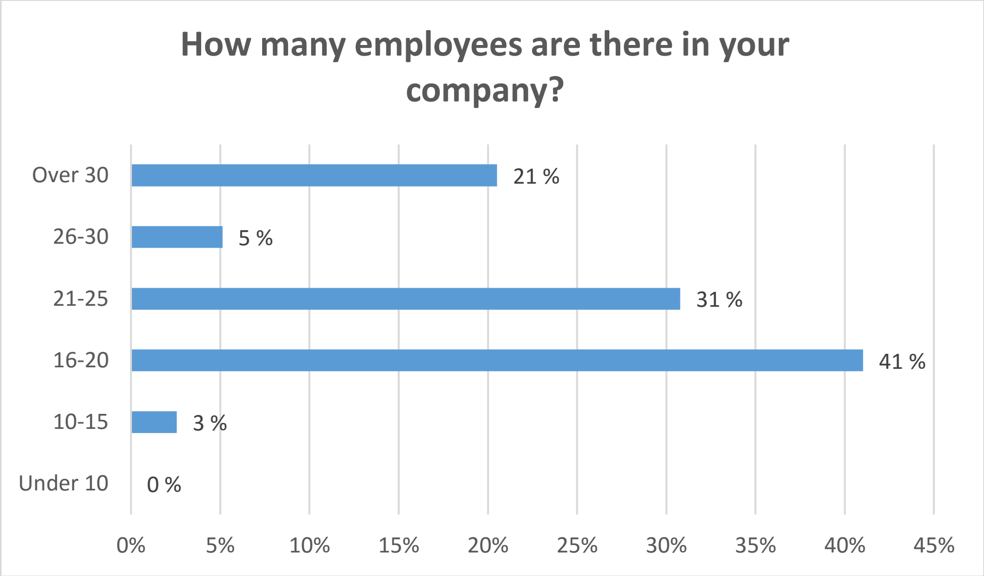


FIGURE 7.2: SURVEY, NUMBER OF EMPLOYEES IN THE COMPANY.

Of the companies that responded, everyone except one (10-15) have a staff of at least 16-20 people. Eight have a staff of over 30 people.

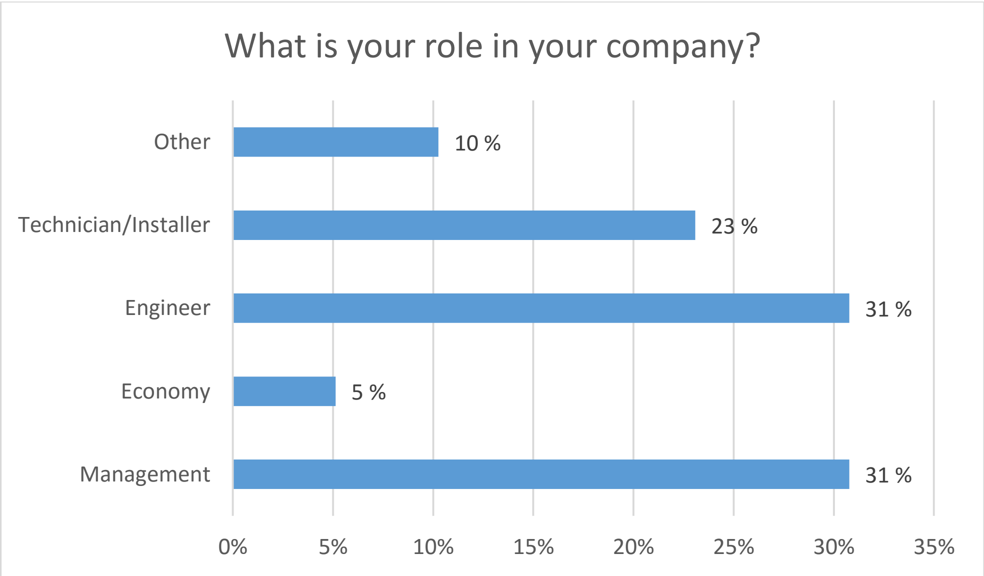


FIGURE 7.3: SURVEY, COMPANY ROLE.

The survey asked what position the respondent has in the company. The majority of respondents came from persons in a management positions. Twelve with a pure management position and twelve with engineering positions. Nine responses came from technicians/installers that mostly work out in the field. Only two of the respondents work with economy (bookkeeping, salary management etc.). Four respondents have other (unknown) positions in the company. It is positive that 54% of the responses came from engineers and technicians/installers, which are the users that make the most use of work orders in ELA. The overall diversity in respondents makes up for a representative group of potential application users.

7.3.2 Question 1: What is the functionality of this app?

Developers need to know what functions they want to include in their application and if the application require access to hardware or other built-in platform functions. Not every development method will be able to access these types of functions and will be an important factor for choosing the best method. If the method does not support the required features of an application developers have to choose another alternative. It is also important to note that even if a specific platform have support for a feature it is not given that every version of the platform have the same level of support.

7.3.2.1 ELA Mobile

Hardware features on phones can be quite a fragmented, something that the ELA customer survey confirms. It is evident from Figure 7.4 that there are multiple variations in the hardware of phones that the respondents use. Looking at the hypotheses from RQ1 in section 3.1 it is evident that H0 has been refuted. Since the survey have found this many variations in the features of the mobile devices it is not possible to state that the ELA customer group does not contains fragmentation.

The types of input on the various phones varies the most. 23.1% of the phones uses a T9 keyboard while 38.5% has a full keyboard available, but at the same time, 92.3% has a touch screen. This means that some phones have both a hardware keyboard and a touch screen.

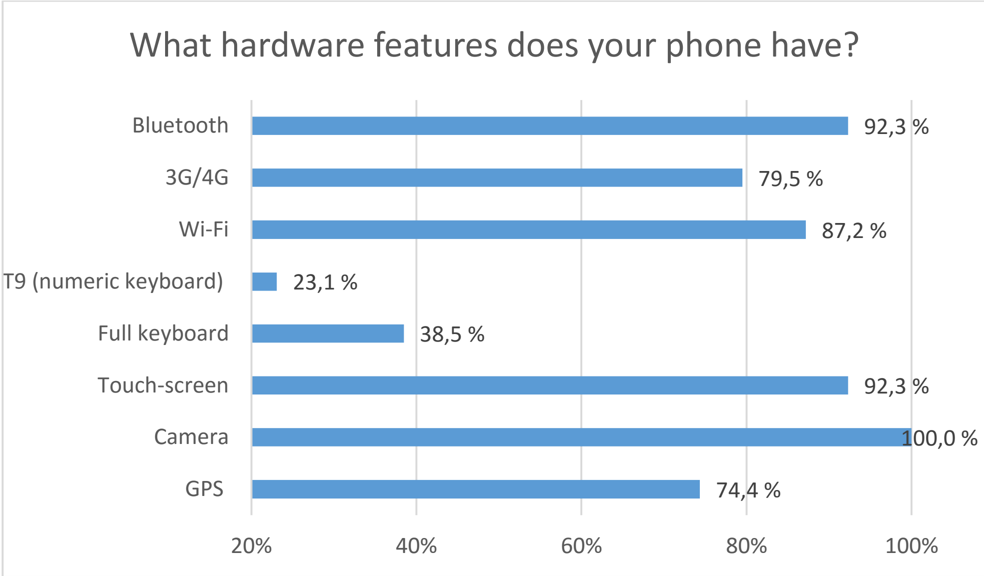


FIGURE 7.4: SURVEY, PHONE HARDWARE FEATURES.

All respondents have a camera available on their phone and over 74% have GPS and some type of communication device (Bluetooth, 3G/4G or Wi-Fi). The most important thing to take away from this is that 7.7% of the phones does not have a touch screen. These users cannot easily navigate a mobile application the same way users with a touch screen can.

The survey question in Figure 7.5 tried to find out what aspects, other than its features, of a mobile application user values the most (in a scale from one to five). The most important aspect is user friendliness with an average of 4.79 out of 5. Availability was a bit behind with 0.38 points and a total of 4.41, followed by a photo function with 4.36 points. Security rates as middle tier with 4.21, followed by a responsive interface with 4.10 points. Geographic location (3.74) and offline access (3.77) was almost rated the same as the least important functions in a mobile application.

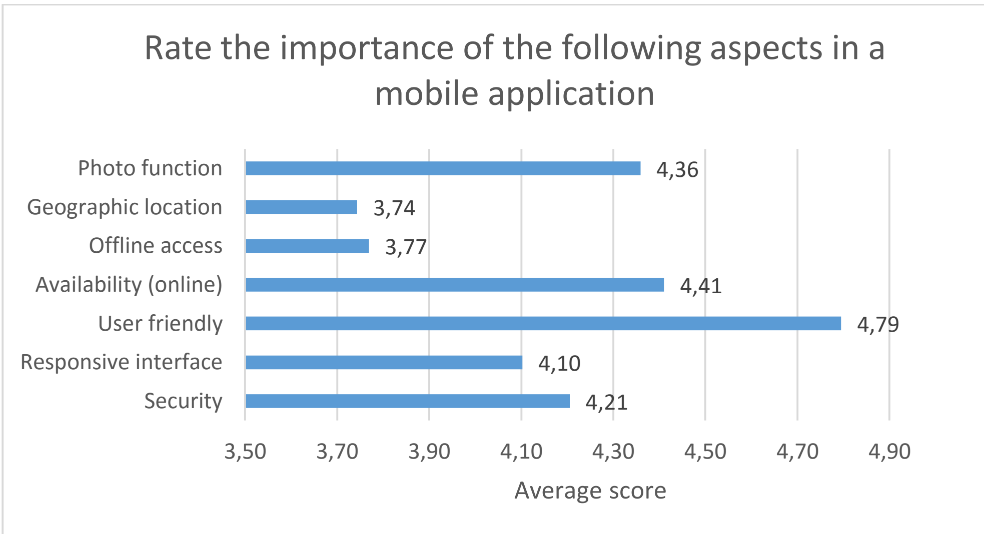


FIGURE 7.5: SURVEY, APPLICATION FEATURES RATING.

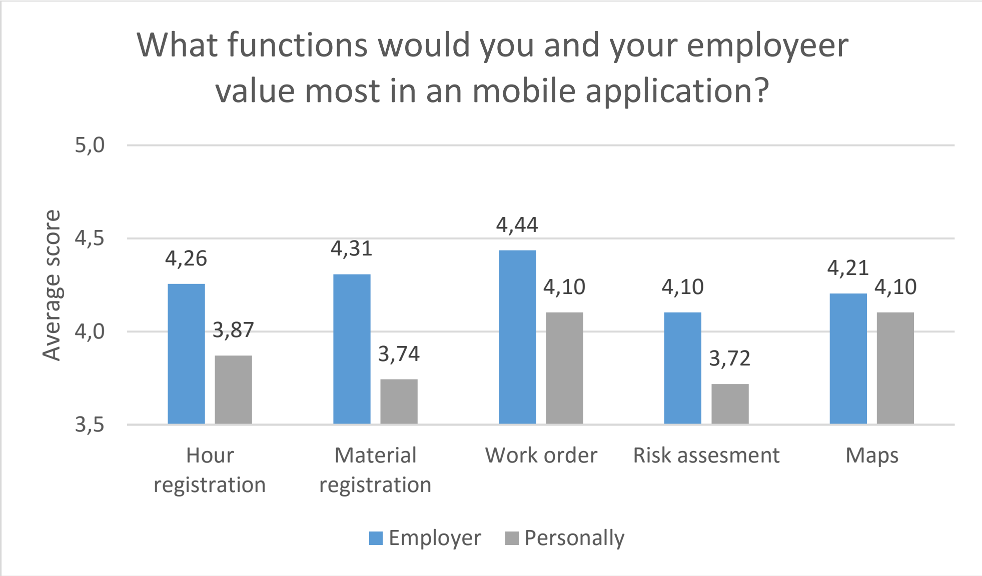


FIGURE 7.6: SURVEY, FUNCTION VALUES.

In Figure 7.6, respondents were asked to rank (in a scale from one to five) what features they personally would like in an application, and what kind of features they think their employer would prefer. Personally, the respondents equally prefers work order (4.10) and maps (4.10) in an application. They also say that their employee would prefer work order (4.44), but with material registration (4.31) as the second most favoured feature. Interestingly to notice is that in all features the score is lower in the personal ranking than in the employee ranking. Perhaps this can suggest that the application is more valued with the employer than with the employees. The least favoured feature, both personally and for employer, is the risk assessment with respectively 3.72 and 4.1.

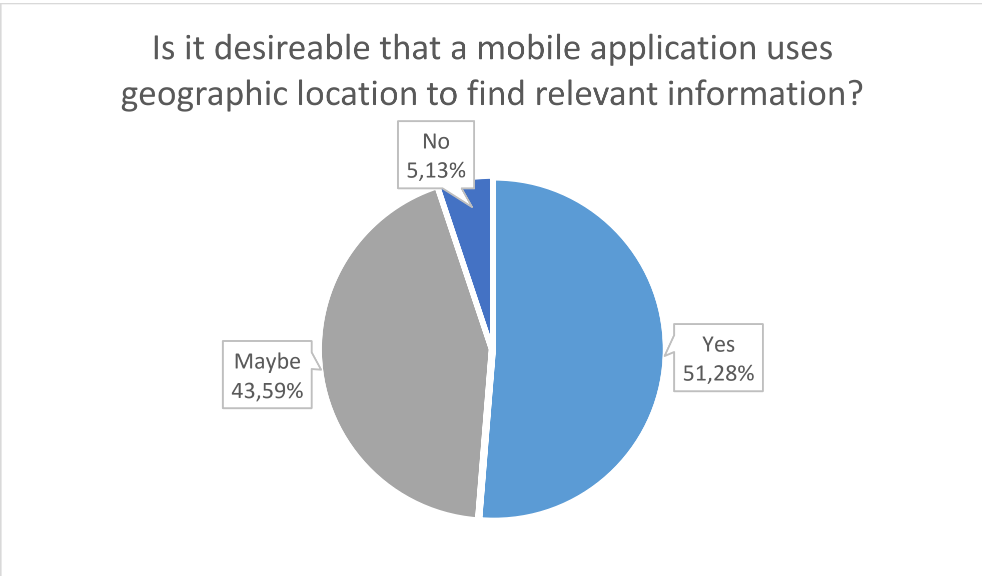


FIGURE 7.7: SURVEY, GEOGRAPHIC LOCATION.

Figure 7.7 asked if was desirable for the application to use geographic location to display relevant information. The survey respondents were also asked to rank geographic location compared to other application aspects in Figure 7.5, where it was ranked lowest in the list. Even if geographic location was ranked lowest it is still a feature that would be desirable for the majority of respondents (51.28%). There are still 43.59% that said they might be interested in this feature, something that could indicate that there is too little information available on what this feature will mean for users. For instance, they could feel that it is important that the geographic information is used to find information such as a list of work orders that needs attention in the vicinity. And not gather information about the user's whereabouts for filling out their hour forms etc. Only 5.13% said that this is not a feature they would like in a mobile application.

Question 1, summary:

The initial goal of the ELA Mobile application would be collecting data for the ELA application. The first version of the mobile application should be able to display work order information, register material and hours, in that order. This is also desirable for users as seen in Figure 7.6 where these function are rated the highest with the exception that users personally would prefer maps. Maps are however something that requires more work compared to features and the framework that already existed in ELA. However, knowing that it is desirable makes it a feature that will have to be reconsidered in later versions.

In addition to this, user should be able to take photos through the camera API and upload them to the document archive on a work order.

Later versions of the application were thought to include access to the location/GPS API so the application could display relevant information, such as unfinished work orders in proximity of the user. The majority of respondents (51.28%) in the ELA survey stated that it would be desirable if the application could find relevant information with GPS, while 43.59% stated that it might be desirable with a feature like this. Only 5.13% said no. This type of functionality also requires information on the location on each work order. For instance can engineers manually pinpoint a location on a map or register information from other systems about Network Information System (NIS) components or customers on the work order with geolocation information or a searchable address.

To find out more about the connectivity of the mobile devices several key questions was asked in the survey about, mobile carriers, connection types and mobile network connectivity at home and at work.

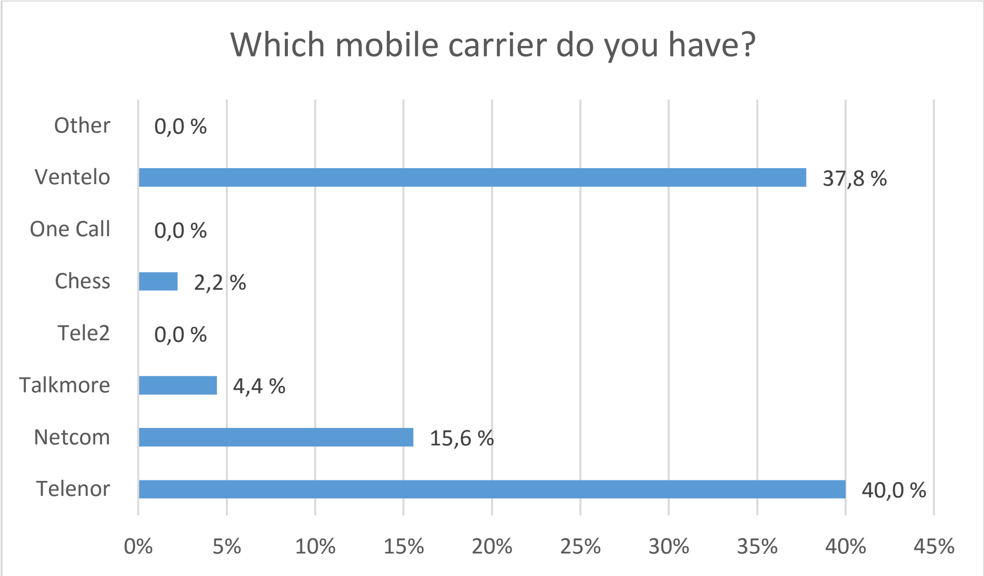


FIGURE 7.8: SURVEY, MOBILE CARRIERS.

** Respondents may select more than one checkbox, so percentages may add up to more than 100%.*

There are three large mobile carriers in use with the respondents. The majority being Telenor with eighteen (40%) followed by Ventelo (now Phonereo[33]) with seventeen (37.8%) and Netcom on third with seven (15.6%). Six (15.4% of the respondents) answered that they have two different mobile carriers, all of these have a both a work phone and a private phone. Only one (2.6% of the respondents) that has both a work phone and a private phone uses the same mobile carrier both at work and at private (Telenor). 4.4% uses Talkmore and 2.2% uses Chess. No one uses One Call, Tele2 or other mobile carriers.

[33] Ventelo/Phonereo (refer to: http://www.ventelo.no/index.html)

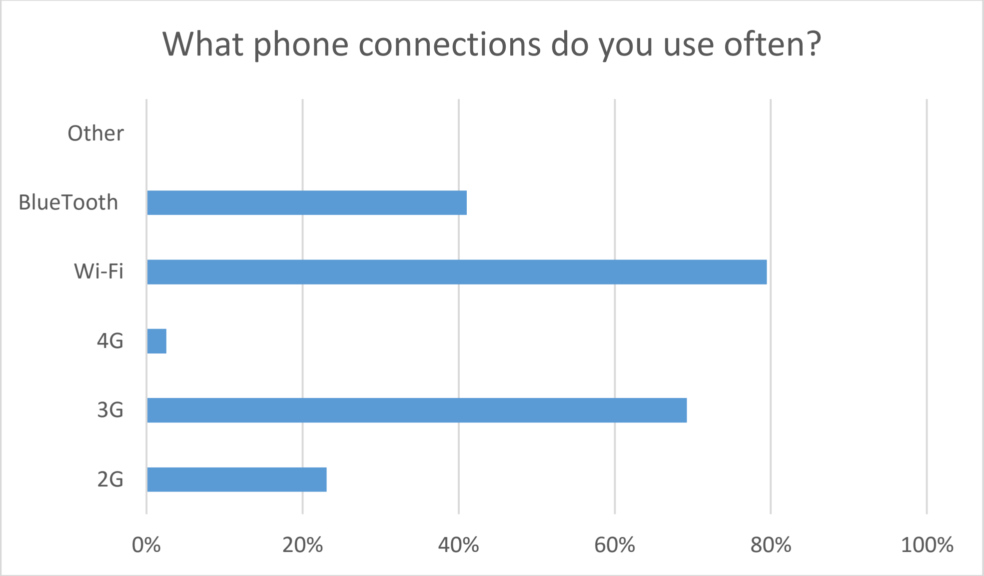


FIGURE 7.9: SURVEY, PHONE CONNECTIONS.

The survey did not specify different technologies within the 2/3/4G terms so that 2G will include everything below 3G (such as GSM, GPRS and EDGE) the same applies to 3G and 4G.

The most used connection is Wi-Fi where 79.5% uses it frequently, followed by 3G with 69.5%. Many workplaces provide Wi-Fi for their employees/customers and many have Wi-Fi installed in their home. Most often there are no data transfer limits or restrictions on Wi-Fi. The only cost of Wi-Fi is the monthly fee to the ISP, but it is cheaper and (usually) faster than the mobile connections such as 2/3/4G where subscribers normally pay for the data amount they transfer. Wi-Fi can also give access to resources inside the local network (file archives etc.), 3G normally does not give users access to this unless the local administrator sets it up in the router and/or firewall. However, mobile connection gives access to other types of resources, such as receiving MMS messages. The mobile connections also have a much larger range than Wi-Fi, but also require more battery power.

Twenty three point one percent often use a 2G connection on their phone. Mobile phones will automatically select the best available connection, so when 3G access is not available the phone will switch to 2G. 2G have a higher availability as seen on Figure 7.10, but has a smaller data transfer rate.

Forty-one percent uses Bluetooth frequently. This type of connection connects different devices to the mobile phone, such as hands free and sound systems or transferring data to other mobile phones etc.

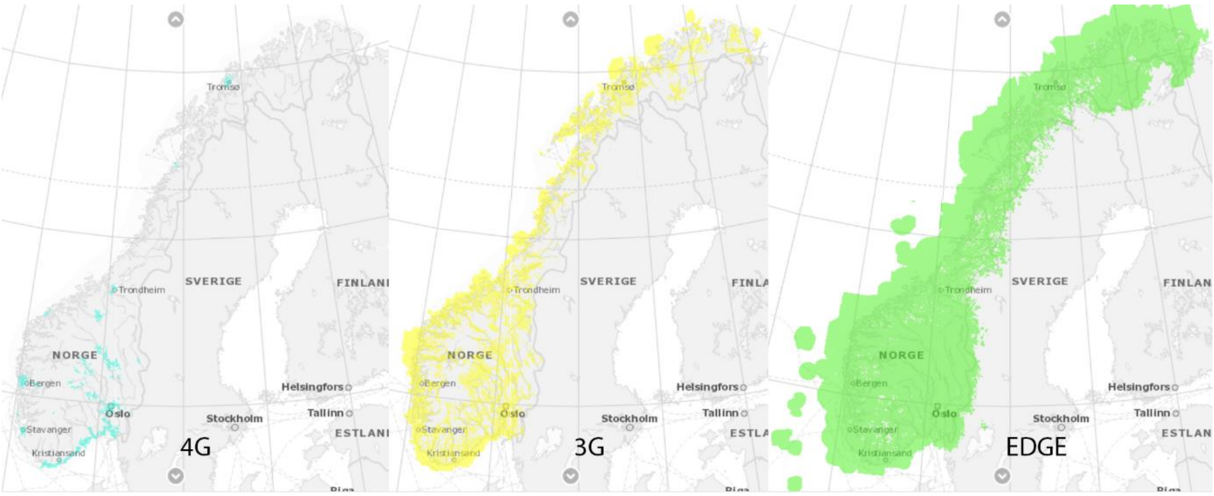


FIGURE 7.10: MOBILE COVERAGE IN NORWAY 28.11-2013 (TELENOR 2013).

As seen in Figure 7.10, the Telenor has good mobile network coverage in entire Norway. The majority of survey respondents have Telenor as their mobile carrier (see Figure 7.8). In 2014 and 2015 there have been a very rapid expansion of 4G coverage from both Netcom and Telenor. The goal for Netcom is a coverage of 96% of the Norwegian population in 2016 (Valmot 2015).

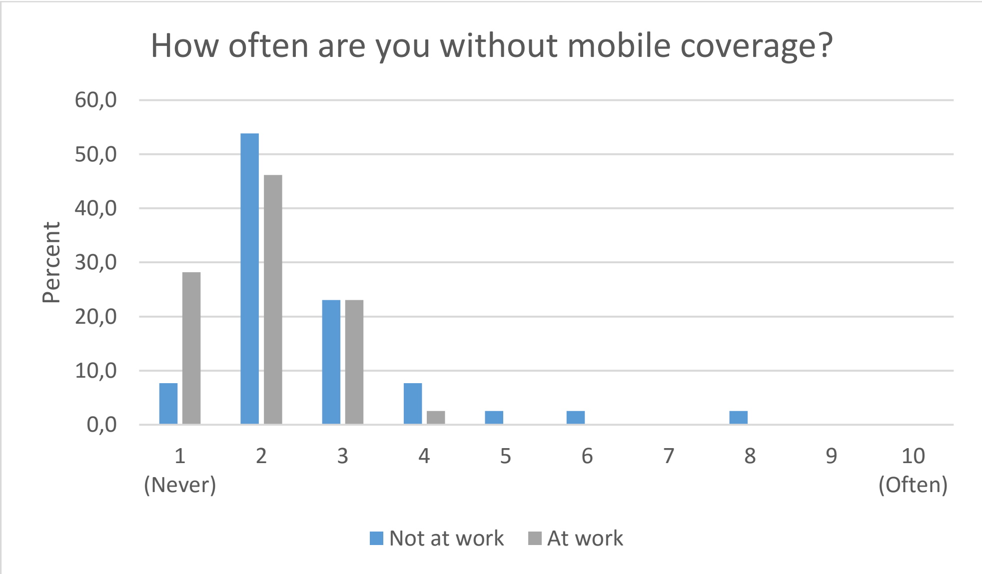


FIGURE 7.11: SURVEY, MOBILE COVERAGE.

As seen in Figure 7.11, the respondents of this survey are more often without mobile coverage when they are at home than when they are at work. This could be because some are stationary to a specific area when they are at work, while they are often at various locations when they are not at work. The majority of time, no users are rarely without mobile coverage. Since mobile network access is very good in most parts of Norway (see Figure 7.10) and based on information about mobile coverage from Figure 7.11 the ELA Mobile application was not required to work offline. In addition, some of the reasons is also the level of complexity it adds. Offline access was also rated second last of importance for users as seen in Figure 7.5, while online access and availability was rated second. If only online use of the application would become a problem for users, the topic of offline functionality would have to be reviewed in later version.

7.3.3 Question 2: Who is going to be using the app?

If developers target a small section of the market, for instance users with an iPhone 6, then developers know exactly what types of hardware and features the phone model have and it is easier to support. Usually this is not the case and developers have to support a larger number of user and ergo different devices.

7.3.3.1 ELA Mobile

With ELA Mobile the potential user base is every user of the ELA application. The ELA survey tried to find out more about these users and their preferences.

One part of the survey tried to find out more about what features of the phones are used. This was to find out more about the level of device usage the users utilize and if they are experienced user.

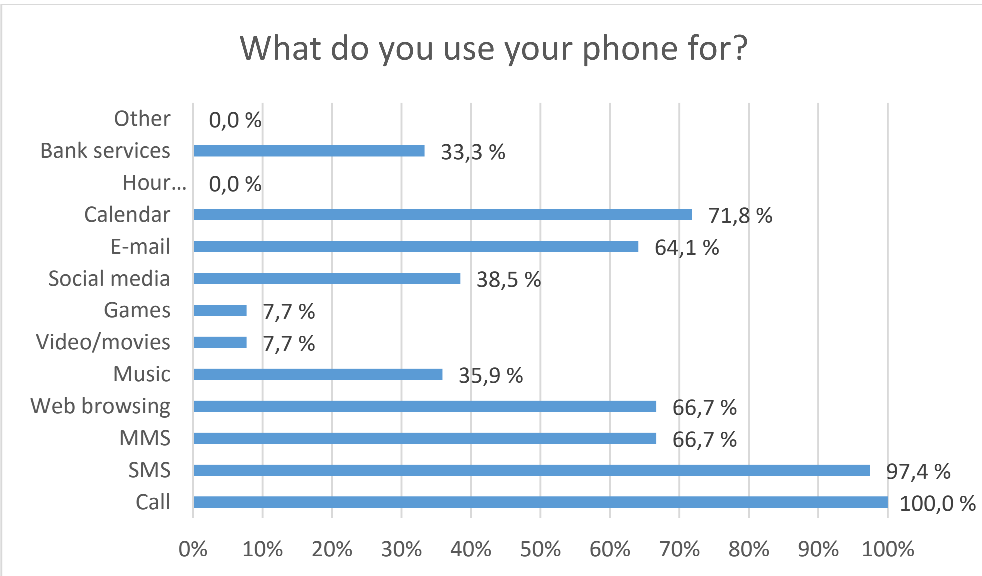


FIGURE 7.12: SURVEY, PHONE USAGE.

The two most basic functions in mobile phones are also the most used. Every one of the respondents use their phone to make calls and 97.4% uses their phone to send SMS messages. Followed by the calendar at 71.8% (a smartphone function). MMS and web browsing is equally used with 66.7%. E-mail is close to web browsing and MMS with 64.1%. Social media, music and bank services are almost the same with 38.5%, 35.9% and 33.3% respectively. Games and videos/movies are the least used with 7.7% except for hour registration, which no one uses. No respondents stated that they use their mobile device for other things.

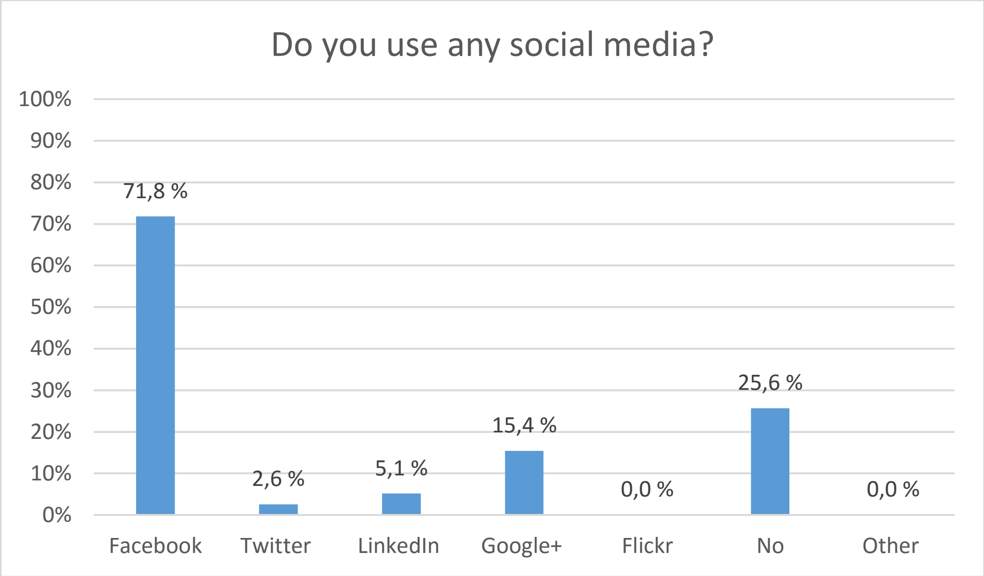


FIGURE 7.13: SURVEY, SOCIAL MEDIA.

The majority of respondents uses social media (74.4%) and some uses more than one type of social media. The most popular by far is Facebook with 71.8% followed by Google+ at 15.4%. Only 5.1% uses LinkedIn and 2.6% uses Twitter. One in four does not use social media (25.6%).

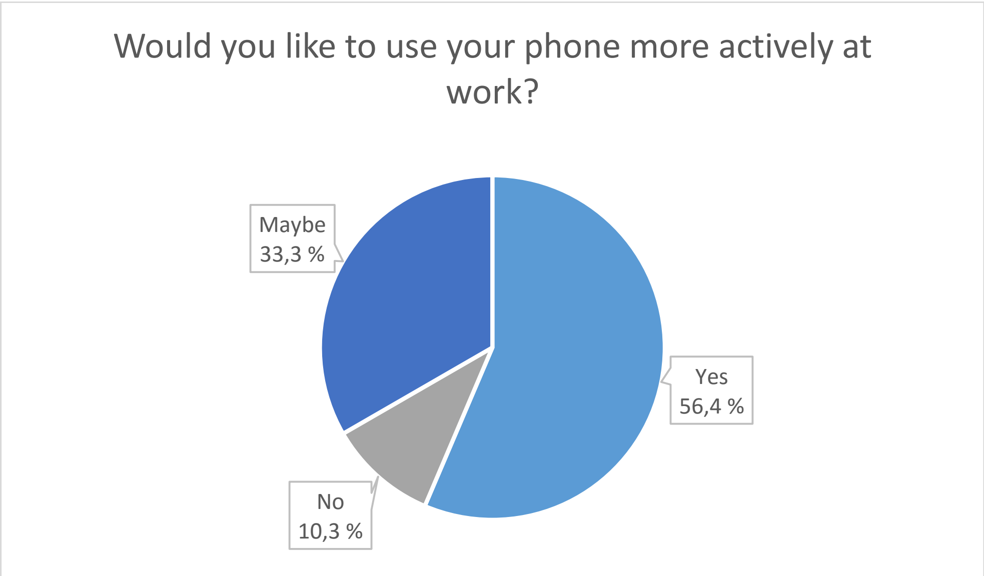


FIGURE 7.14: SURVEY, PHONE USAGE AT WORK.

As seen in Figure 7.14, only 10.3% of the survey participant's do not wish to use their mobile phone more while they are at work. However, 56.4% of them wish to use their phone more actively in their workday while 33.3% answered that they maybe want to use their phone more. The group that answered "maybe" are perhaps pending on what type of applications they could use that would make it worthwhile to use their phone more actively.

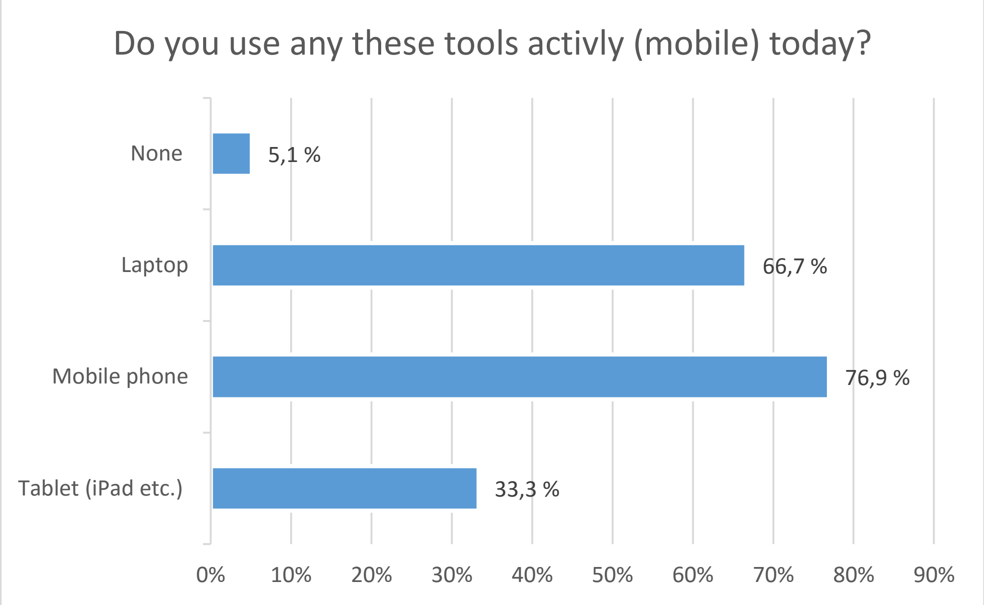


FIGURE 7.15: SURVEY, TOOLS THAT ARE CURRENTLY USED IN THE ORGANIZATION.

The question in Figure 7.15, tries to learn more about what tools the respondents already use in the field. The survey found that the mobile phone is a much-used tool with 76.9% using it. The survey does not take into account what the functions of the tool is so some respondents could just be using it to make calls and send SMS messages. Laptops are also widely used, with 66.7% using them as a mobile tool. As stated previously some customers of ELA use remote desktop or VPN to access software and databases on the intranet. 33.3% uses tablets in the field something that suggest that they already have applications and tools, perhaps even business applications that work on tablets specific to their work area.

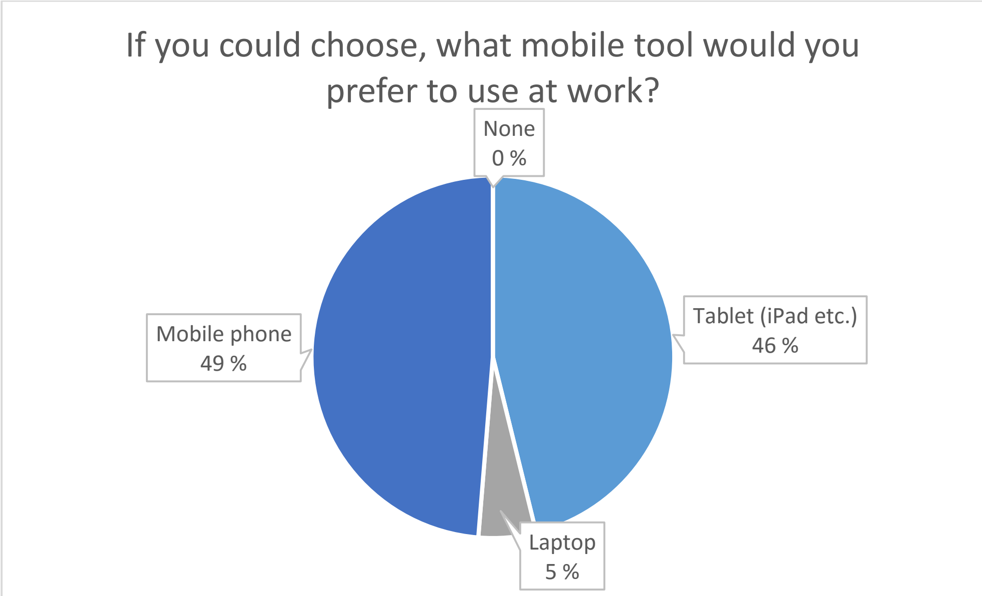


FIGURE 7.16: SURVEY, MOBILE TOOL PREFERENCE.

When it comes to the respondent's choice of mobile tool in Figure 7.16, 49% of the respondents prefer the mobile phone, followed by tablets at 46%. Very few, only 5%, prefer laptop even though 66.7% uses them actively as seen in Figure 7.15.

Question 2, summary:

The potential users of ELA Mobile are user that use their mobile devices actively in their daily job. Only 5% does not use mobile devices in their daily job. The majority of users would prefer to use their device more actively at work, either a mobile phone or a tablet. Three of four uses social media, and the majority uses their phone for more than making phone calls and send SMS. An interesting thing to notice in Figure 7.12 is that 66.7% of the users use their device for web browsing and are very familiar with this type of usage on their mobile device. It is probably safe to assume that the majority are intermediate user that know how to utilize their device and to access both native applications as well as web applications.

7.3.4 Question 3: How is the app going to be distributed?

In many cases the purpose of developing an application is to offer customers a good product, but at the same time make money on it. As discussed in section 5.3 the monetization and thus the distribution of an app is an important choice. This question also relates to question one and two, what are the functionality of the application and who are the users? Developing a non-native application will not give developers access to the applications stores and if the developer does not know the target users yet, these applications stores will give them access to potential users, but at the same time takes a percentage of the earnings and means giving up some control over the app distribution.

7.3.4.1 ELA Mobile

There are several possibilities to distribute mobile applications. In Figure 7.17 the survey asks if the respondents have experience with application stores.

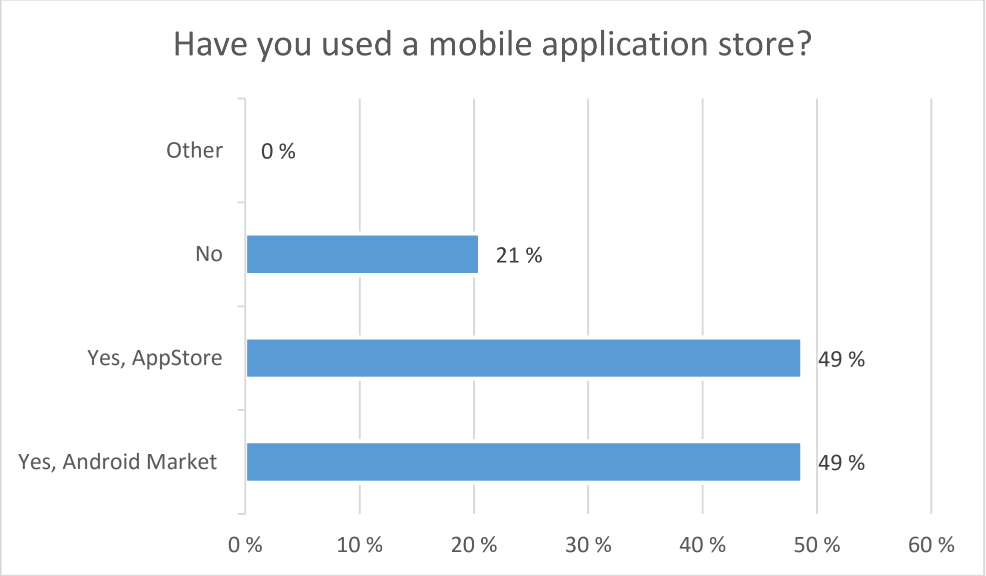


FIGURE 7.17: SURVEY, MOBILE APPLICATION STORE.

The application stores are features of the new smartphones and 79% of the respondents have tried an application store. 17.9% have tried one or more of the application stores, while 21% have never tried it. The only application stores mentioned in the survey are Google's Android Market and Apple's AppStore, however no-one have answered to have used any other types of stores. Equally many have tried the two different application stores.

Question 3, summary:

Usually native applications installs via the application store that is available on the mobile device. It is however possible to install applications without the use of an application store. Android allows installation of applications from a single install package. These applications does not need pre-approval by the application store (Google 2014). iOS also allows users to install applications outside the application store. However, there are still some restrictions on the distribution of the applications. One option is to subscribe to a B2B (business-to-business) distribution through iTunes, there is a yearly fee of $299 USD for this service. Another option is the Ad hoc distribution. With this distribution type, developers can share the application through email or their own server. However, this distribution type is limited to 100 different iOS devices (Apple 2014).

If ELA Mobile was a native or a hybrid application, it would distributed by offering a free version of the application in the application stores. Once the application was installed, the users would be required to log in on the application in order to use it. One application in the application store that uses this method is Plex[34]. The same method would be used if the application was a web application, with the exception that the application would be not be downloaded from an application store. The web application would have to be hosted on a server, either on a server hosted by the customer itself or on a server provided by Adcom Molde. Common for both development methods is that in order to use ELA Mobile, a yearly license would be required. This license would be an extension of the yearly license customers have to pay for using ELA. Since the targeted users are the same as the ELA users, the easiest licensing method would be to use the same licensing method with no limitation in the number of users.

The user account that the users would log in with would have to be connected to a system that contains the login information. More information about this system can be found in section 8.2.5.

7.3.5 Question 4: How many platforms will the app need to run on?

How many platforms the application have to support, is tightly connected to who the users of the applications are from question 2. If developers aim to support a set of users that uses multiple platforms, the developers have to evaluate the impact on their users and the consequences of only supporting one platform. On the other hand, if developers want to create an application for the general market, where everyone are a potential user this might not be an issue. Except for the potential revenue losses of only supporting one platform, after considering the development costs.

7.3.5.1 ELA Mobile

The survey also tried to get information about what mobile devices the ELA customers have in use today.

[34] Plex (refer to: https://play.google.com/store/apps/details?id=com.plexapp.android)

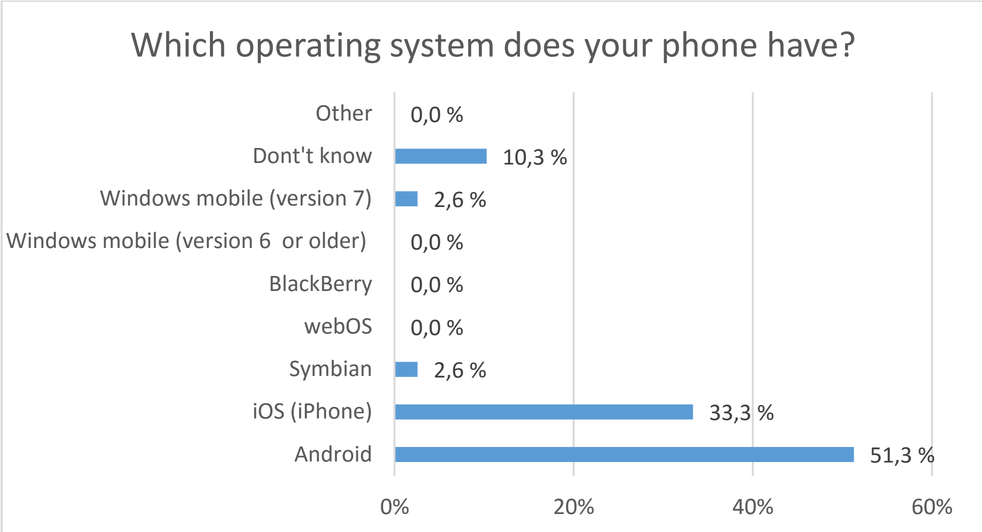


FIGURE 7.18: SURVEY, OPERATING SYSTEM.

The distribution of mobile operating systems in Figure 7.18 shows that the preferred platform with the survey respondents are Android (51.3%) and iOS (33.3%) is the. One respondent (2.6%) has a Symbian phone and one (2.6%) has a windows phone 7. However, since there are four (10.3%) of the users that does not know what operating system they have so it is not possible to know what operating system is the third largest. The most important thing to notice here is the fact that the ELA customer group uses multiple operating systems. This means that the targeted user group contain software fragmentation and that Adcom Molde need to support more than one operating system.

The H0 hypothesis from RQ1 in section 3.1 has already been refuted, however it is also further confirmed by Figure 7.18. Since we have found that there are more than one operating system, it is also evident that the ELA customer group contains software fragmentation. These findings, together with the findings of hardware fragmentation in Figure 7.4 confirms the HA1 hypothesis.

To some extent it is possible to compare the software fragmentation seen in Figure 7.18 with the statistical data from Gartner in Table C.1 to see if there are similarities with the global market and within the ELA customer group. However, it is important to keep in mind that the survey of ELA customer shows devices that are in use while the statistics from Gartner shows the share of operating systems that have been sold on the global market, not how many of them are actually in use. There are a couple of things to note about the ELA customers. Firstly, there are some older phones in use. This is also evident in Figure 7.20 and Figure 7.4. Secondly, the share of various platforms within the ELA customer group does to some extent correspond with the global statistics from Table C.1. This means that the HA2 hypothesis from RQ1 is to some extent confirmed as well. The largest platform is Android followed by iOS and then (possibly) Microsoft/Windows Phone, taking into consideration that the Symbian platform is an older platform and will be discontinued (see section 5.2.3 for more information). However, there seem to be a larger share of iOS within the ELA customers, compared to the global statistics.

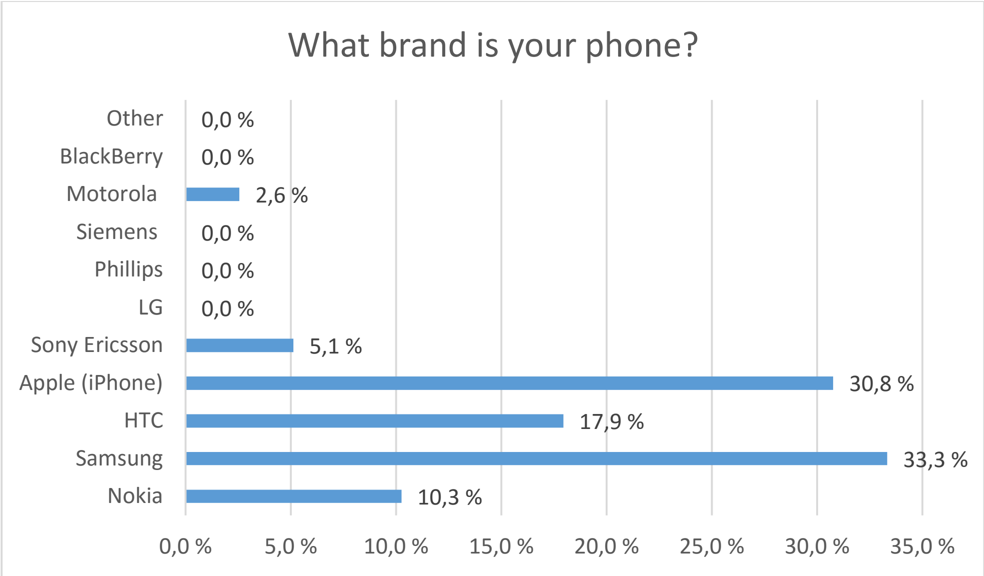


FIGURE 7.19: SURVEY, PHONE BRAND.

Samsung (33.3%) and Apple (30.8%) are the two most used brands by the respondents. From the list of phone versions the respondents stated to have, it is visible that the Samsung galaxy series is a very popular phone. HTC is in third with 17.9% and Nokia in fourth with 10.3%. Sony Ericsson and Motorola is in last with 5.1% and 2.6%, respectively.

As seen in Table H.1, there are great variations in the hardware and software of phones listed by the survey respondents. Every operating system are represented with multiple versions, such as Android (v2.1 to v4.1.2), iOS (v4 to v9.1), Symbian (^3 and v9.3) and Windows Phone (v8 to v8.1). The screen sizes varies from 2.2 inches with 240x320 pixels to 4.8 inches with 720x1280 pixels. Most phones have sensors such as GPS, accelerometer, gyro and compass, but there are also a couple that have no sensors. The phone release dates vary from July 2009 to February 2013. The phone age variations are also seen in Figure 7.20.

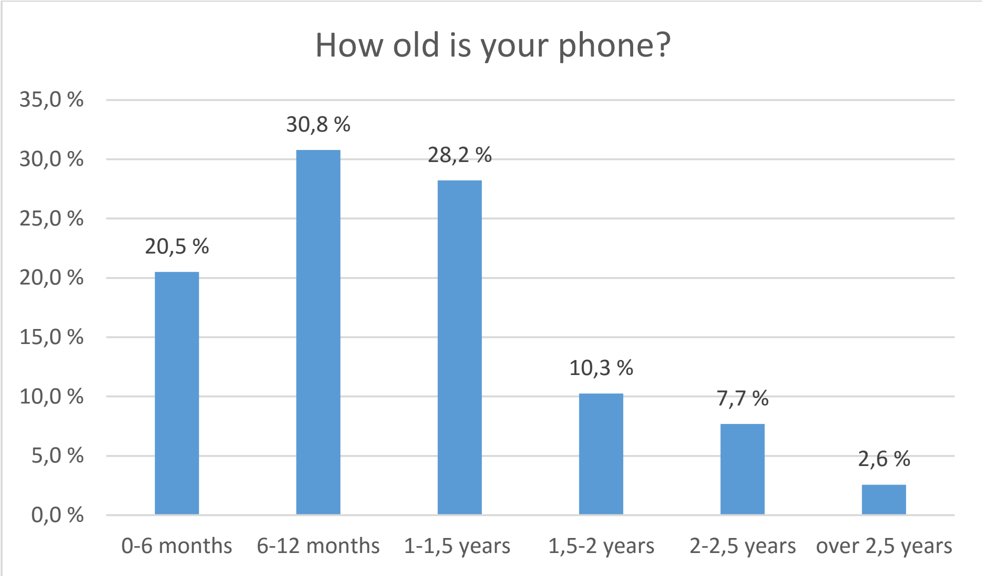


FIGURE 7.20: SURVEY, PHONE AGE.

As stated previously, the lifetime expectancy and warranty of a phone should be three to four years (Elektronikkbransjen 2007). However in this survey, only 2.6% has a phone that is older than 2 ½ years. While 79.5% of respondents has a phone that is less than 1 ½ years old. Over half of the respondents (51.3%) has a phone that is less than 12 months old.

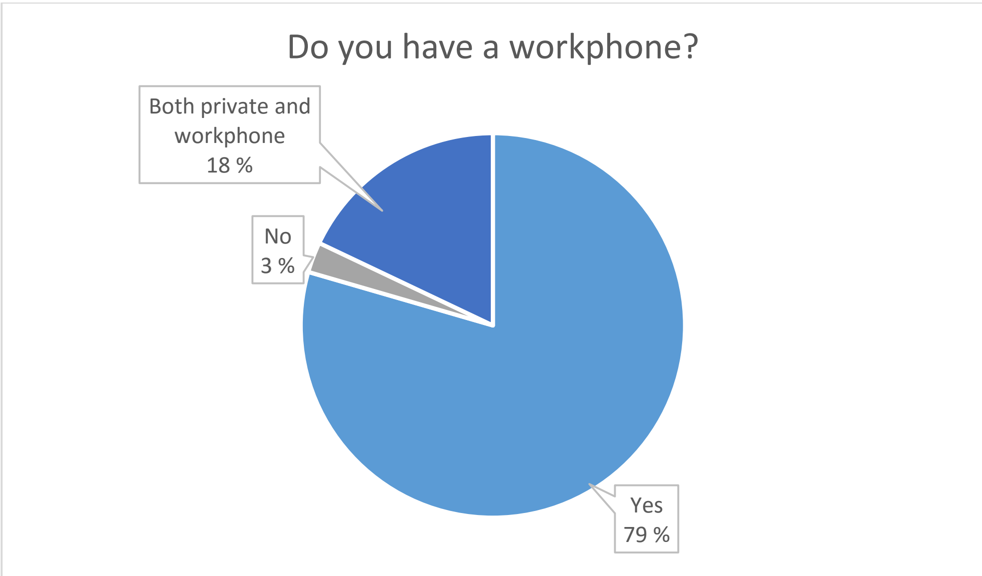


FIGURE 7.21: SURVEY, WORK PHONES.

The majority of respondents (79%) said they have a phone provided by the company where they work. Eighteen percent have both a work phone and a phone for private use. Therefore, a total of 97% have a phone provided by the company. Only 3% (one respondent) does not have a work phone (the survey did not specifically ask if this group had a private phone, but the respondent listed what phone he had, ergo the respondent only had a private phone). The respondents that has two phones are not able to define what phone they are answering for in this survey, but since this survey mainly focused around work related topics, it is natural to assume that they answer the survey questions with their work phone in mind. Further discussion on this topic can be found in section 5.5, “Bring your own device (BYOD)”.

These models also seem to be used in some of the ELA customer companies, as seen in the survey presented in section 7.3. The survey shows that some employees have multiple phones, possibly to differentiate between personal and corporate use, while others only have one phone. It seems like the majority of companies allow users to use one device for both private and business applications, and possibly, a mobile device that they prefer to use and have chosen themselves.

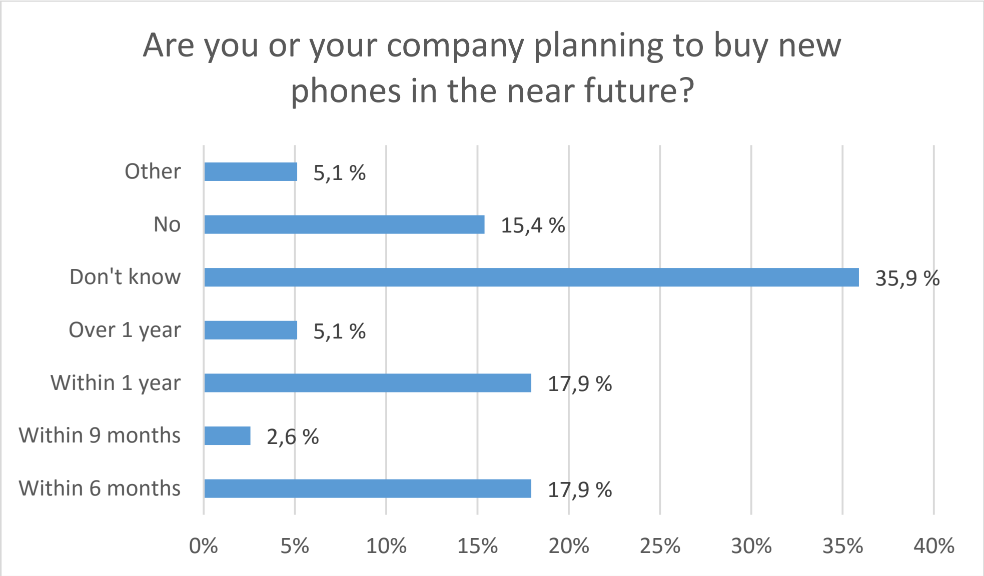


FIGURE 7.22: SURVEY, FUTURE COMPANY INVESTMENTS.

Most respondents know that they are going to buy a new phone within one year (38.5%) and 5.1% know they will buy a phone after one year. 15.4% answered that they do not have any plans of buying a new phone, while 35.9% do not know when they or their company are going to buy new mobile phones. The two respondent (5.1%) that answered “other” said they buy phones continuously and whenever needed.

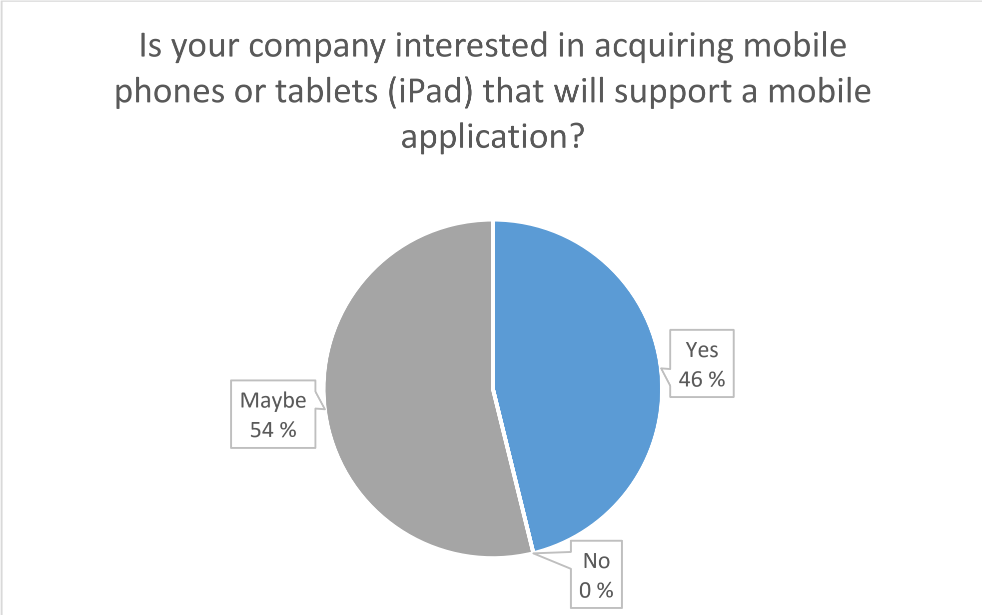


FIGURE 7.23: SURVEY, ACQUIRING SUPPORTED HARDWARE.

In Figure 7.23, forty-six percent answered that they would be interested in acquiring a mobile device that would support a mobile application. 54% percent said they might be interested, this could indicate that the respondent does not know if the company is willing to purchase a device or of it depends on the quality and need for the mobile application in question. None said that they would not be interested in purchasing a mobile device.

Question 4, summary:

The survey of ELA customers found that our customers and their employees have many different mobile devices and use several different mobile platforms. When developing an application for the ELA customers a requirement was that, if possible, Adcom Molde would supported the different types of mobile phones their customers use. This implied that the application had to be able to support more than one platform. Based on the result of the ELA survey and the statistical data seen in chapter 5, it was decided that the application should support Android, iOS and Windows Phone.

7.3.6 Question 5: Developer knowledge and available resources?

All of the questions presented by Anderson (2013) were concerning functions, users, distribution and operating systems, which deal with the various OC's of the mobile application. There is however an additional question that is important to consider before selecting a method which is about the environment where the application will be developed and who will be developing the application. This was a part of what Adcom Molde took into consideration before selecting a development method.

7.3.6.1 ELA Mobile

Question 4 revealed that the ELA Mobile application had to support multiple platforms. Because of this, the difficulty and cost of development increases. The order of these questions are also of importance, if the application is required to utilize a specific feature of mobile devices that may also mean you have to develop a native application. Using “lowest common denominator” (see the AIM-LOW method on page 33) as an approach is not a good way to develop cross-platform applications, since developers might end up with an application that does not work well on any platforms (Anderson 2013).

Knowledge

None of the Adcom Molde employees had any prior knowledge about developing applications (see Appendix A for details about the development department) for mobile devices such as Android and iOS. Both developers had some basic knowledge about Java (Android), but not about C and C# (iOS). However, they had worked with web technologies such as HTML, CSS, JavaScript and server side programming like PHP (Hypertext Pre-Processor) for several years.

Budget and timeframe

There was no defined estimates for a budget or timeframe before starting the project, since there weren’t enough knowledge in the organization about the development methods and processes to calculate this. However, time and costs were an important factor when choosing a platform for the mobile application. Adcom Molde did not have much time to develop a mobile application and could not invest the resources necessary to educate and train the programmer’s in new programming languages before developing the application.

Consumer interest

The ELA survey also asked the respondents about the potential interest for this kind of a mobile application in the power plant industry.

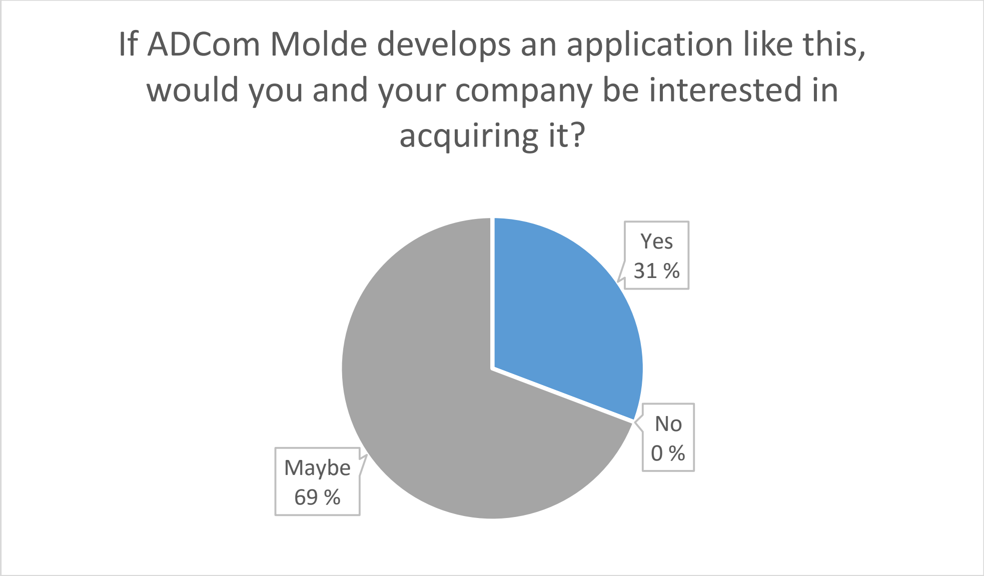


FIGURE 7.24: SURVEY, MOBILE APPLICATION INTEREST.

Thirty-one percent responded that they would be interested to purchase a mobile application from Adcom Molde if it was developed. None said that they would not be interested, but 69% would maybe be interested in purchasing the application. The large amount of responses that answered maybe could be because they are hesitant to answer yes. Investing in an application would depend on the quality of the application. It could also be that the respondent did not have enough information or is in a position to make such a decision on behalf of their employer.

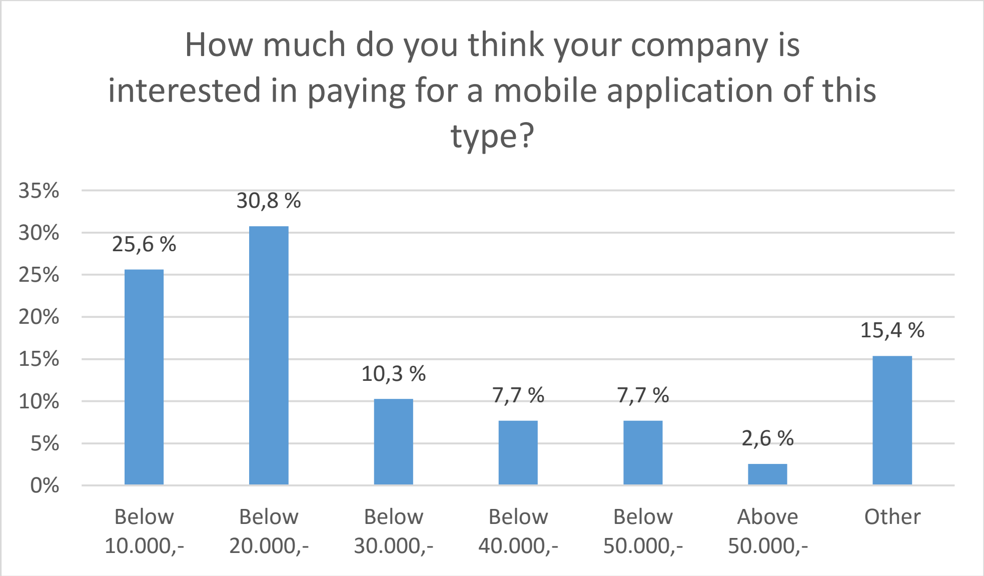


FIGURE 7.25: SURVEY, PRICE TOLERANCE.

How much the respondents would be willing to pay for an application varied some. Most of them (56.4%) would like to pay below 20,000. 25.6% of these would like to pay below 10,000.

Ten point three percent would like to pay less than 30,000, and 7.7% would like to pay less than 40,000. Seven point seven percent would like to pay below 50,000, while only 2.6% would be interested in paying more than 50,000.

Fifteen point four would be interested in paying another type of amount, perhaps some sort of license fee per user.

7.4 *Selecting a development method for ELA Mobile*

This section will try to summarize the questions analysis from section 7.3.2 to 7.3.6. These questions together with the pros and cons of the available development methods from OQ3 and OQ4, creates the outline for selection a mobile application development method with regard to the Adcom Molde organization and OQ1 and OQ2.

A comparative study of cross-platform apps by Andrade et al. (2015) found that hybrid applications can offer a simple solution for developing applications for smart phones and tablets while developing a native application offer limited to no cross-platform capabilities. Native development requires a high level of specialized knowledge in programming in addition to specific training for mobile development. Developing cross-platform applications using web technologies requires minimal level of investment in technical knowledge and time, but in a more limited way than hybrid applications.

Even if the level of fragmentation within the largest platforms (iOS and Android) are relatively small compared to older platforms such as Symbian and Java ME (see section 5.6), the fact that developers who develop native applications have to create a new codebase for each platform is unavoidable. Adcom Molde only had two developers (Appendix A). Since one of the developers only had time to focus on the development of the mobile back-end in addition to development on the ELA application this meant the other developer had to create the actual mobile application. As shown in Figure 2.2, the average learning curve for one developer to master the platforms Android (ca. 5 months), iOS (ca. 7 months) and Microsoft (ca. 9 months) would be over 20 months. A tripling of the code base combined with the average time to master each platform, did not make native development feasible for Adcom Molde.

The development options that remained were to use either a cross-platform tool or HTML5, both methods were considered good candidates. The specifications of ELA Mobile did not require other API's than access to the camera and potentially the GPS sensor. These API's where readily available in CPTs and the media capture[35] and geolocation[36] APIs for HTML5 were available for browsers that had implemented them.

Since Adcom Molde were scarce on time and resources and with a limited budget available for the project, it was key to keep the development simple. Deciding to keep the licensing method the same way as with the ELA software would make things simpler for both Adcom Molde and the customers. However, this was not something that was significant in deciding on what method to choose.

The thing that weighed the most in the decision was the existing knowledge within the organization. Since both developers in Adcom Molde had extensible knowledge and experience with web technologies it was almost a unanimous decision to choose HTML5 as a platform. In addition would a web application allow Adcom Molde full control over the entire distribution of the application and allow for easier and quicker updates of the application, more information about the distribution can be found on page 107 and in section 8.2.4. There was some hesitation from the Adcom management to whether or not HTML5 would be able to cover the customer needs since, at the time, it was not a finished standard yet. However, considering there where CPTs such as PhoneGap where a web application is wrapped inside a native application. A solution for this could be to merge the web application with PhoneGap or similar CPTs to be able to deliver a hybrid application to clients (more information about this topic can be found in section 9.6.4).

Since Daniel Huus (the researcher) had most the most experience and knowledge about the ELA application and its database structure he was assigned to develop the back-end of the mobile application. A web service would define the interface between the mobile application and the ELA database. Pål Gammelsæter was assigned with the task to create the front-end of the mobile application that would use the back-end to send and receive data to and from the ELA database. The technical implementation was discussed together with the rest of the development department, more information about the implementation of the development method in the organization can be found in chapter 8, "Adcom Molde mobile application development".

---

[35] HTML5 Media capture (refer to: http://www.w3.org/TR/html-media-capture/)

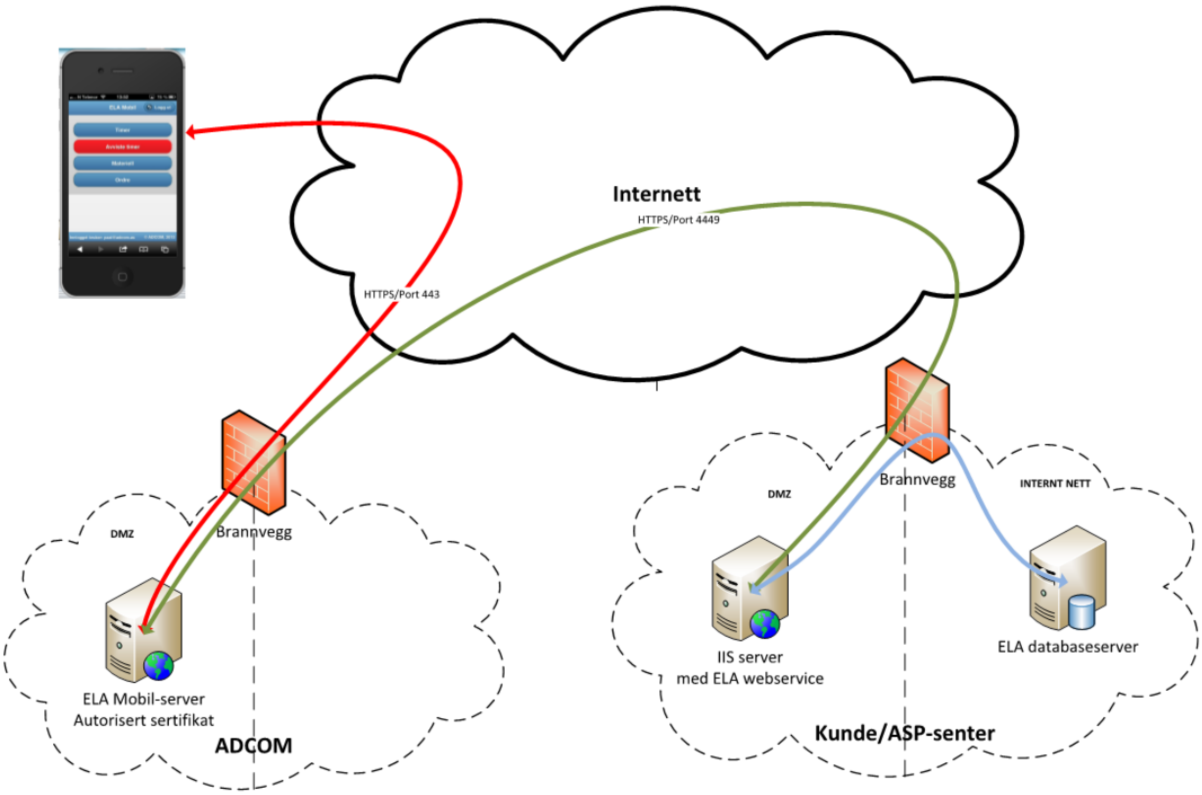
[36] HTML5 Geolocation (refer to: http://dev.w3.org/geo/api/spec-source.html)

8. Adcom Molde mobile application development

This chapter will give a detailed and chronological description of the action that was implemented in this research. Based on the operational questions OQ1 to OQ4 from the GQM reviewed in 7.4, this will define the operational question OQ5.

8.1 *ELA Mobile network topology*

By design, every ELA customer host their ELA database on their own servers and because of security reasons this database is not available from any outside networks. Since ELA Mobile needs to access data that is stored in the database the first thing Adcom Molde did after choosing a development platform, was to define how the communication should work between the mobile application and the customer's database server. Figure 8.1 shows a model of the network topology that was developed for ELA Mobile. Figure M.1 shows a model of the same network as in Figure 8.1, but created using Coloured Petri Net (CPN)[37] tool. The CPN model describes the flow of requests and responses in the network and allows for a better understanding of how the mobile application will communicate with the back-end servers. A detailed description of the ELA Mobile network is provided in Appendix M .



[37] Colour Petri Net (refer to: http://cpntools.org/)

FIGURE 8.1: ILLUSTRATION OF THE MOBILE APPLICATION NETWORK (ADCOM MOLDE).

8.2 *ELA Mobile development and frameworks*

When using HTML5 as a development platform, developers does not need any particular tools or frameworks to create applications. However, a lot of time-consuming tasks can easily be utilized to reduce the workload of creating mobile applications. Since HTML5 was chosen, there were multiple frameworks available. The next sections reviews the frameworks used in the implementation for this project.

8.2.1 **jQuery Mobile**

One of the frameworks chosen by Adcom Molde was jQuery Mobile[38]. The framework was selected since it offered vast support for multiple platforms and had full support for Android, iOS and Windows Phone, the platforms that ELA Mobile needed to support.

The majority of modern desktop, smartphone, tablet and e-reader platforms are supported by the jQuery Mobile framework. The ELA Mobile application even works on computers (as shown in 9.1.4), however this was not a consideration in the decision to use jQuery Mobile. The framework uses a three-tier (A, B and C) graded list of the platforms that are supported by the framework. Tier A has full support with an option of AJAX-based page transitions. Tier B includes the same except for the page transitions, while Tier C is the basic HTML experience (Hadlock 2012). ELA Mobile uses tier B, the reason for this is explained in section 8.2.4. The full list of supported platforms and browsers can be found on http://jquerymobile.com/.

jQuery Mobile is based on the original jQuery framework, a lightweight and robust JavaScript library that simplifies JavaScript coding and extends the capabilities of CSS. The framework allows for quicker scripting, less testing and coding (Hadlock 2012). The jQuery Mobile framework was first announced on August 11 in 2010 and is essentially a UI (user interface) framework which focuses on HTML. Building an application with jQuery Mobile does not require developers to write CSS or JavaScript. This means that web developers can utilize their existing knowledge to easily get started creating web applications. The jQuery Mobile framework is very different from other frameworks such as Sencha Touch[39] which

---

[38] jQuery Mobile (refer to: http://jquerymobile.com/)

[39] Sencha Touch (refer to: https://www.sencha.com/products/touch/)

uses JavaScript to define and layout pages. Another advantage with the HTML approach that jQuery Mobile uses is that the web application becomes touch friendly, since the content of the page is enhanced to fit mobile devices with smaller screen. For instance, regular buttons becomes larger and links can be converted to list based navigation systems. The content of a page is also possible to split into multiple pages with transitions. jQuery Mobile is open source and free to use for everyone. The framework is sponsored by many large companies such as Nokia, BlackBerry and Adobe etc. These companies have contributed with capital, hardware and developer resources (Camden and Matthews 2012).

8.2.2 Yii framework

The Yii framework[40] is a high-performance, component-based application development framework written in PHP, for projects of any scale. The name is pronounced Yee and is an acronym for "yes" and stands for "easy, efficient and extensible. Development of the framework was started January 1st, 2008 by by Qiang Xue. The first official release was December 3rd, 2008. To use Yii, developers have to know PHP and object-oriented programming[41] (OOP). Yii allows reusability in web programming since all Yii applications are built using model-view-controller[42] (MVC) architecture. The Yii framework also comes with a code-generation platform called Gii, previously known as the "yii shell" command that was available as a command-line tool. This feature allows developers to generate classes from database tables (Winesett 2010). The generated code can also contain constraints that verify data input for the applications.

8.2.3 Httpful

ELA Mobile also utilizes the Httpful framework/library. This is a small framework that allow developers to create a simple REST (Representational State Transfer[43]) client in PHP, through its API. With the client developers can call a web service through HTTP and retrieve JSON (JavaScript Object Notation) or XML (Extensible Mark-up Language). The framework supports custom headers, basic authentication and data parsing (Httpful 2015). More information about the web service can be found in section 8.2.5.

---

[40] Yii framework (refer to: http://www.yiiframework.com/)

[41] Object-orienter programming (refer to: https://en.wikipedia.org/wiki/Object-oriented_programming)

[42] Model-view-controller (refer to: https://en.wikipedia.org/wiki/Model-view-controller)

[43] REST (refer to: https://en.wikipedia.org/wiki/Representational_state_transfer)

8.2.4 Application composition

The Yii framework works as the skeleton in ELA Mobile. It provides the business logic in the application and merges the various frameworks together to create the application. When the clients access the URL of the web application, the Yii framework is initialised and generates the requested web page using a combination of PHP, HTML, CSS and JavaScript. The various pages created in the application are single page files called "views" in Yii. These pages reference to jQuery Mobile framework in the HTML code using div and form with various tags and attributes to create the user interface that is displayed on the device.

Most of the information that is displayed in the web application comes from the client database using a RESTful web service. These data are accessed using the Httpful library in PHP. Using PHP also adds to the level of security, other alternatives such as Python and Angular could also have been used. An example from the ELA Mobile login page is shown in Appendix N.1 and the result is presented in Appendix N.2.

The jQuery Mobile page transitions with AJAX is disabled in ELA Mobile since the Yii framework is not compatible with this mode. The various pages is created with Yii on the server when it is requested by the user. To do this the server has to get a new URL request, something that can't easily be achieved with AJAX. The code that disables this feature is shown in Appendix N.3.

The ELA Mobile application is hosted on a web server in Adcom Molde's network (seen to the bottom-left in Figure 8.1). Each customer access the web application with their own sub-domain (see Table 5 for details) such as "http://customername.elamobile.no". By dividing the customers into subdomains it is possible to retrieve the specific configuration for the customer so the correct is stored.

8.2.4.1 Media Capture API

The only API that was needed in the first version was the Media capture API. The API enable users to take a photo and upload it to the selected work order in ELA Mobile. If the feature is supported on the device it will give the web application access to the camera and on some phones users will also be able to choose an existing image from the phone gallery. Only one line of code is needed to access the camera function in HTML5. The code for the Media Capture API is shown in Appendix N.4 together with an example of how the user interface looks to the users.

8.2.5 **Back-end development**

To be able to read data to and from the ELA database, Adcom Molde created a web service that query the database on behalf of the mobile device and delivers data in a light-weight and readable format. The web service handle all communication between the ELA database and the ELA Mobile application.

8.2.5.1 Windows Communication Foundation

The web service was created using Windows Communication Foundation[44] (WCF). WCF is Microsoft's framework for building service-oriented applications that allows for interoperability with security layers and is AJAX, JSON/XML and REST compatible (Microsoft 2013). Since the ELA software and database is based on Microsoft technology it was natural to use the same platform to develop the web service. The first version of WCF was released as a part of .NET 3. The back-end for ELA Mobile was built using .NET 4.0. The web services task is to expose functions that deliver, store and manipulate data from a Microsoft SQL database. The service exposes the available service interfaces using technology-neutral metadata using Web Services Description Language (WSDL). This metadata can be imported by non-WCF clients to its native environment as native types (Löwy 2010).

Service contracts are used to describe what operations the client can perform on the web service. The service contract attribute can be applied to either an interface or a class, this will allow other applications to see and access it. Each service contract can contain multiple operation contracts. This attribute can only be applied to methods. The operation contract expose the method as a logical operation that can be performed on the service contract. As shown in the example provided in Appendix O.1, the ELA Mobile web service uses an interface called "IELAPPService" with the Service Contract attribute. A class in the web service implements the interface and handles all methods exposed in the service. Everything from login and authentication to file uploads.

Together with service contracts the service also need to send and receive data between the clients. This data has to be in a format that other applications can read. Data contracts define the data types that are sent to and from the web service. It is possible to define custom datatypes, as well as default types such as strings and integers. Meta information about the

---

[44] Windows Communication Foundation (refer to: https://msdn.microsoft.com/en-us/library/ms731082)

various data contracts are published together with the WSDL. The data contract allows for serialization so the data can be read from all platforms (Löwy 2010).

8.2.5.2 Entity Framework

To easier exchange data with the database the web service uses the Entity Framework[45] from Microsoft. The Entity Framework was first released in July 2008 as a part of Visual Studio 2008 SP1 and .NET SP1. The Entity Data Model (EDM) is the core of the Entity Framework. EDM is a client-side model that describe the structure of business object from an underlying database model with tables and relationships between them. The Entity Framework can connect to multiple database types such as Microsoft SQL server, Oracle, MySQL, SQLite and VistaDB and import the database schema into the EDM of a Visual Studio project. The items that are described in the EDM are called entities. (Lerman 2010).

The EDM of the pay art table from the ELA database (Microsoft SQL) can be seen in Figure 8.2.

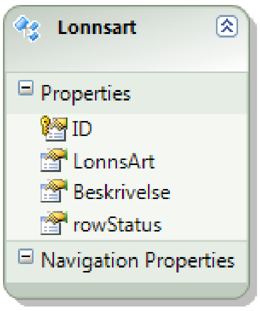


FIGURE 8.2: THE PAY ART TABLE FROM THE ELA DATABASE IMPORTED TO THE EDM.

From the entities in the EDM we can automatically generate code and create entity classes/entity objects. This is achieved using Visual Studio's Text Template Transformation Toolkit[46] (T4). These templates can be altered to output customized classes, if needed (Lerman 2010). After adding a template, the Entity Framework creates classes from the selected tables when the project is saved. An example of the auto generated class for the table in Figure 8.2 is shown in Appendix O.2.

The web service sends messages by using REST. A REST service is stateless client-server protocol that is platform and language independent and uses well-known standards

[45] Entity Framework (refer to: http://www.asp.net/entity-framework)

[46] Text Template Transformation Toolkit (refer to: https://msdn.microsoft.com/en-us/library/bb126445.aspx)

such as HTTP/HTTPS. REST does not have any built-in security features like encryption (except when using HTTPS), session management, quality of service (QOS) guarantees etc. However, it is possible to implement these features in the service if needed (Elksten 2014). The code example on page 204 have an Operation Contract called “getLonnsart()”. The return type “ELALibrary.Domain.ELA.Lonnsart” is the same class that is shown above. The web service also specify the UriTemplate, this is a URI (Uniform Resource Identifier) that define where the resource is located. The UriTemplate “/get/lonnsart” will translate to the URL (Uniform Resource Locator) “http://domain/get/lonnsart”. Each operation contract also specify what Method is used on the resource, either POST, GET, PUT or DELETE which translate to create, read, update, delete (CRUD[47]).

WCF will automatically translate the class objects to the preferred return format. The operation contracts shown on page 204 has specified the RequestFormat and ReturnFormat to send and receive JSON. However, the client application can override the request format if it prefers another format, such as XML. JSON is a simple text format that represent objects as a collection of name and value pairs or arrays. The JSON data structure is the same as the structure of a programming language. JSON is not a mark-up language like XML. There is no formal description of the data with other attributes, except a name, value and that strings have quotation marks around them. JSON is however an object language so it is of course possible to describe the content of the JSON to the extent you need, but the syntax will then be your own and not pre-defined by the document standard (JSON 2014). Two JSON examples is shown in Appendix O.3.

[47] CRUD (refer to: https://en.wikipedia.org/wiki/Create,_read,_update_and_delete)

8.2.6 Major events and version releases

Figure 8.3 below shows a timeline with a summary of every major event that occurred during the research timeframe.

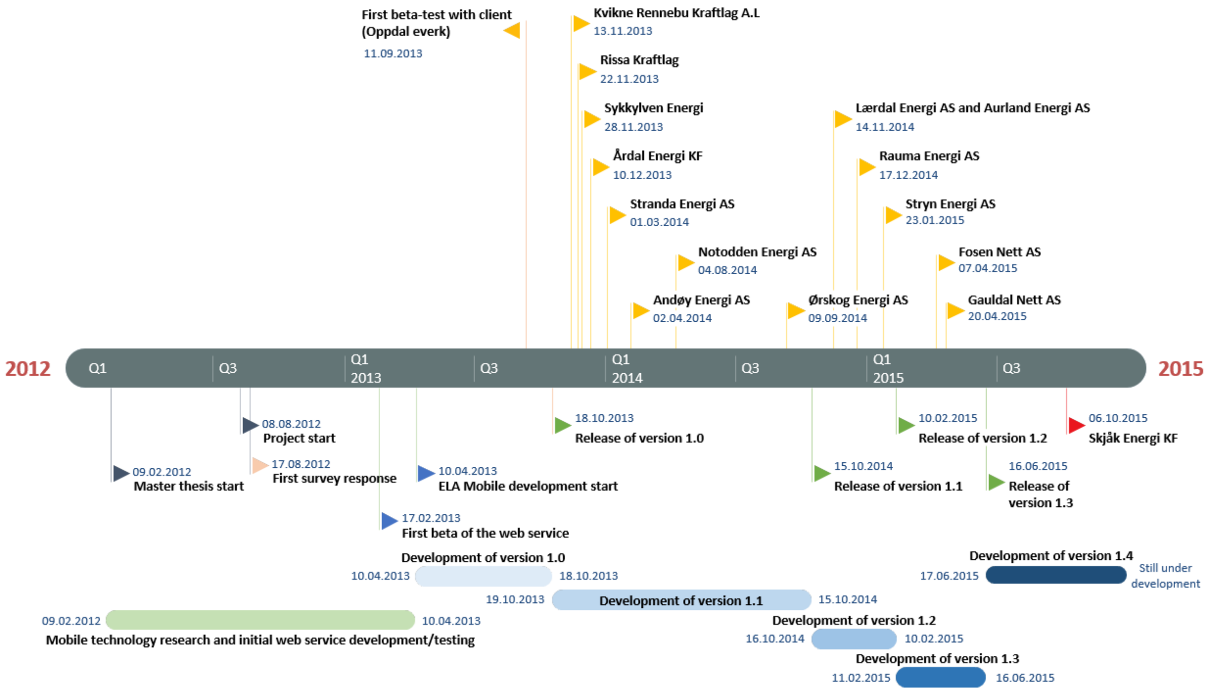


FIGURE 8.3: PROJECT TIMELINE.

During the first year of the research (in 2012) the ELA customer survey was designed and a technology review was conducted. Six months after (08.08.2012) the master thesis start, development/testing of the web service technology for the ELA Mobile back-end was initiated. As seen in Table L.1, ca. 32 hours was spent on this. The first survey response was registered 17.08.2015 and the ELA Mobile development phase started eight months later on 10.04.2013, when the ELA Mobile project was created and the first code was written.

Below is a list of major releases of the ELA Mobile application and some of the details on each release.

8.2.6.1 Version 1.0

The first version of ELA Mobile was released with version 6.3.5 of the ELA application on the 18.10.2013. The first release contained hour and material registration and displayed work order information. There was also created pages to view a summary of the hour registrations with day, week and month intervals. These pages also displayed specific details on how many workhours there are in the selected interval, any hour lines that was disapproved by a manager would be highlighted in the hour list with a red colour and a red button with a shortcut to the disapproved hour lines would be displayed in the main menu.

8.2.6.2 Version 1.1

The second major version was released together with version 6.3.6 (dependent on upgrade of ELA to upgrade ELA Mobile) on 15.10.2014. Almost one year after releasing version 1.0.

- Various bug-fixes (none related to compatibility issues).
- Added file archive on work orders.
- Added file upload (by using the HTML5 media capture API)
- Bug fix in ELA admin (configuration tool): Password on the file upload user in was being reset in the admin panel.
- Added Google Analytics (see section 9.1 for more information).
- Various changes in the user interface and optimizations of misc. API calls.
- Updated JavaScript framework versions (jQuery and Httpful).
- A new top five work orders list, as well as a top five hour type list (pr. User).
- Optimized search (changed to only retrieved 50 hits pr. Page with an option to go the next 50 hits).
- Fixed a bug related to iOS:
    - If user had version 8.1.2 of iOS, dropdown menus made with optgroups[48] in JQuery would fail if the first element was not selected. This meant writing code that checked if the user agent of the mobile device was iOS v8.1.2 and then display an alternative menu.

8.2.6.3 Version 1.2.x

Version 1.2 was release 10.02.2015 together with ELA version 6.3.7, ca. four months after version 1.1. This version was also dependent on upgrading ELA before upgrading ELA Mobile. This version was able to release smaller releases to each customer if needed (1.2.1, 1.2.2 … 1.2.x).

- Fixed a bug related to iOS:

[48] Optgroups (refer to: http://www.w3schools.com/tags/tag_optgroup.asp)

    - The bug with optgroups from the previous version came back as an issue when Apple upgraded iOS to version 8.1.3 and v8.2. Apple had not resolved the bug and since ELA Mobile only checked for version user agents with v8.1.2 the optgroups menu would be shown when the user agent was updated to v8.1.3/v8.2.
        - https://github.com/jquery/jquery-mobile/issues/7848
    - The issue was traced back to AppleWebKit v600.1.4 and that several browsers experienced the same issue:
        - https://bugs.webkit.org/show_bug.cgi?id=137261
    - The bug was fixed in version 8.3 of iOS.
- Various bug-fixes not related to compatibility issues.
- Various changes in the user interface.
- Possible to show hour bank summary in the hour lists.
- Fixed a bug related to Internet Explorer (windows):
    - Internet Explorer could not handle ÆØÅ symbols when searching for articles on the material registration page.

8.2.6.4 Version 1.3.x

The latest version of ELA Mobile was released 16.06.2015, together with ELA version 6.5. No major releases for ELA version 6.4 where made available.

- Possible to change work order status to "work completed" in ELA Mobile.
- Risk analysis forms created in ELA where available to fill out and sign on ELA Mobile.
- A new comment function where made available on work orders. Instead of on large comment field, new continuous commenting was possible (together with name and date of commenters).
- Possible to search in all orders in the hour registration form.
- Customizable hour registration form. Possible to choose a pre-set or a blank for hour line input.
- Various changes to the user interface.
- Various bug fixes.
- Updated Httpful version.
- Fixed a bug related to iOS:

    - A bug in iOS caused a bug while searching in the work order list.
- Fixed a bug related to iOS:
    - A bug in iOS caused an error with autofocus on an element after pressing a link/popup. In this case it was a drop-down list that would display all elements in the list when it received focus.
        - https://forum.jquery.com/topic/how-to-prevent-autofocus-on-first-input-when-clicking-a-link-on-popup-in-jquery-mobile
        - http://stackoverflow.com/questions/28861943/how-to-prevent-autofocus-on-first-input-item-when-i-click-a-link-on-popup-in-jqu
    - To resolve the issue the upgrade of the JQuery framework was put on hold until the issue would be resolved
        - https://github.com/jquery/jquery-mobile/issues/7955
        - https://github.com/jquery/jquery-mobile/issues/7856

8.2.6.5 Version 1.4 (under development)

Version 1.4 is currently under development and is planned to be released with ELA version 6.6 in Q1 2016.

8.2.6.6 Customers and version distribution

Figure 8.3 also lists the dates of when installation of the ELA Mobile application was conducted with customers. Every yellow flag represent a new ELA Mobile customer. Rissa Kraftlag was merged with Fosen Nett AS in 2015 and is therefore not shown on the list of companies that currently use ELA Mobile in Table 5. Skjåk Energi KF (red flag in Figure 8.3) is also not the list since they wanted a trial of ELA Mobile before they would choose to implement it in their organization. Their trial is now completed, however they have currently chosen not to implement ELA. The feedback from Skjåk Energi KF was that they experienced that the screen on their devices were too small to use ELA Mobile. And it is possible that they will have to upgrade their devices in order to use ELA Mobile.

Aurland Energi AS and Lærdal Energi AS is shown as one new client since both companies use ELA and ELA Mobile and the companies share servers. The installation for both companies was conducted on the same day.

As seen in Table 5, there are 14 different companies that are using ELA Mobile. Most of them are using the latest version 1.3.x, except for two customers who are still using 1.2.x.

Customer	Domain	1.0	1.1	1.2.x	1.3.x
Andøy Energi	andoy.elamobil.no			x	
Aurland Energiverk AS	aur.elamobil.no				x
Fosen Nett AS	fosen.elamobil.no				x
Gauldal Nett AS	gn.elamobil.no				x
Kvikne-Rennebu Kraftlag A/L	krk.elamobil.no				x
Lærdal Energi AS	lar.elamobil.no				x
Nordvest Nett AS	nvn.elamobil.no			x	
Notodden Energi AS	ne.elamobil.no				x
Oppdal Energi	oev.elamobil.no				x
Rauma Energi AS	rauma.elamobil.no				x
Stranda Energi AS	stranda.elamobil.no				x
Stryn Energi AS	stryn.elamobil.no				x
Sykkylven Energi AS	sye.elamobil.no				x
Årdal Energi	aae.elamobil.no				x

TABLE 5: ELA MOBILE CUSTOMER LIST AND VERSION DISTRIBUTION. LAST UPDATED 28.11.2015 (ADCOM MOLDE).

9. Evaluation and analysis

This chapter will review the data analysis process as well as the findings from the action that was implemented. This will define the final operational question (OQ6) from the GQM.

9.1 *Google Analytics*

Since ELA Mobile is a web application it was possible for Adcom Molde to use Google Analytics[49] to gather quantitative usage statistics of the application. Unfortunately, there was an error in the implementation and usage of the Google Analytics API and not all subdomains (*.elamobile.no) gathered data, therefore only data from 08.10.2015 to 03.12.2015 are used. The next section will present the findings from these statistics.

9.1.1 Data filters

In addition to the date interval, some data have been excluded since it is not relevant or is false traffic generated by web crawlers and bots. The data filters applied to the dataset include country and hostname. Only visits from Norway and visits that match one of the hostnames listed in Table 5 are included.

9.1.2 Data types and data analysis

Between 08.10.2015 and 03.12.2015 ELA have seen 803 users and a total of 521 from a mobile device. However, these numbers are most likely inaccurate. The data from Google Analytics state that Sykkylven Energi have had 123 unique users visit their domain sye.elamobil.no. However, data from PROFF[50] state that Sykkylven Energi has 27 employees, this is also the same number of users that are set up to have access to ELA Mobile in their ELA database. A user is identified by using first party cookies with a unique identifier that is sent with each request. The cookie can last for two years, but if the cookie is deleted a new cookie and identifier will be generated making the data on distinct users inaccurate. One user can of course use more than one device to access the application, but this would mean that all 27 of Sykkylven Energi's employees would had accessed ELA Mobile from over 4 different devices (not considering a reset of the cookies). Instead of users it is more reliable to analyse session data. A sessions is a collection of hits from the same user that are grouped together

---

[49] Google Analytics (refer to: http://www.google.com/analytics/)

[50] PROFF, Sykkylven Energi (refer to: http://www.proff.no/selskap/sykkylven-energi-as/sykkylven/energiforsyning/Z0IR2EUK/)

(Cutroni 2014). One session is limited to 30 minutes in ELA Mobile. This means that interactions with ELA Mobile with a time interval above 30 minutes will count as two sessions. If a user deletes the cookie a new user will be counted, but at the same time only generate one new session. Google Analytics data are not 100% accurate, but they will provide an indication of the real world situation.

9.1.3 User sessions

Figure 9.1 shows a summary of the usage statistics from ELA Mobile collected using Google Analytics. The graph show an overview of sessions from ELA Mobile. There are three metrics represented in the graph. The blue line represent all sessions and the orange and the grey line is a subset of the blue line. The orange line represent sessions where the user used a mobile device such as a mobile phone or a tablet and the grey line are users that only accessed ELA Mobile with one session.

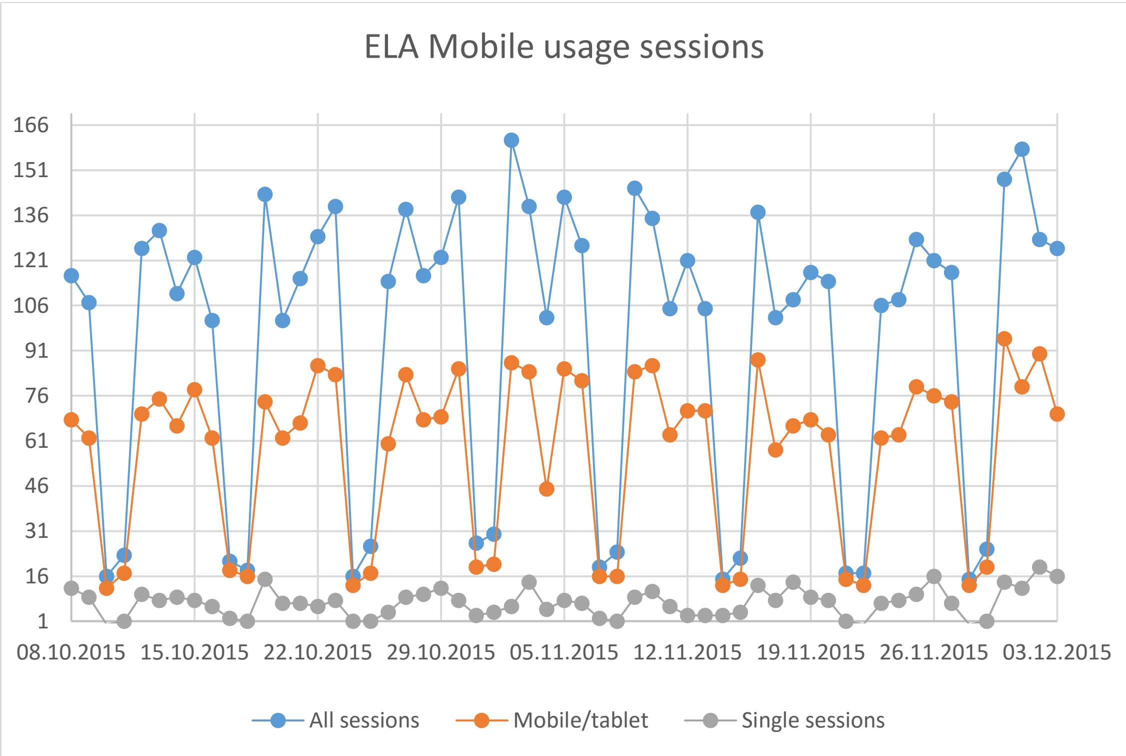


FIGURE 9.1: ELA MOBILE USAGE STATISTICS

Most users have more than one session with ELA Mobile, only 7.52 percent of the total number of sessions are single sessions. According to the Google Analytics dashboard the average number of sessions per user is 6.68 across all devices and 6.25 for mobile devices. Of the total amount of sessions in ELA Mobile, mobile devices account for 60.33% across all weekdays and 59.3% for workdays. However, during the weekend the percentage of sessions on a mobile device is 76.13%. Ergo, 16.83% higher than during weekdays. This indicates that some of the ELA Mobile users have a desk job and uses ELA Mobile actively on a computer during work hours. The average session duration for all sessions are 7 minutes and 10 seconds and 5 minutes and 20 seconds for mobile and tablet sessions. The single session user's use on average 6 minutes and 34 seconds in ELA Mobile.

9.1.4 Operating systems

Table K.1shows a list of all operating systems that have been registered on ELA Mobile. A total of 6 different platforms have been used to access ELA Mobile. Three mobile platforms account for 60.34% of the sessions (Android, iOS and Windows Phone) and three desktop platforms that account for 39.66% (Windows, Linux and Macintosh). The most surprising about these statistics is the large number of user sessions from a PC platform and that the Windows Phone platform only account for two sessions. The global sales statistics in Table C.1 shows that Windows Phone had a market share of 3.3% in Q2 2013 and the ELA customer survey found that 2.6% had a windows phone (one respondent). With only two sessions across two months means that the usage of Windows Phone in the ELA customer group is close to non-existent (0.04%). The same can be said about Linux and Macintosh, however desktop platforms was not a part of the platforms that was said to be supported in the initial application requirements.

The two sessions (one on 19.10.2015 and one on 04.11.2015) from Windows Phone both used the same versions and is according to Google Analytics the same user.

As seen in Table K.2, Windows is the preferred platform with Windows 7 as the most used version. Something to notice is that there are many sessions that are still using older platforms such as Windows XP and Windows Vista. It also appear to be multiple users on the Macintosh platform since multiple versions are represented. The Linux platform only seem to provide information about its system architecture. According to the ELA customer survey in Figure 7.15, 66.7% utilize laptops as a tool at their workplace. However, it is unknown if the user sessions in Table K.2 are from a laptop used out of office or a desktop computer.

9.1.5 Mobile devices

Table K.4 shows a complete list of the 39 device names that where registered in the dataset. The list of Android devices is the most diverse and include both tablets and mobile devices. The list of Android devices also contain some data that does not reveal the device name such as "Mozilla Firefox for Android/Android tablet" which is a browser. 175 sessions did not include a device name as seen at the top in the “(not set)” category. Excluding these there are 34 different Android devices using ELA Mobile in the dataset.

There are also many sessions from iOS devices. At first glance there seem to be only two device models represented on iOS. However, looking at Table K.3 we see that there are 12 different versions of iOS which can indicate that iOS does not provide the model version for these statistics. As stated earlier there is only one user that uses Windows Phone in two sessions, the phone model here was a reported as Nokia Lumia 925.

9.1.6 Browser types

In addition to multiple operating systems, Table K.5 shows that there are also a large diversity in the types of browser that are using ELA Mobile. There are seven different browsers types represented in the statistics with many of them being used in multiple versions. The most fragmented browsers are Chrome (19 versions), Safari (6 versions) and Opera (5 versions).

As discussed in section 6.2.1.2, implementation of web standards can vary from one browser version to another. The ELA Mobile user experience can potentially be different in the various browsers and some browsers may even lack support of features being used.

9.2 *Customer feedback*

A couple of customers that had started using ELA Mobile held a presentation about their experience in the user forum in 2014.

9.2.1 Oppdal Everk

The presentation from Oppdal Everk[51] was given by Bjørn Størvold[52] a coordinator at Oppdal Everk. In 2010 Oppdal Everk tried to use laptops with a mobile connection to access ELA while out of office, similar to the method presented by Adcom in the user forum in 2007. However this method was not very successful and was not satisfactory to their requirements.

Oppdal Everk was a pilot customer and started using ELA Mobile in August 2014. The feedback from their employees was very positive. To their experience their hour lists are much more accurate than before since their users are now able to register their hours during the day. The same goes for material registration, also here there is less deviations than before.

[51] Oppdal Everk (refer to: http://oppdal-everk.no/oppdal-everk/)

[52] Bjørn Størvold (refer to: http://oppdal-everk.no/ansatte/)

Oppdal Everk also stated that there have been little training and instructions in the usage of the application, in spite of this there have been few questions after the implementation of ELA Mobile in their organization. Finally, Bjørn Størvold ended the presentation by saying “ELA Mobile is like Facebook – it is here to stay”.

9.2.2 Rissa Kraftlag

The presentation from Rissa Kraftlag[53] was given by Ståle Rostad[54] an operations manager and engineer at Rissa Kraftlag. He stated that their company consists of 30 employees, where 10 works in the maintenance department (in Norwegian, “nettavdeling”). Their employees used to write their hour lists on paper that later where manually typed in to ELA. Later they tried setting up a computer where all technicians could fill in their hours, but this method caused a lot of queues and employees had to wait in line to fill out their hours. When they implemented ELA Mobil they bought every technician a mobile tablet that they could use the application on. The usage statistics from Google Analytics shows that 78 of 455 sessions from a mobile device are from a tablet, while the rest are from various mobile phones. 381 sessions are from windows users. They met little resistance and the feedback have been positive. Their hour lists are now up to date and with few errors. The only thing they miss is hour bank functionality for time off, holiday, absence and sickness (since then hour bank functionality have been implemented in ELA and ELA Mobile, see section 8.2.6.3 for more information). They had to use Excel sheets to keep track of this type of information by registering this information as own work orders in ELA. They feel ELA have good reports that shows an overview of various pay arts such as over time etc.

The only negative feedback is that user’s register hours on work orders that they are not connected to and it is hard to go back and find orders in the order list and that in the hour registration form you search in all work orders. This have since been improved in later versions of ELA Mobile, see section 8.2.6 for more information.

By using ELA Mobil Rissa Kraftlag are able to see who is behind on their hour registration and they have good control over projects and their costs since everything is registered continuously, which in turn makes the information more correct. All in all they feel that it is working well.

[53] Rissa Kraftlag (refer to: http://rissakraft.no/)

[54] Ståle Rostad (refer to: http://rissanett.no/stale-rostad/)

The maintenance department in Rissa was merged with Fosen Nett 01.01.2015 and continues to use ELA Mobile.

9.3 *Mobile developer interview*

Pål Gammelsæter was in charge of developing the front- end of the mobile application for Adcom Molde and answered some questions in an interview (Appendix J) about the development process and which experiences he made when developing the application. Gammelsæter is a web developer and have best knowledge of PHP and HTML. In addition to the Yii framework that is used in ELA Mobile he have some experience with other frameworks such as Concrete5 (CMS) as well. Gammelsæter states that Adcom chose the jQuery Mobile framework for ELA Mobile because it was the largest available framework at the time. He felt it was safe to use a well-known framework since it would provide technological stability in addition to a predictable future with continued development. Since he did not have any prior experience with other mobile frameworks it was hard to compare the learning curve with other frameworks he had used. Overall he felt that it was a medium challenge to learn jQuery Mobile. However, he thinks that developers have to be very familiar with web technologies and that developers have to know HTML/CSS to be able to use jQuery Mobile and start development quickly. Knowledge about server configurations will also be an advantage. One challenge with the jQuery Mobile framework is that it uses attributes that are not standard in HTML and have to refer to the frameworks documentation every time he need to change an element.

Gammelsæter was surprised that there weren't more variations between devices. The interoperability of the jQuery Mobile framework is very good and is compatible with the different platforms, this is also evident from the usage statistics in section 9.1. Internet Explorer have caused some issues, but since it is a desktop browser it is not expected to work well with a mobile framework. In addition, as previously mentioned desktop computers was not one of the platforms that Adcom Molde wanted to support. However, since users also utilize desktop computers Adcom Molde have been doing bug-fixes towards these browsers as well, to ensure that it works for all users. One of the more time consuming issues with the framework or rather the various devices/browsers were the keyboard layout for some types of input fields. Input fields where users register decimal numbers need either a comma or punctuation before the decimal digit. However, some layouts only show numbers for these inputs. To overcome this Gammelsæter had to create configuration page where users could select the correct keyboard type. The selected keyboard was saved in the user settings and used to configure the attributes on the input field so it would display the correct keyboard layout in the registration form.

In retrospect, Gammelsæter thinks that developing a web application for ELA and Adcom Molde was the right decision, since it is not an application that require a fast interface and all the user need is a shortcut to the application on the home screen on the device. Personally he is not a fan of this type of application, but for this purpose it was the best choice. Gammelsæter states that during the development of a web application developers might find that they should have developed a native application instead, since the application is dependent of an internet connection and that the response time in the user interface is not very good. Knowing what limitations web applications have compared to native applications is key. ELA Mobile supports a large amount of devices and a large part of the customer's needs. In addition it is very fast to update the application, as soon as the application is updated the same update is available for all users. Adcom Molde have not received any complaints yet and Gammelsæter also feel that people are impressed with the application.

9.4 *Observations*

There were some challenges that had to be tackled while developing ELA Mobile. As stated earlier, ELA Mobile uses multiple third party frameworks that supply multiple features and functions that held reduce the workload of development. Pål Gammelsæter had never used jQuery Mobile (see section 8.2.1) or the Httpful (see section 8.2.3) frameworks before. Implementation of the Yii framework (see section 8.2.2) had just started in another project, so he was fairly unexperienced with that framework as well. Gammelsæter states in retrospect, that the first code for ELA Mobile was cluttered and did not use what was considered best practise for the frameworks since he did not know the full structure and the inner workings of the frameworks. Even if he was familiar with HTML and web development there was still a learning curve to code in a fashion that the frameworks was intended for.

Communication with the back-end also faced some challenges, after the release of the first version of ELA Mobile there had to be made several adjustments to various API calls from ELA Mobile to the back-end service with the customer. Since every call to the service is synchronous and not asynchronous the mobile application have to wait for a response from the back-end. If there was transferred a large amount of data or many API calls at the same time there would be a delay and an unresponsiveness in the mobile application. In some cases there was even some time-out issues, but these were related to SQL queries that queried too much data at the same time (multiple joins etc.). Since the web service was waiting for a response from the SQL server it did not respond back to the mobile application in time and a time-out occurred between the mobile application and the back-end. By tweaking the SQL query this issue was resolved. The greatest transformation in reducing the delays and unresponsiveness laid in the hour list summary pages, since these pages can consist of many hour line rows as well as multiple calculations. As seen in Figure 9.2, a summary of every week in a month is displayed in a list. In version 1.0 all data for day, week and month view was retrieved when first loading the page. This mean that every hour line for the selected view was retrieved from the back-end and then the mobile application would calculate the sums based on these data rows. This was a time-consuming process and meant transferring an unnecessary large amount of data. This caused clients that had many employees or who was thorough with hour registration to experience slow loading time on these pages, in extreme cases a loading time of over 10 seconds could occur. To optimize the loading time and reduce the amount of data transfer the calculations was built-in to the back-end. The mobile application query the back-end for a date interval or an array of intervals and the sums of the intervals is returned. The result was a reduction of loading time from several seconds down to milliseconds. This simple, but effective change was vital for a good user experience in ELA Mobile as well as reduce data traffic costs for customers.

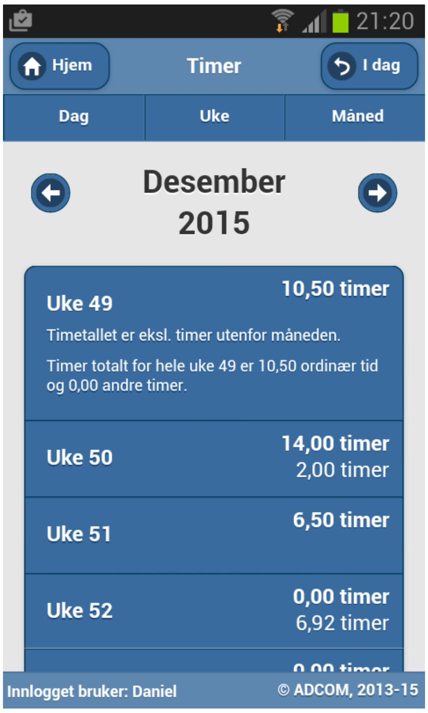


FIGURE 9.2: ELA MOBILE HOUR LIST, MONTH VIEW (ADCOM MOLDE)

9.5 *ELA Mobile project costs*

This section will review the costs associated with the ELA Mobile development. After reviewing the costs for the ELA Mobile project in section 9.5.1, the costs of a native development approach will be reviewed in section 9.5.2. This is done to try and get a comparison of two different mobile development methods.

9.5.1 Web development costs

Table L.1 shows a summary of the hours each employee in the development department have used on the ELA Mobile project. The list is a summary of every hour line that has a reference to the ELA Mobile development, such as an attached job description or a comment. The summary is thus just an estimation of the hours spent on the project and is reliant on the prerequisite that the attached description is accurate. Hours that have no accurate description and no obvious link to the project attached to them is not included.

Since the start of the project and up until 28.11.2015, the development department in Adcom Molde spent a total of 1665 hours working on the ELA Mobile project which translates to ca. 0.951 year's work. With an internal cost of ca. 500 NOK pr. Hour this is a labour cost of ca. 832,500 NOK. The total cost of the ELA Mobile development for Adcom Molde has ha negative result of -235,500 NOK.

9.5.2 Comparison with a native development approach

The internal cost of the front-end development of ELA Mobile for the first version of ELA Mobile has an internal cost of 153,500 NOK (Appendix L.3.4). As seen in Figure 8.3 version one of ELA Mobile was released after ca. 6 months of development, from 10.04.2013 to 18.10.2013. Figure 2.2 show that six months is also the average time it takes to master mobile web as a platform. However, Pål Gammelsæter was already very familiar with web development and as stated in an interview (Appendix J) that he estimates that it would only have taken ca. two months working full time on the ELA Mobile development. As seen from the hour list this is also a close estimate.

By using the average time it would take to master native platforms, it is possible to get an estimate of how many hours it would take to develop the first version of ELA Mobile as native applications. As discussed in section 7.4, 20 months would be the average time to master the native platforms Android, iOS and Microsoft. If development would have taken a 20 months with full-time development the internal cost of these hours would then have been 1.62 million NOK (Appendix L.4.1). However, since the first version of ELA Mobile was a very light weight application that only viewed and registered data, a more careful estimate should be considered.

By using the same timeframe of 20 months, but with the same average of hours used pr. month for the ELA Mobile development we find that that Pål Gammelsæter would have used 1020 hours to develop the first version of ELA Mobile as native for the three platforms. This would have an internal cost of 510,000 NOK and would be 356,500 NOK more than the web application that was implemented in Adcom Molde. In addition to 14 months longer development time (Appendix L.4.2).

Using the same ratio (native vs. web) on the total amount of hours spent on front-end development during the whole project estimates to 2636 hours and would have a cost of 1,318 million NOK. This is 1.506 year's work and would mean that the total front-end development would have been over a year's work more. In addition it would have been an additional half year's work more than compared to the total year's work spent in the entire development department in the implemented web approach (Appendix L.4.3).

9.6 *ELA Mobile improvements*

This section will review some of the improvements that could be applied to ELA Mobile in the future. The improvements are based on observation and discussion in the development department.

9.6.1 Responsiveness and offline functionality in ELA Mobile

As discussed earlier ELA Mobile does not use transitions when moving between views/pages (see section 8.2.1 and 8.2.4). If pages would load in the background the responsiveness in the application would improve since the loading time would be reduced. To achieve this asynchronous API calls to the back-end would also have to be implemented. Local storage (Appendix F.7) could also improve loading time by preserving data between sessions and possibly even add offline functionality to preserve data when the mobile device loose connection to a network. Offline functionality will be the topic of a new research project that Adcom will conduct together with Sør-Trøndelag University College[55] (HIST) in the spring of 2016. The project will be a part of a student's bachelor thesis and will be conducted as co-operation between the University and Adcom Molde similar to this thesis.

[55] Sør-Trøndelag University College (HIST) (refer to: http://hist.no/)

9.6.2 WebSocket

The back-end of ELA Mobile is something many users know about since it can't be seen and it works in the background. However, developers and technicians have to set this up in the customer's local network which requires that the customer have a server in the DMZ. Usually, this is not something that customers already have in their network and is therefore an expense that comes with implementing ELA Mobile. Some network administrators feel that this type of server/service is a security threat, even with the measures that are implemented to keep it secure. Because of this Adcom Molde is considering WebSocket as an alternative method for the back-end. WebSocket will allow for interactive communication between the web server located in the Adcom Molde network and the customer server. This interactive session can be initialized from inside the customer network without the need for a server in DMZ (Fette and Melnikov 2011). This will reduce the cost of implementing ELA Mobile in a new organization and will reduce the amount of configuration with each installation (firewalls, access rights etc.) and can possibly also be an even more secure method to exchange data with the customer database.

9.6.3 Frameworks

In October 2014 the Yii2 framework was released. This new version of the Yii framework have been completely rewritten, and upgrading from version 1.1 to version 2.0 is not as simple as upgrading between minor versions. Version 1.1 that ELA Mobile uses is considered the old generation and development is only in maintenance mode (Yii Software LLC 2014). This means that upgrading to the new version in future versions of ELA Mobile will have to be considered.

9.6.4 ELA Mobile as a hybrid application

As discussed in section 7.4 a possible alternative to the web application approach was to create a hybrid application. The existing code in the current ELA Mobile application could be used to create a hybrid-application with a cross-platform tool such as PhoneGap (see section 6.2.2.1 for more information about PhoneGap), since jQuery Mobile is supported in PhoneGap (Camden and Matthews 2012). There are however a major obstacle that will make it difficult to achieve this. Since ELA Mobile is also built with the Yii framework that uses PHP and relies on server side processing, this design will not be compatible with the PhoneGap framework (PhoneGap 2015a). This design decision could become a problem if ELA Mobile where to require a feature that is was not available in a web application. This might never become an issue to ELA Mobile since it is a mobile application that does not require access to many advanced functions and API's. It is however more likely that Adcom Molde would use this method to improve ELA Mobile.

This issue was not considered before starting and selecting a development method. This shows that even with detailed and thorough planning it is possible to come across fall pits in SE. This issue will have to be further discussed as a part of the development strategy for ELA Mobile. An alternative to upgrading to the new version of the Yii framework as discussed in 9.6.3 could be to remove it from the solution and instead rely on distributing the mobile application as a hybrid version.

10. Reflections and learning

As reviewed in the methodology in chapter 4, Action Research have a twofold goal. The first goal is to create a learning outcome for participants and add practical value for the organization where the research is conducted. The other goal is to simultaneously build theories and contribute with new theoretical knowledge. Section 10.1 “Reflections”, will describe the learning and practical outcome of the research. 10.2 “Learning” will review the outcome of the research against existing knowledge found in the literature.

10.1 *Reflections*

The dataset from Google Analytics reviewed in section 9.1, shows that the ELA Mobile application is used by a very diverse set of users. There is registered a multitude of platforms, devices and browsers types, many of them are also represented in different version. Similar to the mobile sales statistics in Appendix C the Android and iOS platforms are the most used. However, there is a larger share of iOS devices used in the ELA customer group than what was found in the mobile sales statistics. As discussed on page 110, this was also seen in the ELA customer survey (Figure 7.18). Another deviation from the global sale statistics is that there is a virtually non-existing share of Windows Phone users. Only one user have used ELA Mobile from a Windows Phone in the dataset from Google Analytics (Appendix K.1). With such a low share of users from this platform it is debatable that Adcom Molde should consider having support for this platform in addition to Android and iOS. This issue is not a problem for the current web application, but should a native approach be relevant in the future, the additional cost of development would probably not be worth the added support for this platform.

Table K.3 shows the version distribution for both Android and Apple. At first glance it might seem like the iOS platform is more fragmented than the Android platform since there is a larger number of different version shown in the list. However, when looking at the number of sessions from each version there is a higher concentration of users on the iOS platform that uses the latest versions 9.0.2 and 9.1. This can indicate that iOS are good at upgrading their older devices to the newer versions of their platform as previously seen in Smith (2012) and pxldot (2012) in section 5.2.2.

As a contrast the amount of session for Android is much more distributed across multiple versions and is similar to what is shown in the data from Google (2012a, 2013). This adds to the confirmation that Android is having trouble updating their devices to newer versions and that software fragmentation is a bigger issue on this platform, for the reasons indicated by VisionMobile (2011a, 32) and Whitwam (2011) reviewed in section 5.2.1.

As stated by (Rajapakse 2008a) variations in implementations of standards can be one of the most tiresome types of fragmentation. This is also something that Adcom Molde experienced when developing ELA Mobile. As reviewed in sections 8.2.6 and 9.3, many costly development hours have been spent trying to debug errors that was related to either implementation variations between software vendors or bugs. Even if Android is rumoured to have issues with fragmentation, the majority of errors have been related to issues with iOS. This is also in contrast to the statistics presented by VisionMobile (2011b), where iOS is presented at the platform that requires the least amount of customization. This might be true in other cases, however it has not been the case during the development of ELA Mobile so far.

To make the challenge of these bugs/variations greater, it was also difficult to find relevant information about these errors. Most of the issues have been solvable by using alternative approaches or for instance use JavaScript to do execute the desired task, such as reset a timestamp input for iOS users. However, these issues can also implement braking changes, one of the iOS bugs made it impossible to upgrade to a newer version of the jQuery Mobile framework. This can prevent other bug fixes to be implemented and potentially cause security issues if there are major security errors or exploits in the framework. Nevertheless, the overall experience is that these issues have been few compared to the general development bugs and issues introduced by Adcom developers themselves, related either to the back-end or the mobile application.

Distribution is seen as one of the main disadvantages of the web application method VisionMobile (2011a, 11). Similar to the majority of developers that use the web application method, Adcom Molde also avoided this issue by going “straight to browser” (VisionMobile 2013c). By not using an application store and being fully in-charge of the application distribution and monetization themselves and not having to deal with a third party, Adcom Molde assumes that they have saved both time and money during the development process.

It is unknown how good the user experience is on all of the devices that uses ELA Mobile. Especially for users that are on desktops, since the resolution of the interface is intended for smaller screens. However, since there are also many recurring sessions from PC’s, ELA Mobile seem to be an application that the users feel they can utilize even from desktop computers. This should perhaps be seen as an additional bonus. Since the sessions are recurring and the number of sessions is consistent throughout the timeframe of the Google

Analytics dataset, combined with very little bug reports from the customers during the project indicates that ELA Mobile works well on all utilized platforms and devices.

The stability of the implementation across various OC's reviewed in section 9.1 as well as the positive feedback from the ELA customers in 9.2 indicates that no new cycles of Action Research is necessary (Kai Petersen et al. 2014). There are however some of the issues in ELA Mobile discussed in 9.6 (perhaps especially 9.6.4) that may need to be improved and investigated for future development of ELA Mobile, and that could be grounds for a new action research cycle.

Global Intelligence Alliance (2010, 21), states that web application have a clear cost advantage over native development. This is also the findings in this thesis, even with a careful estimate of development hours in section 9.5.2, native development seems to be an alternative that would have been very costly for Adcom Molde with only 20 customers as their initial target customer group. Even by using the web application approach that is the seemingly cheaper alternative (section 9.5.1) the result of the ELA Mobile development, so far, have provided a negative balance. Nevertheless, even if the result of the ELA Mobile development provide a negative result when viewed isolated, the total product that Adcom Molde can deliver as a package have been greatly improved with the addition of a mobile application and will allow Adcom Molde to be more competitive. As a result Adcom Molde may sell more ELA installations to other customers.

A budget surplus was not the expected outcome for this project, at least not within the first years. The development of a mobile application was viewed as a necessary measure to keep existing customers and secure them as a source of income for the future. Even though the development have been longer than first anticipated, it is considered as a long term investment. John Erik Johnsen (head of the development department) states that in addition to keeping all existing customer during the project period, adding ELA Mobile to the product package have made ELA more interesting for customers and new modules and features have become easier to sell. For instance, have the integration between ELA and Huldt & Lillevik[56] had sales for 50,000 NOK in 2015.

During this research it is very apparent that the changes in the mobile environment is happening very fast and the topic of mobile development with regard to fragmentation and

[56] Huldt & Lillevik (refer to: http://www.huldt-lillevik.no/)

development methods will have to be evaluated on an ongoing basis by practitioners and researchers.

10.2 ***Learning***

With no prior knowledge about mobile development before this research project started, it has been very educational for both the researcher and for Adcom Molde. Exploring and learning about various topics regarding development methods and how they work is very useful considering that mobile development has become such an important market. Although the PC is a platform that Adcom Molde will keep developing for, for the foreseeable future, the portability and functionality of mobile devices makes them very useful tools for both professional and personal use. It might have been more beneficial it the researcher had a more active part in the development of the front-end. However, due to organizational structure and areas of expertise it would have been too time-consuming and costly to divide this task between the development resources.

In addition to the learning outcome, the practical outcome of this research have improved a real world problem in Adcom Molde. A solution that fit the need of the Adcom Molde organization was implemented and a mobile application was developed during the research period. The majority of the ELA customers now use the mobile application on a daily basis and several customer's state that they have an improved overview of their costs, since data registration have become more accurate. In addition, customer's report that their employees are positive to using the mobile application. The board of directors in Adcom Molde is happy with the outcome of the project and it is decided that the development department will start using ELA together with ELA Mobile during January 2016 followed by the rest of the Adcom Molde organization during 2016. Hopefully, several Adcom offices and potential customers in Norway will also be using the application in the years to come.

Appendix A The development department

Appendix A contains a short description of the employees in the development department in Adcom Molde and their role in the company.

Appendix A.1 *John Erik Johnsen*

The head of the department is John Erik Johnsen. He has a background in programming, journalism, sales, marketing, IT-training, IT-consulting, IT-management and production management in newspapers etc. He has worked in Adcom Molde since 2002. From 2006 as the head of the software department. John Erik has designed much of the functionality of ELA Mobile, and also done a lot of the testing.

Appendix A.2 *Arild Kjølseth*

Arild Kjølseth has been working as a system consultant and project manager in Adcom Molde since 2007. He is in charge of software solutions and configurations for many of the Adcom Molde customers. He is in addition, responsible for maintenance and support of all ELA client installations. He has experience with programming in COBOL and was chief technical officer in Tollpost-Globe AS between 1992 and 2000 were he worked with UNIX and Win-NET. In 2000 he started worked as a service manager in Capgemini until 2005 and as Senior Chief Executive Officer in EDB Business Partner Norge AS between 2005 and 2006. His involvement in the ELA Mobile project have been testing, sales support and is in charge of installing and maintenance of all back-end installations.

Appendix A.3 *May Britt Solheim*

May Britt Solheim had several years of experience with sale from IT and construction, before she started working for Adcom Molde in 2006 as an ERP and software sales consultant. She was product and marketing manager for ELA and responsible for solution sales from other ERP software such as Visma, SuperOffice and Aditro (Huldt & Lillevik). In January 2015 she was promoted to sales and marketing manager and in August 2015 she started working as chief executive for Dahl Kontor AS.

Appendix A.4 *Pål Gammelsæter*

Pål Gammelsæter is a software developer and have worked in Adcom Molde since February, 2011. Previously he worked in iTrio AS in Molde where he developed web applications for two years (2009 to 2011). He studied programming in the University in

Bergen and has a bachelor's degree in Informatics. In Adcom Molde he works on developing applications for customers and further development of in-house web applications. He is also in charge of maintenance for the company's Linux-servers and the services that run on these servers. He mainly use PHP, JavaScript, HTML and CSS, but have some experience with Java from School. He had no prior experience with mobile application development before working on this project.

Appendix A.5 ***Daniel Huus – author of this master thesis***

Daniel Huus is a software developer with a bachelor in informatics from Molde University College. He has been working in Adcom since March 2009 and have been in charge of the development of the ELA software since late 2009. In addition to the ELA software he also works on client projects from time to time. His main programming language is Visual Basic .NET and have some experience with Java from School. He has his own web development company, Pixelfusion and has been working with professional web development (HTML, CSS, JavaScript and PHP) since 2007. Like Pål Gammelsæter, he also had no prior knowledge about mobile application development.

Appendix B   The ELA application

The most important feature in ELA is the work orders. The main page of the work order form is shown in Figure B.1. The work orders keep track of the various activities in the company. Material and hour registrations can refer to work orders; this gives the company details about costs involved in each activity. Each work order refer to different cost carriers, such as projects. This information can give details about costs across multiple work orders. It is also possible to refer work orders to specific components and customers in the power plant grid to track maintenance etc.

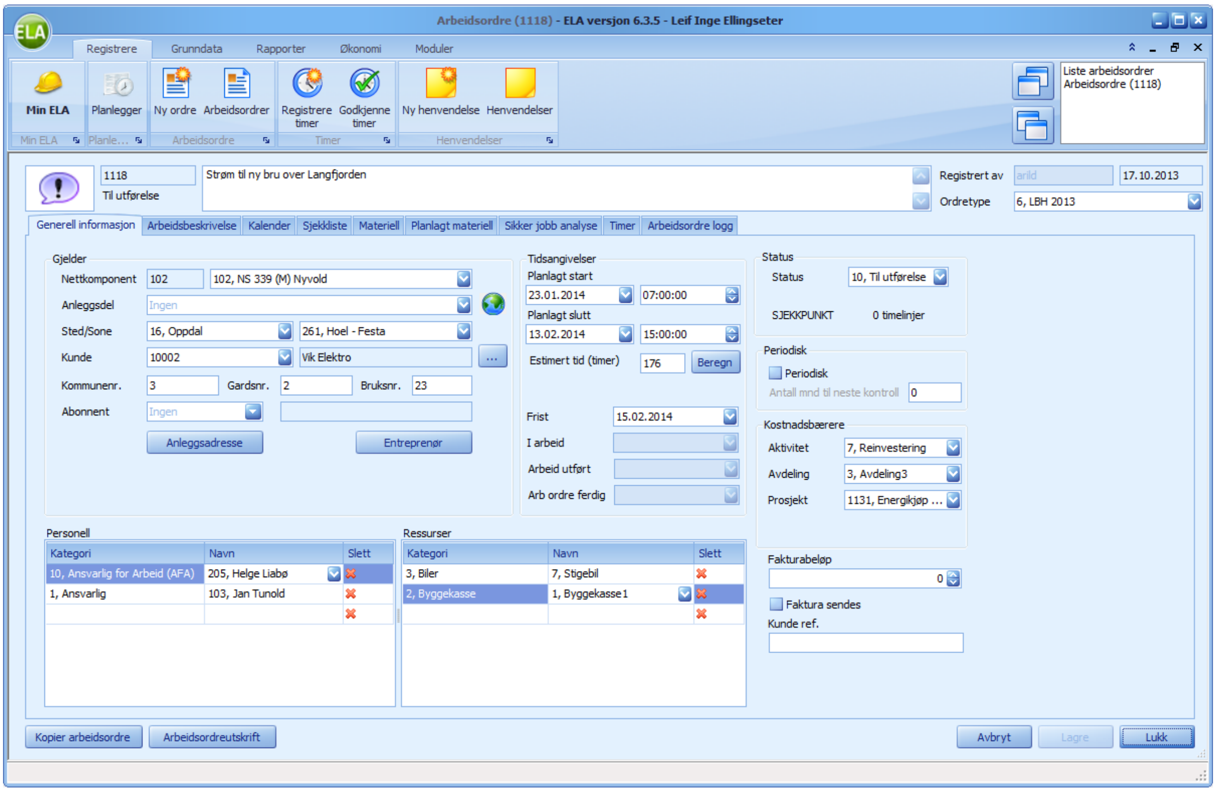


FIGURE B.1: ELA, WORK ORDER FORM (ADCOM MOLDE).

Materials used in repairs, upgrades and new installations are registered on work orders, these costs can then be transferred to accounting software that monitor budgets and sends invoices to customers. Hour registration is one of the most diverse features with ELA customers. Customers have their own specific way of registering hours. ELA try to make the registration as versatile as possible.

When out of office, the workers have to assess risks with the work they are conducting. This is important because much of the work involves life-threatening tasks. If the workers need to risk assess a job, they have to sign papers saying that they have done just so. If anything happens to them and they are injured, an investigation of the incident will reveal if the worker had assessed the risk before starting the job. If the worker has failed to do so, an insurance claim can be disapproved. The company can then face fines and charges for reckless behaviour and poor environmental, health and safety (EHS) routines. Custom EHS forms (Figure B.2) can be created in ELA based on the industry standard from Rasjonell Elektrisk Nettvirksomhet[57] (REN).

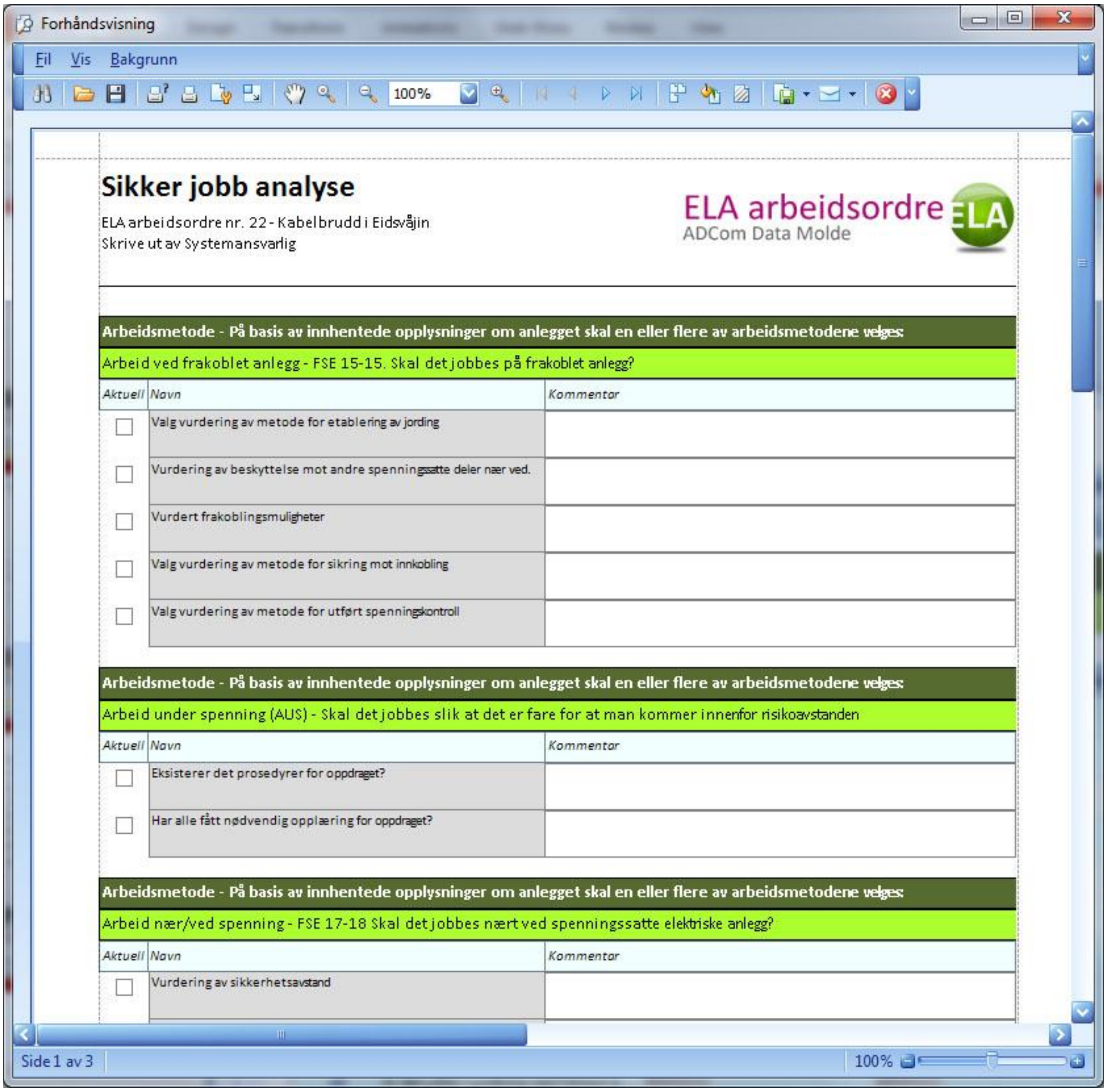


FIGURE B.2: ELA, CUSTOM RISK ASSESSMENT FORM (ADCOM MOLDE).

One of the newest and most comprehensive feature in ELA is the planning software (Figure B.3). This allows managers to create plans for work orders in a calendar for each

---

[57] Rasjonell Elektrisk Nettvirksomhet (REN) (refer to: http://www.ren.no/)

employee. It is also possible to plan and delegate tools and equipment to work orders in the calendar. Integration with Microsoft Exchange synchronizes appointments to MS Outlook calendars and allows the same information to be available on mobile devices. Each user can also create and update their own calendar in their own personal view called “My ELA” (Figure B.4), here they also get an overview of all work orders that they are connected to.

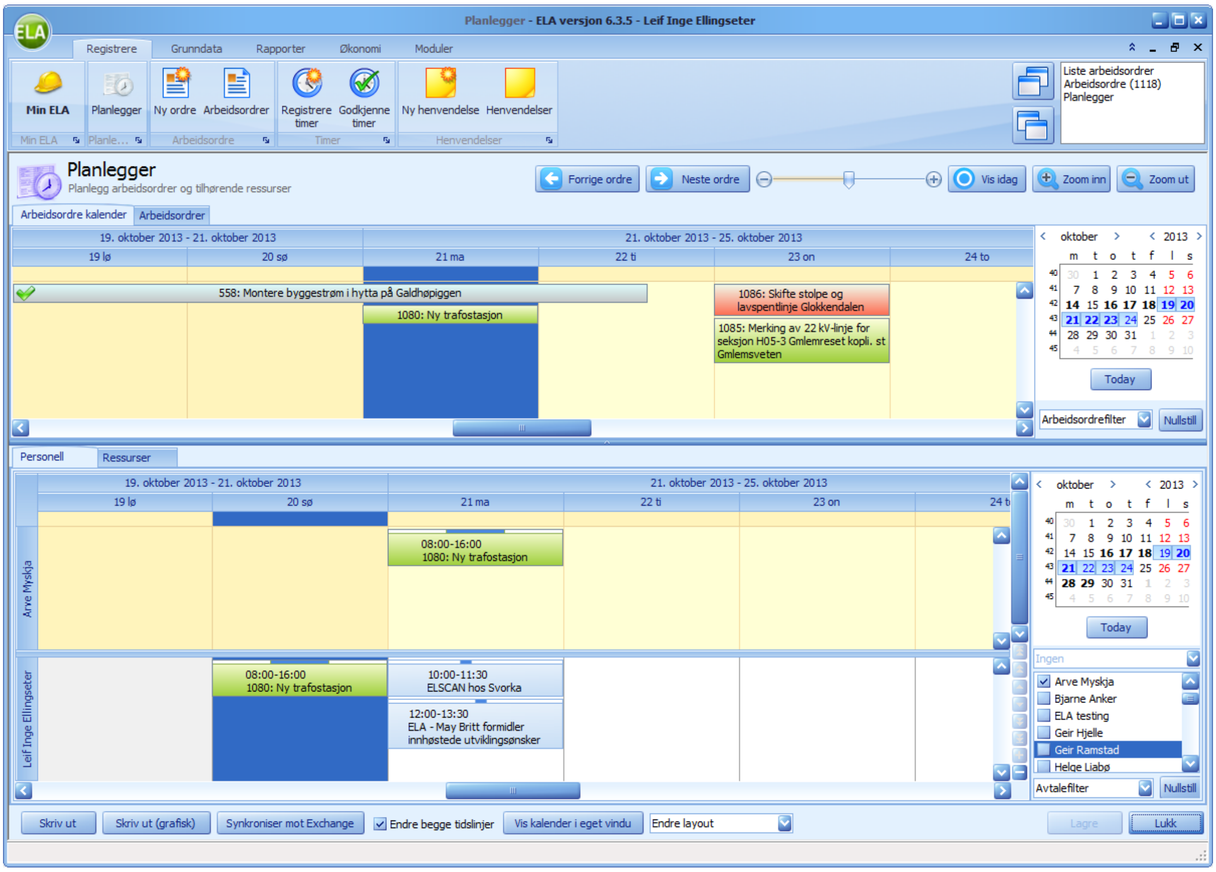


FIGURE B.3: ELA, CALENDAR PLANNER (ADCOM MOLDE).

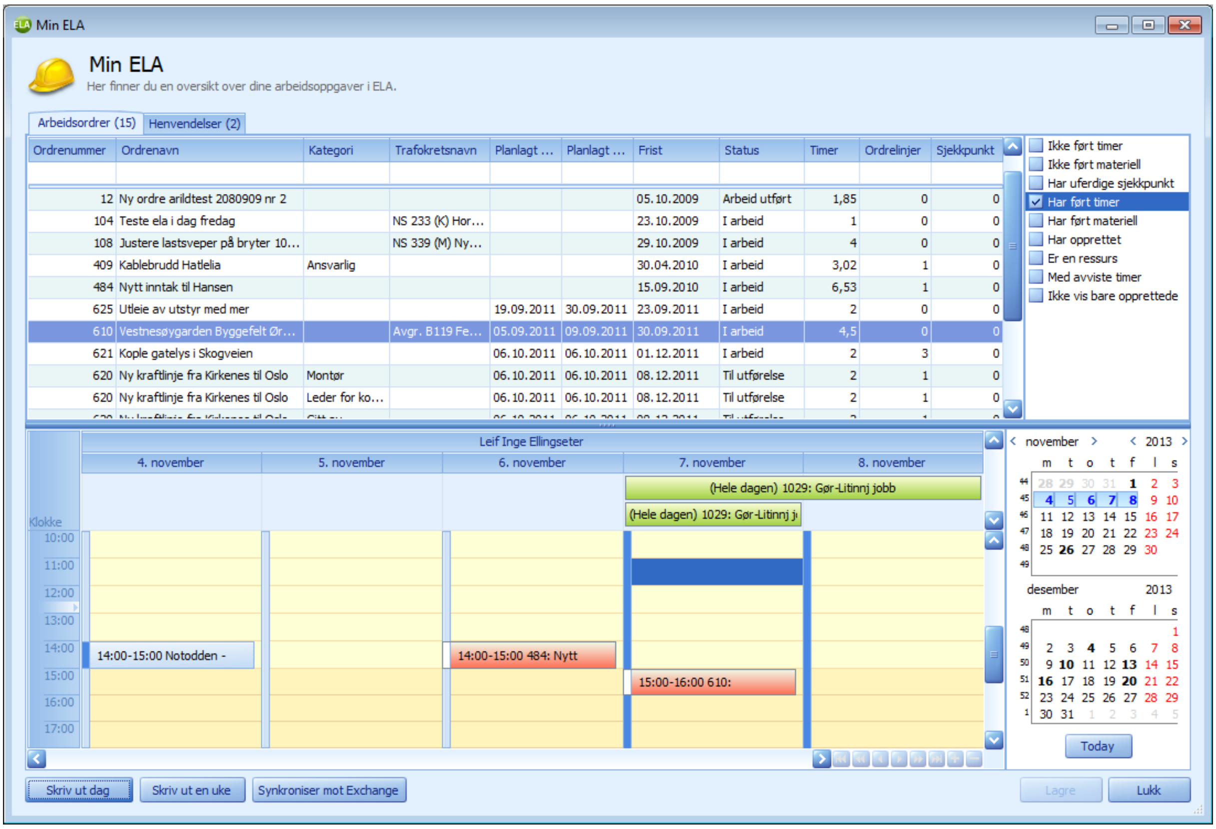


FIGURE B.4: ELA, PERSONAL VIEW OF JOBS AND CALENDAR APPOINTMENTS CONNECTED TO THE CURRENT USERS (ADCOM MOLDE).

Appendix C Mobile market statistics

Operating system	2008 (average)	2009 (average)	2010 (average)	Q1 2011	Q2 2011	Q3 2011	Q4 2011	Q1 2012	Q2 2012	Q3 2012	Q4 2012	Q1 2013	Q2 2013
Symbian	52.4	46.9	37.6	27.4	22.1	16.9	11.7	8.6	5.9	2.6	1.2	0.6	0.3
Research In Motion	16.6	19.9	16.0	12.9	11.7	11.0	8.8	6.9	5.2	5.3	3.5	3.0	2.7
iPhone OS	8.2	14.4	15.7	16.8	18.2	15.0	23.8	22.9	18.8	13.9	20.9	18.2	14.2
Microsoft	11.8	8.7	4.2	3.6	1.6	1.5	1.9	1.9	2.7	2.4	3.0	2.9	3.3
Linux	7.6	4.7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Android	0.5	3.9	22.7	36.0	43.4	52.5	50.9	56.1	64.1	72.4	69.7	74.4	79.0
WebOS	0.0	0.7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Bada	0.0	0.0	0.0	0.0	1.9	2.2	2.1	2.7	2.7	3.0	1.3	0.7	0.4
Other OSs	2.9	0.6	3.8	3.3	1.0	0.9	0.8	0.9	0.6	0.4	0.3	0.3	0.2

Table C.1: Mobile smartphones market shares based on sales, listed by operating system from 2008 to Q2 2013 (Gartner 2010, 2011d, a, b, c, 2012d, a, b, c, 2013a, b).

Appendix C.1 *2008*

Symbian had the majority of the sales in 2008 with over half of the market, at 52.4%. The nearest competitor was Research In Motion with 16.6% and Microsoft Windows Mobile with 11.8% of the market.

Appendix C.2 *2009*

In 2009 Symbian dropped 5,5% and ended up on 46.9%, but it was still the best selling operating system with Research In Motion gaining 3,3% as the second most selling operating system. Microsoft Windows Mobile dropped to 8.7%. iPhone almost doubled their market share going from 8.2% to 14.4% and Android got almost an 800% higher market share going from 0.5% to 3.9%.

Appendix C.3 *2010*

Symbian was still the most selling in 2010, but continued loosing market shares to its competitors dropping an additional 9%. Research In Motion was still second, but also dropped 3.9% almost on the same market share as iPhone with its 15.7%. Microsoft Windows Mobile loses over half its sales from 2009. Android continues to gain market shares and had 22.7% of the market, increasing 4540% from 2008.

Appendix C.4 ***2011***

Year 2011 was very interesting where a lot changed for many of the competitors. Symbian dropped from an average of 37.6% in 2010 to 11.7% in Q4 2011, over 31% of their market shares and went from being the most selling operating system to a third place. The same happened for Microsoft Windows Mobile that dropped from 4.2% to 1.9% in Q4 2011 (lowest in Q3 of 1.5%). Bada managed to surpass Microsoft in with 0.3% in Q4 2011. Android continued to gain market shares, and reached over 50% in market shares, but with a drop of 2.6% from 52.5% in Q3 2011 to 50.9% in Q4 2011. Android increases from 22.7% to 45.7% (on average in 2011) and becomes the bestselling operating system in only four years with iOS as the closest competitor. iOS reached a peak of 23.8% in Q4 2011. Research In Motion continued to drop ending up at 8.8% percent in Q4 2011.

Appendix C.5 ***2012***

In year 2012, the same trends as in 2011 continued leaving Android on 69.7% in Q4 2011 with a peak of 72.3% in Q3 2011. iOS started good with 22.9% and ending on 20.9%, but with the lowest market share since 2008 with 13.9% in Q3 2012. Symbian, Research In Motion and Bada all loose market shares ending up at respectively 1.2%, 3.5% and 1.3% in Q4 2012. Microsoft gains some market shares and ends on 3% in Q4 2012.

Appendix C.6 ***2013***

Looking at Q2 2013 it seems quite clear according to the statistical trend that at least Symbian and Research In Motion have lost to their competitors. Research In Motion is still in fourth place with 2.7%, but if the current trend continues they might not be able to recover from it. Bada also seem to be losing a lot of momentum ending at 0.4% in Q2 2013. Android has become the major market shareholder with 79% with iOS on second with 14.2% and Microsoft in third with 3.3%.

Appendix D Hardware fragmentation

Screen size and input types also vary between different models. In Figure D.1 there is an Android phone with a big screen, but no keyboard to input text. This phone uses a touch keyboard displayed on the screen itself, when needed. A clever way to optimize the screen size, but reduces the available GUI (graphical user interface) when displaying the keyboard.

FIGURE D.1 SAMSUNG GALAXY NOTE II (SAMSUNG 2012)

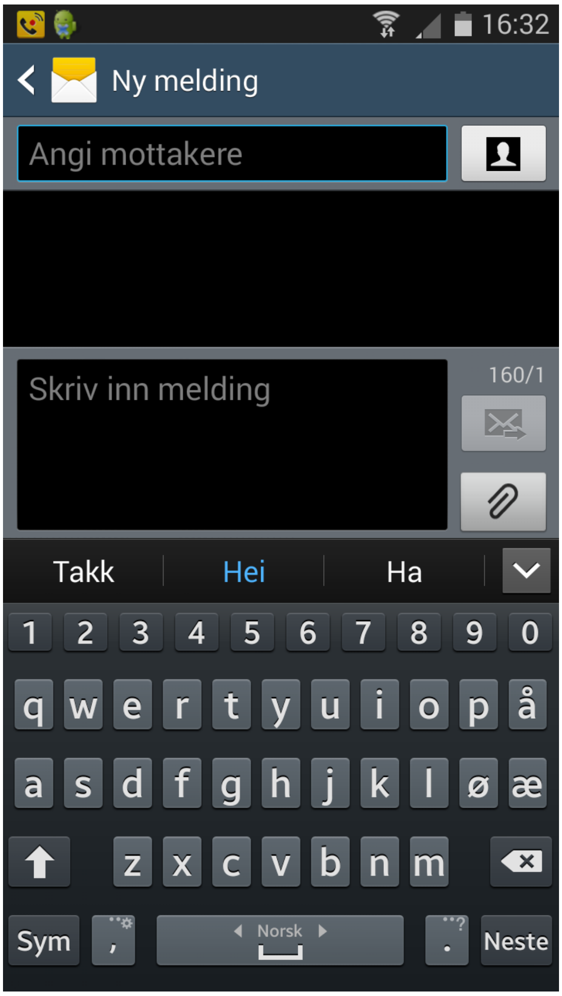


FIGURE D.2: THE STANDARD KEYBOARD OF THE SAMSUNG NOTE 2 ANDROID SMARTPHONE.

Figure D.2 shows the layout of the on screen keyboard when the phone is vertical. The keyboard covers about 50% of the available screen resolution. There are also room to read the SMS history and a back button is available on the top of the screen.

FIGURE D.3: THE STANDARD HORIZONTAL KEYBOARD FOR SAMSUNG NOTE 2 ANDROID SMARTPHONES

When the orientation of the phone is horizontal like the one in Figure D.3, the keyboard take up a lot of space on the screen, approximately 59%. This leaves no room to display the SMS history or the back button.

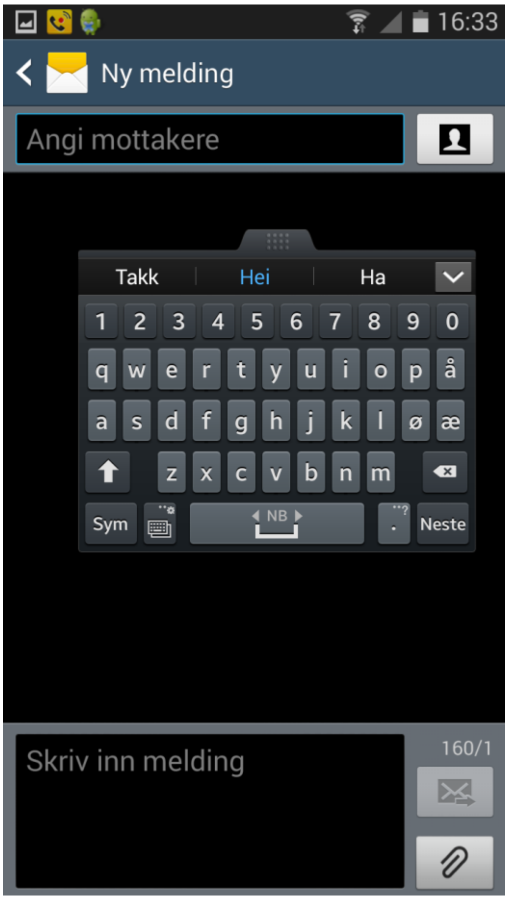


FIGURE D.4: FLOATING LAYOUT FOR THE SAMSUNG NOTE 2 ANDROID SMARTPHONE.

Another keyboard layout for the Samsung Note 2 Android phone is the “floating” layout shown in Figure D.4. Here it is possible to move the keyboard around on the screen. The keyboard is displayed when the phone is held both vertically and horizontally.

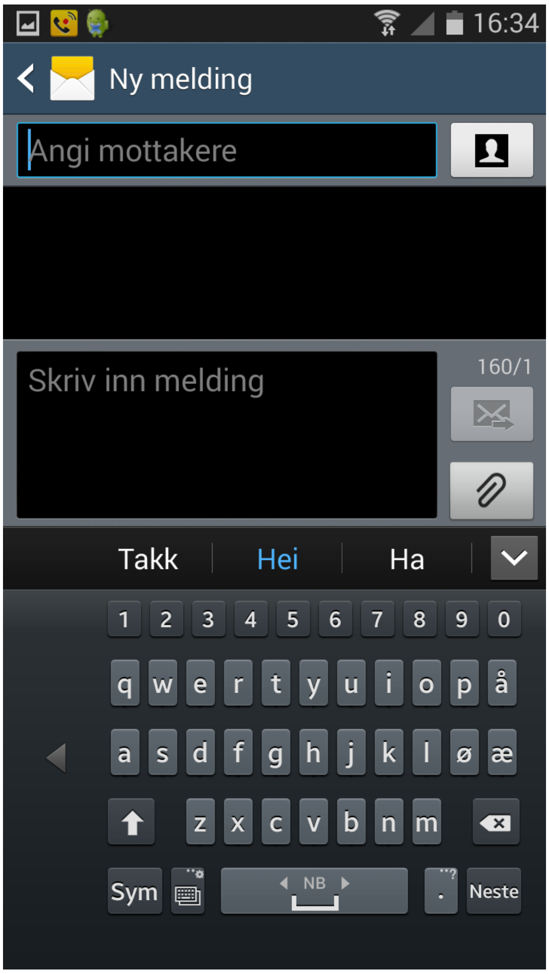


FIGURE D.5: ONE HAND KEYBOARD LAYOUT (RIGHT ARM).

Figure D.5 shows a one-handed keyboard layout. Phones with large screens can be hard to use with only one hand. The keyboard shifts either to the right or to the left making it easier to reach all buttons with one hand, depending on which hand is used. Figure D.6 shows a phone with a smaller screen than the one in Figure D.5, but it has a QWERTY keyboard available in the front of the phone. This means that the GUI will always have the same available space and will not be interfered by a keyboard showing up on the screen, the downside here is that the buttons and keyboard take up a lot of space, making the possible viewable area much smaller.

Figure D.6: Motorola-Milestone with Qwerty-keyboard
http://www.android.com/devices/detail/motorola-milestone-plus

Figure D.6 shows a phone with a big screen and a large QWERTY keyboard that physically slides out from behind the phone, if needed. This combines the big screen with the full keyboard, but when using the keyboard, the orientation and aspect ratio on the screen changes, making the GUI wider than on the other models. This screen also offers the same on-screen keyboard as the phone in 0 if the physical keyboard is not used.

Figure D.7: A phone with an optional hardware keyboard. Source: http://www.android.com/devices/detail/htc-desire-z

Appendix E Mobile browser shares

Appendix E.1 *Overall market shares*

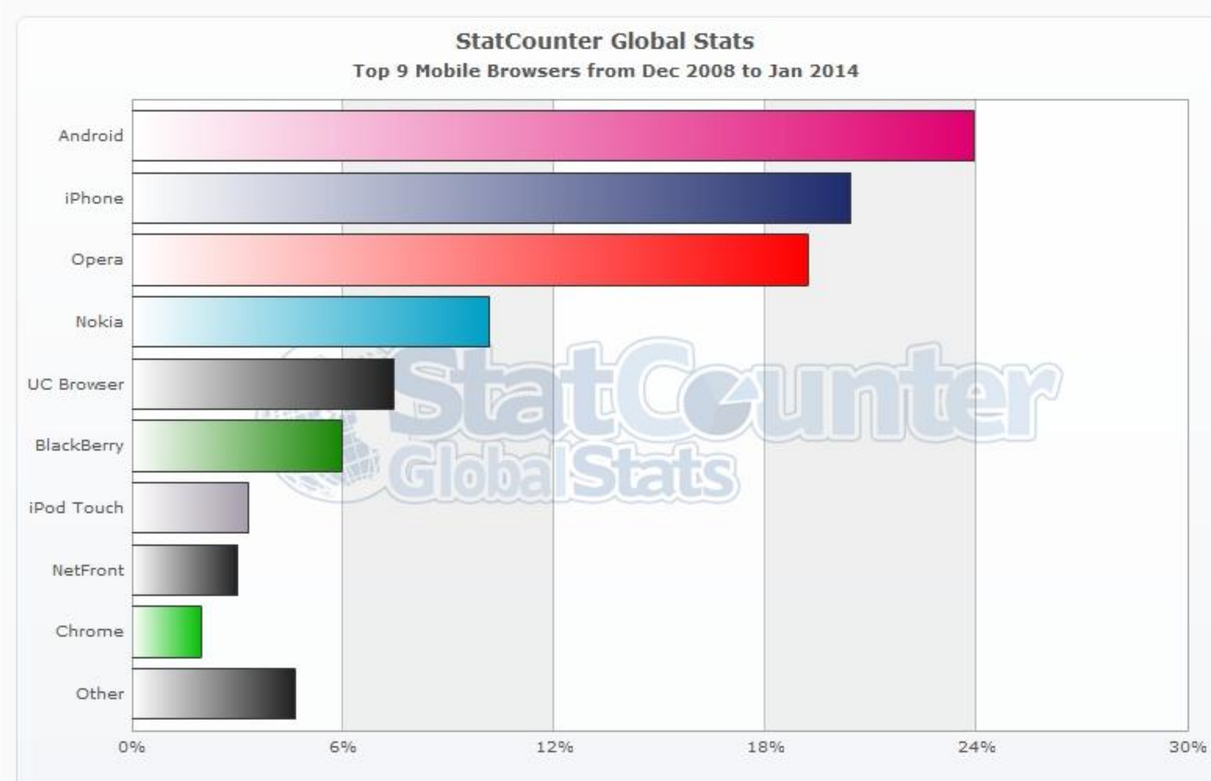


FIGURE E.1 AVERAGE MOBILE WEB BROWSER MARKET SHARES FROM DECEMBER 2008 TO JANUARY 2014 (STATSCOUNTER 2014B).

The graph in Figure E.1 shows which mobile browsers had the overall biggest market share between December 2008 and January 2014. Android, iPhone, Opera and Nokia, UC Browser and Blackberry have the six largest market shares. They each had ca. 6% or more of the mobile market, during this period. Figure E.2 shows how the market shares were during the beginning of this time-period.

Appendix E.2 *In the beginning - 2008*

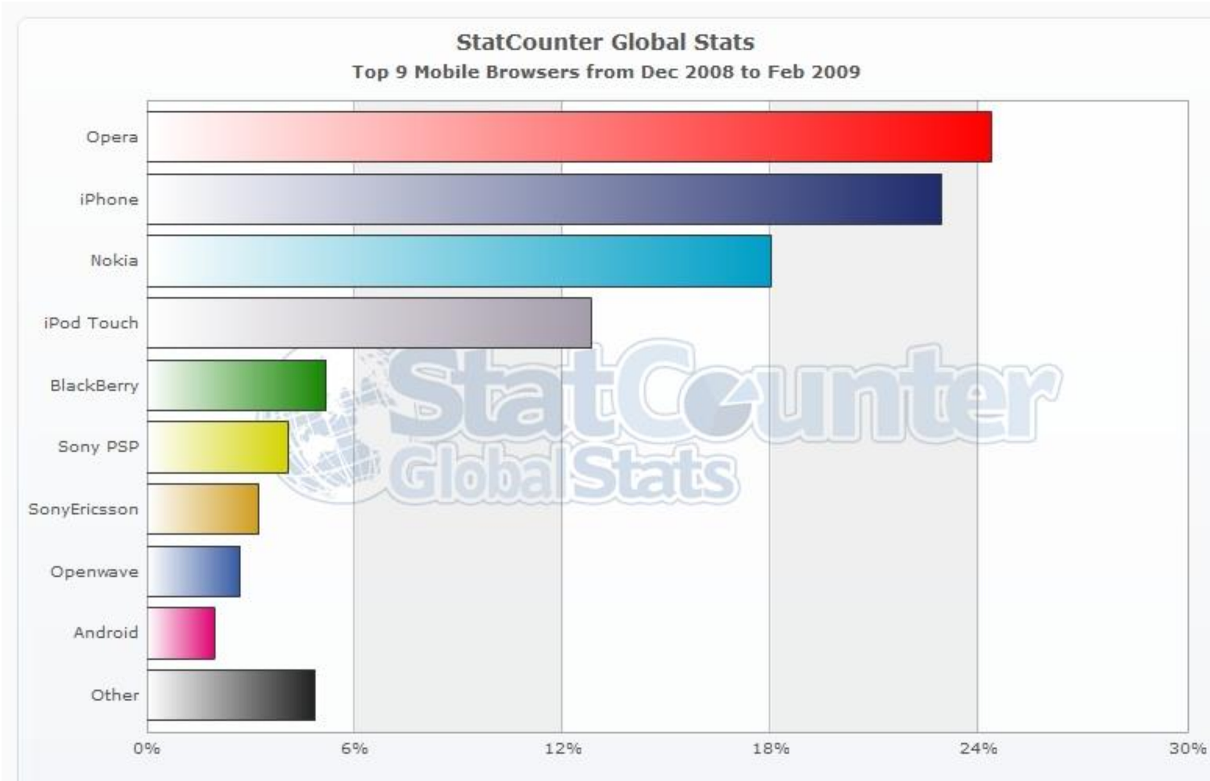


FIGURE E.2: AVERAGE MOBILE WEB BROWSER MARKET SHARES FROM DECEMBER 2008 TO FEBRUARY 2009 (STATSCOUNTER 2014A).

Figure E.2 shows that the market shares in Q4 2008 and Q1 2009 where a lot different from the overall statistics in Figure E.1. Most noticeable from these to graphs is Android who goes from one of the smallest shares with 2% market share, not including the “other” category, to becoming the major browser. Concurrent with the sales statistics in Table C.1. BlackBerry who had a market share of 5% has increased their overall share with 1%. Also noticeable is the iPod going from over 12% to an overall of around 3%. The Sony PSP, Sony Ericsson and Openwave has disappeared from the chart and become one of the browsers in the “other” category.

Appendix E.3 ***Market shares from October 2013 to January 2014***

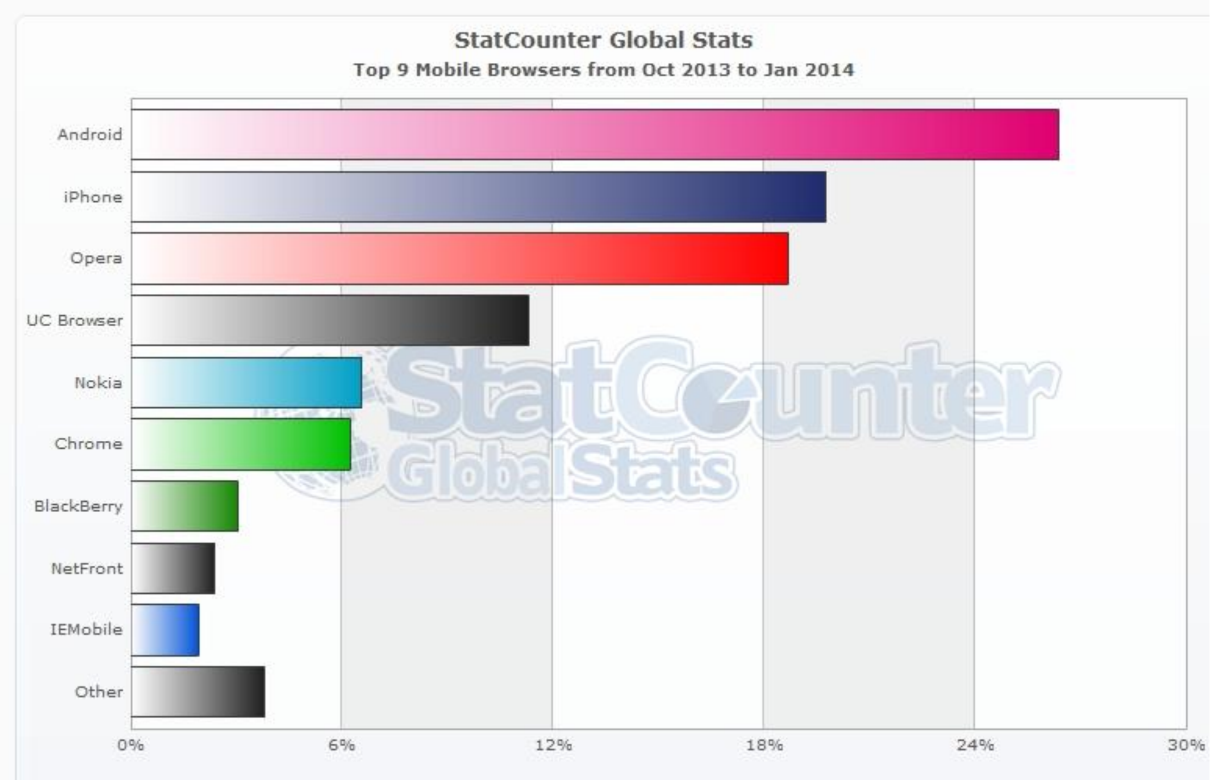


FIGURE E.3: AVERAGE MOBILE WEB BROWSER MARKET SHARES FROM OCTOBER 2013 TO JANUARY 2014 (STATSCOUNTER 2014D).

Q4 2013 and Q1 2014 seen in Figure E.3 is a completely different state than in Q4 2008 and Q1 2009 in Figure E.2. Opera, Android, iPhone and UC Browser has become the four browsers dominating the market. Nokia fell behind with only 7%, almost caught up by Google Chrome. BlackBerry had only 3% compared with an average of 6% since 2008.

The third most popular browser, Opera, has a wide range of supported platforms. Opera mobile 12 supports Android, Symbian/S60, Windows mobile (10), Maemo (labs) and MeeGo (labs). The more lightweight version of the browser, Opera mobile mini (v6.5) supports Java phones, iOS (7), Android, Symbian/S60, BlackBerry and Windows Mobile (5.1) (Opera 2012).

The UC browser become very popular with an average market share of 8% since 2008 and with a market share of 11% it became the fourth largest mobile browser.

> UC Browser is a leading mobile internet browser with more than 400 million users across more than 150 countries and regions. This browser is currently available in 11 different languages on all major operating platforms (UCWeb 2014).

IPod Touch went from 13% to becoming one of the browser in the “other” category. Another browser that emerged was NetFront which states to have support for many different platforms as well. Such as Android, Symbian (Series 60 and UIQ) and Windows Mobile (Access 2014).

Appendix E.4 ***Mobile browser market shares summary***

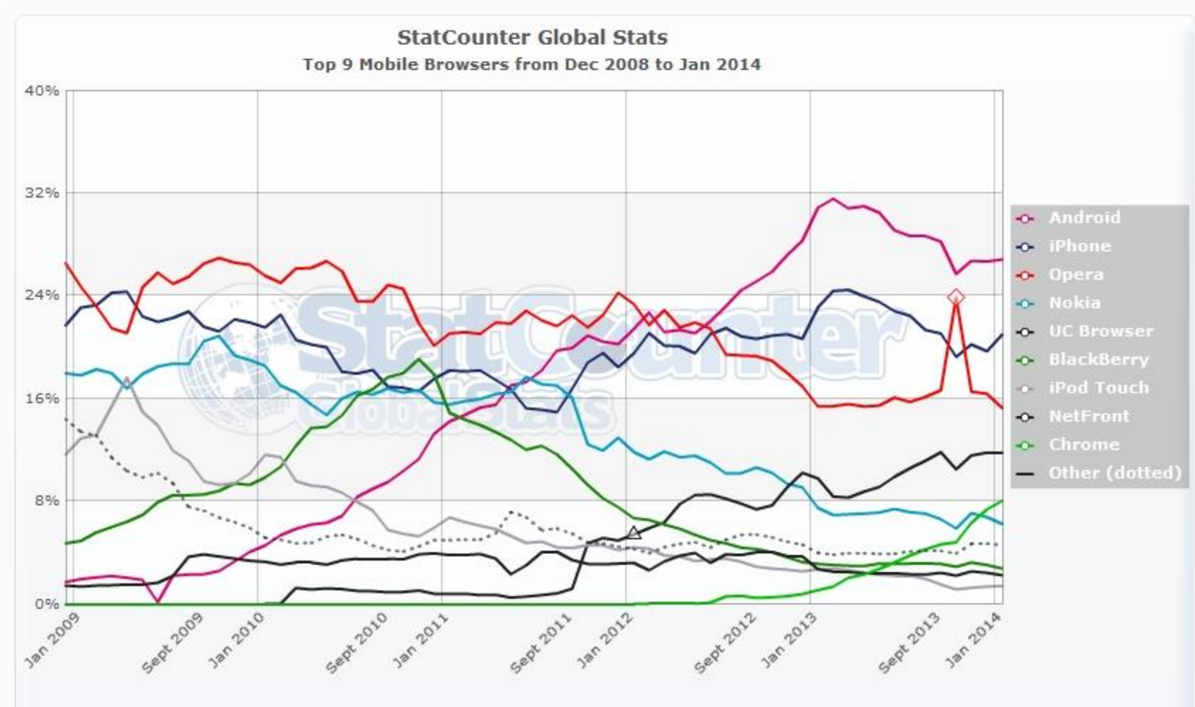


FIGURE E.4: MOBILE WEB BROWSER STATISTICS FROM DECEMBER 2008 TO JANUARY 2014 (STATSCOUNTER 2014C).

Figure E.4 shows that Opera had the majority of the market except during February to May of 2009, where iPhone had a bigger share. In February 2012 Opera was surpassed by Android and in June 2012, iPhone surpassed it once again. Opera had a spike in usage October 2013 due to bot-related traffic. Android and Blackberry had roughly the same market share growth from December 2008 up until October 2010, after this Blackberry’s market share plummeted from a peak of about 19% to ca. 3% January 2014 whereas Android continued to rise, reaching a peak of almost 32% in January 2013 and 26.8% in January 2014. Apple’s IPod touch had a big market share with a peak of almost 18% in Q1 2009 but has since then fallen to ca. 2% in January 2014. Much of the reason for this might be the introduction of smartphones with similar functionality the last couple of years. With the decline of Operas popularity and the increase in popularity of the UC Browser, it can appear that Opera will be in fourth place during 2014, after being number one for many years.

Appendix F HTML5

This appendix will provide details about some the important new features in HTML5.

Appendix F.1 *Device adaption*

The new web standard aims to work better on mobile devices. Since the hardware on these devices are so different from a regular computer such as screen size, keyboard types etc. W3 has developed an API that will enable developers to access information about the device viewing the webpage. This makes is possible to tailor the web site to fit the current device. For instance, a web application can then choose to exclude camera functionality in the application for devices unable to take photographs. This feature has currently none or very limited implementation in current browsers (W3C 2012b).

Appendix F.2 *Graphics handling*

HTML5 will support something called Scalable Vector Graphics (SVG), which can create two-dimensional vector graphics. The graphics are geometric shapes that can be zoomed. This makes it possible to alter the graphics to fit the device that views it, without compromising the visual quality. In addition, the vectors support animation to create nice effects. For instance in a user interface.

On mobile devices, the animation feature requires use with care since they need some processing power. Graphics intensive websites might be best suited for desktop computers, which has more computational power than mobile devices.

A new element called "<canvas>" also enables a 2D programmatic interface. A canvas can programmatically create drawings on the fly with JavaScript. Under development is also an API called WebGL[58] to support 3D graphics. This API is not a part of the W3C standard, but there is support for mobile devices as well.

Appendix F.3 *Font support*

Fonts has been an issue on most webpages for a long time. Before HTML5, it was not possible to use any types of fonts on websites, because the user viewing your website did not necessarily have the same font on their computer. With HTML5, websites can use any font type they want. Web Open Font Format (WOFF[59]) downloads fonts to the clients through

---

[58] WebGL (refer to: http://www.khronos.org/webgl)

[59] Web Open Font Format (refer to: http://www.w3.org/TR/WOFF)

style sheets so that the font will display properly on all devices viewing your site (W3C 2012b).

Appendix F.4 *Multimedia support*

Today multimedia is a big part of everyday life; HTML5 also adds support for use of audio and video on websites. Up until now developers has had to use plug-ins to embed video or audio on their websites, such as Flash[60]. It is not only playback of these media types, which HTML5 aims to support. They also wish to support capturing of media. Most new mobile devices today have a built-in camera and can record audio through the phones microphone. HTML Media Capture[61] works on creating an API that can access these interfaces. This API can for instance enable users to upload and share photos directly in web applications. In conjunction with the new graphics handling, HTML5 also opens for the possibility to edit videos and audio (W3C 2012b).

Appendix F.5 *Forms and data validation*

With mobile devices accessing the internet, HTML5 gets wider support for form input. Filling out forms on a mobile device can be a difficult and tedious task on mobile devices. New features will allow forms to access input specific data to display suggestions to the user for instance (telephone numbers, e-mails, contact names etc.). Native device form controls such as date and time for easier form input.

Another feature is validation of forms. Before the users sends the form to the server, the web application validates the users input. This reduces the need for server side communication, which is important on mobile devices where use of bandwidth can be costly (W3C 2012b).

Appendix F.6 *User interactions*

As a part of HTML5s broadened support for mobile devices, HTML5 support more ways of interacting with the web applications. The Document Object Model (DOM) have incorporated touch interactions events. This means that developers can create web applications that respond to hand gestures from users. Together with animations, they can

[60] Flash (refer to: http://www.adobe.com/products/flashplayer.html)

[61] HTML Media Capture (refer to: http://www.w3.org/TR/html-media-capture/)

create nice and intuitive applications. Phone vibrations and sound notifications are also among the new features, but these are still in the early stages of development (W3C 2012b).

Appendix F.7 *Data storage*

HTML5 offers two new methods of storing simple data for use in web applications. One is the "localStorage" that can store data for as long as the application needs it. In addition, another called "sessionStorrage" that only stores the data for the current user session. Most browsers have integrated support for these two methods today. Other features include reading and writing files that are stored on the phone and a more advanced database type called "Indexed Database API" where the application can query and update records, but these APIs are still in the early stages of development (W3C 2012b).

Appendix F.8 *Personal information management*

Most phones ship with an integrated calendar and address book. With new APIs from W3C, these will be accessible from web applications, but these are in still in development and the implementation of these API's are experimental (W3C 2012b).

Appendix F.9 *Sensors and hardware*

Mobile phones ship with a lot of different hardware functionality that is unique to this device type such as GPS (Global Positioning System), Accelerometer, battery, camera and a variety of sensors. HTML5 will be able to access information from these devices through APIs implemented by the mobile browsers. At the time of writing only the GeoLocation API is ready and implemented (GPS), and work is already being on a version two of this API. Other types of APIs (camera, microphone, motion etc.) are still not ready and is in the early and experimental stages at the time of writing (W3C 2012b).

Appendix F.10 *Communication*

HTML5 will feature APIs that can use e-mail, SMS and MMS features from a web application. There will also be possible to communicate between web applications on the mobile phone. However, there are limited to none implementations of these features today (W3C 2012b).

Appendix F.11 ***Availability***

New web applications using HTML5 can make use of the Application cache, this feature stores elements in the web site on the mobile browser making it available even when you are offline with your mobile device. This manifest can contain images, and other files necessary in order for the page to function offline (W3C 2012b).

W3C Widgets is a feature that allows developers to distribute the mobile web application with a digital signature. This widget packages in to a ZIP file and installs on the mobile devices. The zip file uploads to the mobile device and installed as a local "application". Users can use these applications without connecting to the internet.

Appendix F.12 ***Performance***

One big drawback with the web applications is their performance and according to VisionMobile (2013c) is HTML5 and JavaScript by its own design slower than native since native (compiled) code is faster to render than scripts running in a browser. A web application does not provide the smooth user interface that a native application does. There are however some things that can be done with this. CPUs are getting faster and JavaScript compilers are improving. According to W3C (2012b) will HTML5 provide threading that allows developers to create background threads to do all of the heavy computing while main thread stays responsive. Also while following the best practices and keeping the web application to a minimum with optimized code will help the application to perform better. With HTML5's new navigation hooks, it is possible to measure accurate load times. This feature measures the actual time it takes to load a page. Up until now, this has not been possible with HTML.

Appendix F.13 ***HTML5 developer surveys***

0 lists two surveys conducted with developers to investigate what aspects of HTML5 they were satisfied and dissatisfied with.

Appendix F.13.1 **Appcelerator survey (2012)**

> Appcelerator and IDC surveyed 5,526 Appcelerator Titanium developers from August 22-28, 2012 on their perceptions about current debates in mobile, social, and the cloud as well as their development priorities. This constitutes the world's largest mobile developer survey ever conducted to date and reflects the tremendous growth of the Appcelerator ecosystem. (Appcelerator 2012a).

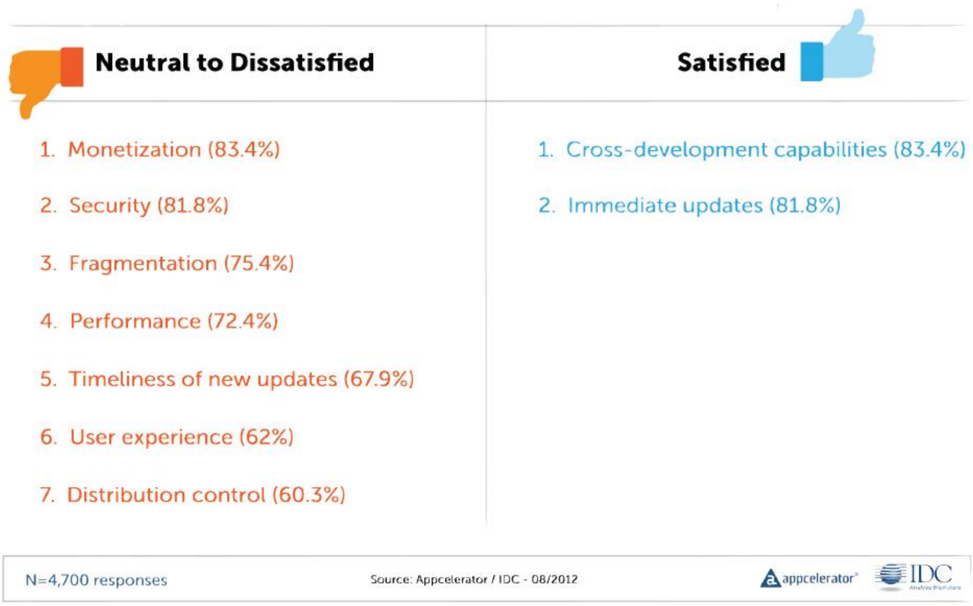


FIGURE F.1: HTML5 POPULARITY WITH DEVELOPERS. (APPCELERATOR 2012A)

In Figure F.1, we can see the ratings that developers gave HTML5. The thing that developers are most satisfied with is the cross-platform capabilities and that the application is fast to update. There are however far more points that the developers are neutral to dissatisfied with. The Appcelerator survey explains this dissatisfaction with the promised features of HTML5. Most of these responses have to do with other things than the HTML5 standard itself not including security, update frequency and user experience. Security is however a problem that the HTML5 standard have today. It is very hard to secure the data stored in a HTML5 application, making it little attractive for companies that need to secure their business data (Appcelerator 2012a).

Even though developers are satisfied with the cross-platform capabilities with HTML5, the large dissatisfaction with fragmentation comes from the various browser implementations of HTML5 across vendors.

Appendix F.13.2 **VisionMobile survey (2013)**

The following are based on VisionMobile's recent survey of 3,460 developers across 95 countries, with a balanced sample across North America, Europe and Asia, plus developer interviews and research insights (VisionMobile 2013a).

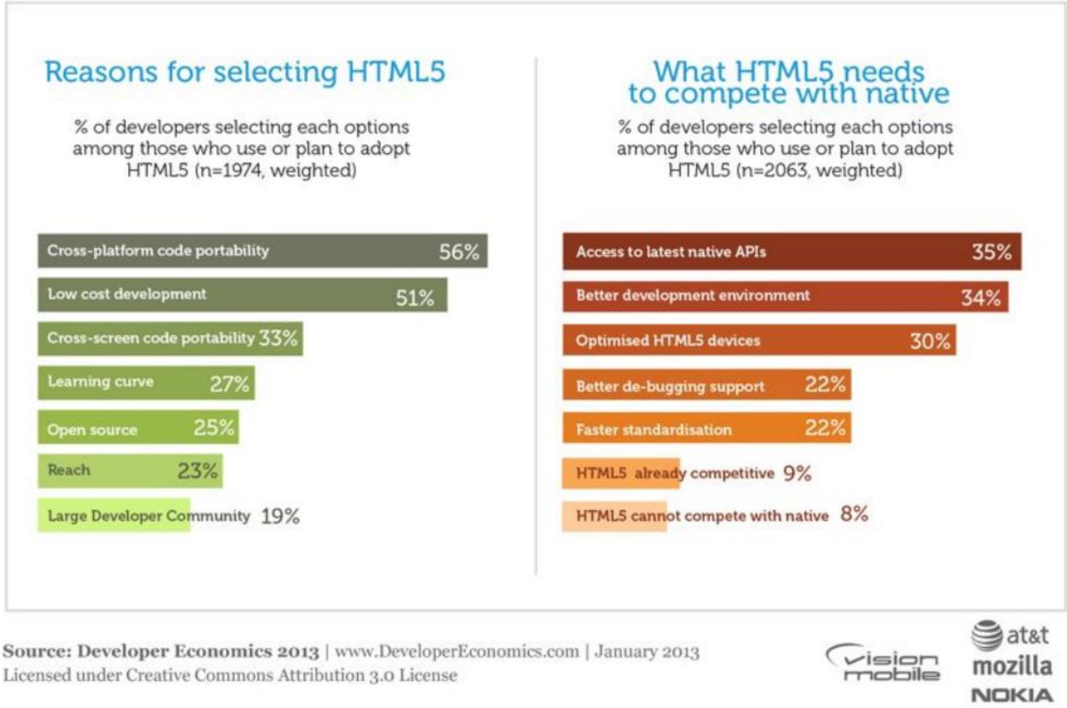


FIGURE F.2: HTML5 DEVELOPER PROS AND CONS (VISIONMOBILE 2013A).

Figure F.2 shows developers answers to what are the best reasons for selecting HTML5 when developing mobile applications. The top argument for selecting HTML5 is its cross-platform code portability (56%) and low development costs (51%). VisionMobile also asked about what HTML5 needs to be able to compete with native. Thirty-five percent answered access to the latest native APIs, but platform vendors are always ahead of platform tools and browser vendors since they create the device features and interfaces. Thirty-four percent answered that HTML5 needs a better development environment and better support for de-bugging (22%). 30% answered that they want optimized HTML5 devices, but developers think that access to the latest native APIs (35%) are more important. VisionMobile also states that because of this HTML proponents (like Facebook, Mozilla and Google) should also focus on cross-platform tools and development environments on the same level as their full platform focus. Important to notice is that 9% thinks that HTML5 already is competitive while 8% thinks that HTML5 can't compete with native at all (VisionMobile 2013a).

Appendix G HTML5 support in mobile browsers

The website www.html5test.com contains a summary of mobile and tablet browser tests. Each browser can get a maximum score of 500 points (500 different tests) and the results changes continuously as new versions of browsers are released (Leenheer 2012). This section is not an in-depth analysis of the various HTML5 features, but is instead a review of the overall scores from www.html5test.com and is just an indicator of overall html support. When developing a web application, developers should look more closely on each feature they wish to use. There are several ways to analyse feature support, some of them include:

- http://caniuse.com
- http://mobilehtml5.org

Appendix G.1 ***Mobile browsers***

current

Firefox Mobile 10	*Multiple platforms*	315	9
iOS 5	Apple iPhone and iPod Touch	305	9
Opera Mobile 11.50	*Multiple platforms*	286	9
MeeGo/Harmattan	Nokia N9 and N950	271	14
BlackBerry OS 7	BlackBerry Bold 9900 and others	266	10
Android 4.0	Samsung Galaxy Nexus	256	10
Bada 2.0	Samsung Wave and others	251	9
webOS 2.2	Palm Pre 2 and HP Pre 3	201	5
Android 2.3	Google Nexus S and others	182	6
Windows Phone 7.5 (Mango)	Samsung Omnia W, LG E906 and others	141	1

development or beta

Chrome Beta	*All Android 4 devices*	343	10

older

Firefox Mobile 9	*Multiple platforms*	313	9
Opera Mobile 11.10	*Multiple platforms*	270	9
Firefox Mobile 6	*Multiple platforms*	285	9
BlackBerry OS 6	BlackBerry Torch and others	258	10
Opera Mobile 11	*Multiple platforms*	254	7
iOS 4.2 & 4.3	Apple iPhone and iPod Touch	215	7
iOS 4.0 & 4.1	Apple iPhone and iPod Touch	182	7
Android 2.2	Motorola Atrix 4G, Samsung Galaxy Pro and others	182	0
webOS 2.1	Palm Pre Plus and Pre 2	160	5
Android 2.1	Google Nexus One, Motorola Droid X, HTC Desire and others	151	0
Bada 1.0	Samsung Wave and others	144	0
webOS 1.4	Palm Pre, Pixi and Pixi Plus	130	5
Windows Phone 7	HTC HD 7, LG Optimus 7, Samsung Omnia 7 and others	25	0

TABLE G.1: MOBILE BROWSER SUPPORT (24.02.2012). COLUMN NR.3 IS SCORE AND COLUMN NR.4 IS BONUS.

Table G.1 shows that the browser with widest support for HTML5 is Firefox Mobile, with 315 points and with the iOS 5 browser right behind with 305 points. Chrome was still in beta and not released yet, but had 343 points and would be even better equipped than the current leader when it was released.

current

		Score	Bonus
Opera Mobile 12.00 »	*Multiple platforms*	354	11
Firefox Mobile 10 »	*Multiple platforms*	315	9
iOS 5.0 & 5.1 »	*Apple iPhone and iPod Touch*	305	9
MeeGo/Harmattan »	*Nokia N9 and N950*	271	14
BlackBerry OS 7 »	*BlackBerry Bold 9900 and others*	266	3
Android 4.0 »	*Samsung Galaxy Nexus*	256	3
Bada 2.0 »	*Samsung Wave and others*	251	9
Nokia Belle FP 1 »	*Nokia 603, 700 and 701*	212	9
webOS 2.2 »	*Palm Pre 2 and HP Pre 3*	201	5
Android 2.3 »	*Google Nexus S and others*	182	1
Windows Phone 7.5 »	*Samsung Omnia W, LG E906 and others*	141	5

development or beta

		Score	Bonus
Tizen 1 »		387	15
BlackBerry 10 »		361	10
Chrome Beta »	*All Android 4 devices*	343	10
Firefox Mobile 12 »	*Multiple platforms*	318	9
Windows Phone 8 »		298	6
Nokia Belle FP 2 »		242	9

older

		Score	Bonus
Firefox Mobile 9 »	*Multiple platforms*	313	9
Opera Mobile 11.50 »	*Multiple platforms*	286	9
Firefox Mobile 6	*Multiple platforms*	285	9
BlackBerry OS 6 »	*BlackBerry Torch and others*	258	3
Opera Mobile 11.00 »	*Multiple platforms*	249	8
iOS 4.2 & 4.3 »	*Apple iPhone and iPod Touch*	215	7
iOS 4.0 & 4.1 »	*Apple iPhone and iPod Touch*	182	6
Android 2.2 »	*Motorola Atrix 4G, Samsung Galaxy Pro and others*	182	0
Maemo »	*Nokia N900*	181	1
webOS 2.1 »	*Palm Pre Plus and Pre 2*	160	5
Symbian Belle »	*Nokia 603, 700 and 701*	157	7
Android 2.1 »	*Google Nexus One, Motorola Droid X, HTC*	151	0

TABLE G.2: *MOBILE BROWSER SUPPORT (01.04.2012).*

Table G.2 shows that the new version of Opera Mobile gained first place over Firefox, just over a month later (see Table G.1). Table G.2 also shows that there are many competitors just waiting to take first place. Especially interesting the new browser Tizen, still on version 1 in development/beta, already had 33 points more than the top browser Opera 12.00.

Appendix G.2 ***Tablet browsers***

current

RIM Tablet OS 2	BlackBerry PlayBook	354	9
Firefox Mobile 10	*Multiple platforms*	315	9
iOS 5	Apple iPad	305	9
Opera Mobile 11.50	*Multiple platforms*	286	9
Android 4.0	Motorola Xoom, Samsung Galaxy Tab and others	256	10
Android 3.2	Motorola Xoom, Samsung Galaxy Tab and others	222	3
webOS 3	HP TouchPad	203	6

development or beta

Chrome Beta	*All Android 4 devices*	343	10

older

Firefox Mobile 9	*Multiple platforms*	313	9
RIM Tablet OS 1	BlackBerry PlayBook	273	9
Opera Mobile 11.10	*Multiple platforms*	270	9
Android 3.1	Motorola Xoom, Samsung Galaxy Tab and others	222	8
Android 3.0	Motorola Xoom	222	8
iOS 4.2 & 4.3	Apple iPad	215	9
iOS 3.2	Apple iPad	182	8

TABLE G.3: TABLET BROWSER SUPPORT (24.02.2012). COLUMN NR.3 IS SCORE AND COLUMN NR.4 IS BONUS.

The overview of available browser for tablets in Table G.3 shows that BlackBerry had a big lead on Firefox and Chrome (still in beta). In addition, the score for Firefox and iOS browsers are identical to the ones on the mobile devices, indicating that the same browser versions are available for both device types.

current

		Score	Bonus
Opera Mobile 12.00 »	*Multiple platforms*	354	11
RIM Tablet OS 2.0 »	*BlackBerry PlayBook*	354	9
Firefox Mobile 10 »	*Multiple platforms*	315	9
iOS 5.0 & 5.1 »	*Apple iPad*	305	9
Android 4.0 »	*Asus Transformer Prime and others*	256	3
webOS 3.0 »	*HP TouchPad*	203	6
Silk 1.0 »	*Amazon Kindle Fire*	167	1

development or beta

		Score	Bonus
BlackBerry 10 »	*BlackBerry PlayBook*	361	10
Chrome Beta »	*All Android 4 devices*	343	10
Firefox Mobile 12 »	*Multiple platforms*	318	9

older

		Score	Bonus
Firefox Mobile 9 »	*Multiple platforms*	313	9
Opera Mobile 11.50 »	*Multiple platforms*	286	9
RIM Tablet OS 1.0 »	*BlackBerry PlayBook*	273	8
Opera Mobile 11.10 »	*Multiple platforms*	270	8
Android 3.2 »	*Motorola Xoom, Samsung Galaxy Tab and others*	222	3
iOS 4.2 & 4.3 »	*Apple iPad*	215	7
iOS 4.0 & 4.1 »	*Apple iPad*	182	6
iOS 3.2	*Apple iPad*	139	7

TABLE G.4: *TABLET BROWSER SUPPORT (01.04.2012).*

Table G.4 shows that the leading browser for tablets was Opera 12 and that there are other browsers brands in development that have even greater support for HTML5.

Support for HTML5 was low on some platforms in 2012 (see Table G.2). Opera who was the most used mobile browser up until January 2012 (see Figure E.4) also had the best support for HTML5. Opera broadened their support with version 12 of Opera Mobile. However, in 2012 this browser was only available on Android, Symbian/S60, Windows Mobile (10), Maemo (labs), MeeGo (labs) (Opera 2012).

Appendix H Survey respondents device information

Survey respondents were asked to type in the name of their mobile device. A selection of these phone types are presented in, together with a selection of data on each model from GSM Arena[62]. Duplicates have been removed as well as names that only contained the phone brand.

Brand	Name	Released	Operating system	Screen size	Sensors
Samsung	Galaxy S2 (GSM Arena 2015k)	2011, Q3	Android OS, v2.3.4 (Gingerbread), v4.0.4 (Ice Cream Sandwich), upgradable to v4.1.2 (Jelly Bean)	4.3 inches (~63.5% screen-to-body ratio) 480 x 800 pixels (~217 ppi pixel density)	Accelerometer, gyro, proximity, compass, A-GPS
Samsung	Galaxy S3 (GSM Arena 2015l)	2012, May	Android OS, v4.0.4 (Ice Cream Sandwich), 4.3 (Jelly Bean)	4.8 inches (~65.9% screen-to-body ratio) 720 x 1280 pixels (~306 ppi pixel density)	Accelerometer, gyro, proximity, compass, barometer, A-GPS, GLONASS
HTC	Desire HD (GSM Arena 2015d)	2010, September	Android OS, v2.2 (Froyo), v2.3 (Gingerbread), not upgradable to v4.0 (Ice Cream Sandwich)	4.3 inches (~62.9% screen-to-body ratio) 480 x 800 pixels (~217 ppi pixel density)	Accelerometer, proximity, compass, A-GPS
Apple	iPhone 4 (GSM Arena 2015a)	2010, June	iOS 4, upgradable to iOS 7.1.1	3.5 inches (~54.0% screen-to-body ratio) 640 x 960 pixels (~330 ppi pixel density)	Accelerometer, gyro, proximity, compass, A-GPS
Apple	iPhone 4s (GSM Arena 2015b)	2011, October	iOS 5, upgradable to iOS 9.1	3.5 inches (~54.0% screen-to-body ratio) 640 x 960 pixels (~330 ppi pixel density)	Accelerometer, gyro, proximity, compass, A-GPS, GLONASS
Apple	iPhone 5 (GSM Arena 2015c)	2012, September	iOS 6, upgradable to iOS 9.1	4.0 inches (~60.8% screen-to-body ratio) 640 x 1136 pixels (~326 ppi pixel density)	Accelerometer, gyro, proximity, compass, A-GPS, GLONASS
Nokia	Lumia (GSM Arena 2015i)	2013, February	Microsoft Windows Phone 8, upgradeable to v8.1	4.3 inches (~61.0% screen-to-body ratio) 480 x 800 pixels (~217 ppi pixel density)	Accelerometer, proximity, compass, A-GPS, GLONASS
Nokia	3720 (GSM Arena 2015g)	2009, July	N/A	2.2 inches (~27.7% screen-to-body ratio) 240 x 320 pixels (~182 ppi pixel density)	None
Nokia	N8 (GSM Arena 2015j)	2010, April	Symbian^3 OS, upgradable to Nokia Belle Refresh	3.5 inches (~50.3% screen-to-body ratio) 360 x 640 pixels (~210 ppi pixel density)	Accelerometer, proximity, compass, A-GPS; Ovi Maps 3.0
Nokia	C5 (GSM Arena 2015h)	2010, March	Symbian OS v9.3, Series 60 rel. 3.2	2.2 inches (~29.1% screen-to-body ratio) 240 x 320 pixels (~182 ppi pixel density)	A-GPS; Ovi Maps 3.0
HTC	One S (GSM Arena 2015f)	2012, February	Android OS, v4.0 (Ice Cream Sandwich), upgradable to v4.1.1 (Jelly Bean)	4.3 inches (~59.9% screen-to-body ratio) 540 x 960 pixels (~256 ppi pixel density)	Accelerometer, gyro, proximity, compass, A-GPS

[62] GSM Arena (refer to: http://www.gsmarena.com/faq.php3)

HTC	Legend (GSM Arena 2015e)	2010, February	Android OS, v2.1 (Eclair)	3.2 inches (~48.4% screen-to-body ratio) 320 x 480 pixels (~180 ppi pixel density)	Accelerometer, proximity, compass, A-GPS

TABLE H.1: MOBILE DEVICES USED BY SURVEY RESPONDENTS AND THE DEVICE PROPERTIES

Appendix I Survey questionnaire

The questions in the ELA customer survey are listed below and is also available on this URL: https://docs.google.com/forms/d/1QokzvxjRdx15LIxnKdmPQm3btT_WUyqlUuqGf_dtnaY/viewform

Mobilitet: ELA spørreundersøkelse

Denne spørreundersøkelsen er laget av Daniel Huus som en del av hans masteroppgave ved Høgskolen i Molde. Oppgaven er laget i samarbeid med ADCom Data Molde. Spørreundersøkelsens formål er å gi et bedre oversikt over hvilke krav som stilles til en mobil applikasjon samt hva dere som kunder ønsker av en applikasjon.

* Required

Hvilket firma/e-verk jobber du for? *

Navn på firma

Er du mann eller kvinne? *

- Mann
- Kvinne

Hvor gammel er du? *

Hvilken rolle har du i ditt firma? *

- Ledelse
- Økonomi
- Ingeniør
- Montør
- Other:

Hvor mange ansatte er dere i din bedrift? *

- Under 10
- 10-15
- 16-20
- 21-25
- 26-30
- Over 30

Har du jobbtelefon? *

- ○ Ja
- ○ Nei
- ○ Både jobbtelefon og privat telefon

Hvilken mobiloperatør har du? *

Dersom du har flere operatører (både privat og på jobb) så kan du krysse av for flere

- ☐ Telenor
- ☐ Netcom
- ☐ Talkmore
- ☐ Tele2
- ☐ Chess
- ☐ One Call
- ☐ Ventelo
- ☐ Other: ____________

Hvilket operativsystem/platform har du på din mobiltelefon idag? *

- ○ Android
- ○ iOS (iPhone)
- ○ Symbian
- ○ webOS
- ○ BlackBerry
- ○ Windows mobile (versjon 6 eller eldre)
- ○ Windows mobile (versjon 7)
- ○ Vet ikke
- ○ Other: ____________

Hvilken produsent har laget din mobiltelefon? *

- ○ Nokia
- ○ Samsung
- ○ HTC
- ○ Apple (iPhone)
- ○ Sony Ericsson
- ○ LG
- ○ Phillips
- ○ Siemens
- ○ Motorola
- ○ BlackBerry
- ○ Other: ____________

Hva heter telefonen din?
(du trenger ikke svare på dette spørsmålet)

Hvor lenge har du hatt mobiltelefonen din? *

- 0-6 mnd.
- 6-12 mnd.
- 1-1,5 år
- 1,5-2 år
- 2-2,5 år
- over 2,5 år
- Other:

Har du eller din bedrift tenkt å kjøpe ny(e) telefon(er) i nær framtid? *

- Innen 6 mnd.
- Innen 9 mnd.
- Innen 1 år.
- Over 1 år.
- Vet ikke
- Nei
- Other:

Hvilke funksjoner har telefonen din? *

- GPS
- Kamera
- Touch-skjerm
- Fullt tastatur
- T9 (talltastatur)
- WiFi
- 3G/4G
- BlueTooth
- Other:

Hvilke tilkoplinger bruker du ofte på din telefon? *

- 2G
- 3G
- 4G
- WiFi/Trådløst nett (lokalt, for eks. på jobb eller hjemme)
- BlueTooth
- Other:

Hva bruker du mobiltelefonen din til? *

- [ ] Ringe
- [ ] SMS
- [ ] MMS
- [ ] Internett surfing
- [ ] Musikk
- [ ] Video/Film
- [ ] Spill
- [ ] Sosiale medier
- [ ] E-post
- [ ] Kalender
- [ ] Timeføring
- [ ] Banktjenester
- [ ] Other: ________

Når du er på jobb, hvor ofte er du uten mobildekning? *

I en skala fra 1 til 10, hvor ofte er du uten dekning når du er på jobb?

	1	2	3	4	5	6	7	8	9	10	
Aldri	○	○	○	○	○	○	○	○	○	○	Ofte

Når du ikke er på jobb, hvor ofte er du uten mobildekning? *

I en skala fra 1 til 10, hvor ofte er du uten dekning når du ikke er på jobb?

	1	2	3	4	5	6	7	8	9	10	
Aldri	○	○	○	○	○	○	○	○	○	○	Ofte

Bruker du noen sosiale medier?
(ikke nødvendigvis via mobiltelefonen)

- ☐ Facebook
- ☐ Twitter
- ☐ LinkedIn
- ☐ Google+
- ☐ Flickr
- ☐ Nei
- ☐ Other: ________

Har du noen gang lastet ned nye program/apps fra AppStore, Android Market eller lignende? *

- ☐ Ja, Android Market
- ☐ Ja, AppStore
- ☐ Nei
- ☐ Other: ________

Ønsker du å bruke telefonen mer aktivt i jobbsammenheng? *

- ○ Ja
- ○ Nei
- ○ Kanskje
- ○ Other: ________

Bruker dere noen av disse verktøyene aktivt (mobilt) i din bedrift idag? *

- ☐ Tablet (for eks. iPad)
- ☐ Mobiltelefon
- ☐ Bærbar PC
- ☐ Ingen av delene

Hvis du kunne velge, hvilket verktøy ville du foretrukket å bruke mobilt i jobbsammenheng? *

- Tablet (for eks. iPad)
- Bærbar PC
- Mobiltelefon
- Other: ________

Hvilke funksjoner ville du personlig verdsatt mest i en mobil applikasjon? *
(på tablet, mobil eller bærbar pc) 1 = lavt, 5 = høyt

	1	2	3	4	5
Timeregistrering	○	○	○	○	○
Materiellføring	○	○	○	○	○
Arbeidsordre	○	○	○	○	○
Risikovurdering	○	○	○	○	○
Kart	○	○	○	○	○

Hvilke funksjoner tror du ditt firma/e-verk ville verdsatt mest i en mobil applikasjon? *
(på tablet, mobil eller bærbar pc) 1 = lavt, 5 = høyt

	1	2	3	4	5
Timeregistrering	○	○	○	○	○
Materiellføring	○	○	○	○	○
Arbeidsordre	○	○	○	○	○
Risikovurdering	○	○	○	○	○
Kart	○	○	○	○	○

Er det ønskelig at en mobil applikasjon bruker geografisk plassering for å finne relevant informasjon? *

- Ja
- Kanskje
- Nei
- Other: ________

Ranger følgende funksjoner i en mobil applikasjon: *
(på tablet, mobil eller bærbar pc) 1 = lavt, 5 = høyt

	1	2	3	4	5
Sikkerhet	○	○	○	○	○
Reponsivt grensesnitt (lite forsinkelser i for eks. menyvalg)	○	○	○	○	○
Brukervennlig	○	○	○	○	○
Tilgjengelighet (online)	○	○	○	○	○
Mulighet for å jobbe offline	○	○	○	○	○
Bruk av geografisk posisjonering/GPS (kart/veibeskrivelse)	○	○	○	○	○
Bildefunksjon (ta bilder med kameraet på tablet/mobil)	○	○	○	○	○

Er din bedrift interessert i å anskaffe mobiltelefoner eller tablets (iPad) som støtter en mobil applikasjon? *

- ○ Ja
- ○ Nei
- ○ Kanskje
- ○ Other: ____________

Hvor mye tror du din bedrift er interessert i å betale for en slik applikasjon? *

- Under 10 000,-
- Under 20 000,-
- Under 30 000,-
- Under 40 000,-
- Under 50 000,-
- Over 50 000,-
- Other:

Dersom ADCom Data utvikler en slik mobil applikasjon, vil du og din bedrift være interessert i å anskaffe den? *

- Ja
- Nei
- Kanskje
- Other:

Kommentar

Dersom du har noe du vil tilføye denne spørreundersøkelsen

Appendix J Interview with the ELA Mobile developer

This appendix contain the questions and answers from an interview conducted with Pål Gammelsæter, the developer of the front-end of ELA Mobile.

How long have you been a developer?

I have been working as a developer for four years, but I have been programming since I was nine years old when I first programmed on my Commodore 64.

What education do you have?

I have a Bachelor in Informatics from the University in Bergen. We mainly used Java, but we looked at many different programming languages to learn about the different programming paradigms.

What programming languages do you know best and how long have you been using them?

I know PHP the best. I have been using PHP since about 2001, and I started with HTML in 1995.

What frameworks have you been working with before?

I have used the Yii framework for about 1 ½ years. I got a tip about it in my last job and started using it while working for Adcom Molde when it was necessary. I have also been using some components from the Zend framework, but it is hard to learn. It is a complex and large framework. I used Concrete5 quite a lot in my last job, but that is mainly a CMS (content management system). I have also used Elgg; it is an open-source framework for creating social websites.

What mobile framework was chosen for the mobile application and why? What other candidates were reviewed?

We chose JQuery Mobile because it is the largest available framework to this date. I looked at others as well, but I felt it was safest to go with the largest framework for technological stability and predictable future.

Was the framework hard to use/learn compared to others you have used?

No, it was a medium challenge. It is a bit hard to compare them to other frameworks since it is a framework for mobile applications. It builds on HTML and uses a JavaScript engine that changes the DOM model. It was difficult to understand the jQuery structure and how the components are connected. The framework has defined its own attributes that are not a part of the HTML standard so I have to use the framework documentation every time I have to change an element.

What HTML5 features does the application use?

Currently the application uses the camera API to capture photos and upload them to the server.

When testing the framework on different devices, where there any variations between devices and platforms?

There were surprisingly little variation between devices. It was very compatible with the different platforms. The only browser we have had some trouble with was Internet Explorer in Windows, but to be fair, this application is not meant to be used with desktop browsers. Our target group is the mobile devices with touch functionality.

Did you have to do any changes to the framework to make it work as intended?

No, I did not have to do any changes. However, the different devices/browsers show different types of keyboard layouts for some types of input fields. When registering hours the users often need to type in decimal numbers, but some keyboard layouts does not have comma/punctuation, only numbers. To fix this I have to manipulate the attributes on the input field so that the browser shows a different keyboard layout. Each user can set up this setting for their device.

How long time did you use to create the mobile application?

We have used about 6 months to develop the application. However, this is not full time. There have been other ongoing projects at the same time. If I had only worked with this project and Daniel (the author) had worked full time, I estimate that we would have used around two months.

What plans does Adcom have for the next versions for the application?

I suppose we will develop an interface for the risk assessment documents. Other than that, I do not know what other plans we have yet.

In retrospect, was developing a mobile web application the right decision for Adcom Molde, or do you think they should have chosen a different approach for the mobile application?

No, I think it was the right decision. I am not a big fan of this kind of applications, but for this purpose, it was the right decision. It works on all mobile devices. We have not had any complaints yet and people are impressed with the application. It is not an application that needs a fast interface and all you need is a shortcut on your home screen to the application.

Do you think you would have had difficulties with developing a web application if you were not already a web developer?

I think you have to be a web developer, to be able to start quickly with JQuery Mobile. You have to know HTML/CSS to be able to use it. It is also an advantage to know some about server configurations. While developing a web application you might find out that, you should have developed a native application instead since you are dependent on an internet connection and the response time in the application is not very good. You need to know what limits web applications have compared to native applications. The user experience will not be the best, but it covers a large need for our customers and supports a large amount of devices. Another advantage is that updating the application is fast. As soon as you have updated the application, the update is available for all users.

Appendix K Google Analytics dataset

The Google Analytics dataset contains a summary of usage data from the ELA Mobile application between 08.10.2015 and 03.12.2015.

Appendix K.1 *Platforms*

Table K.1 shows a list of the various operating systems (both mobile and PC) that were used to access ELA Mobile.

Operating System	Sessions
	5,400 % of Total: 89.06% (6,063)
1. Windows	2,133 (39.50%)
2. Android	1,938 (35.89%)
3. iOS	1,318 (24.41%)
4. Linux	5 (0.09%)
5. Macintosh	4 (0.07%)
6. Windows Phone	2 (0.04%)

TABLE K.1: OPERATING SYSTEM USED WITH ELA MOBILE.

Table K.2 shows a summary of each PC operating system version that was used to access ELA Mobile.

Windows	Sessions	Macintosh	Sessions	Linux	Sessions
10	82	**Intel 10.10**	2	**x86_64**	5
7	1788	**Intel 10.11**	1		
8.1	137	**Intel 10.9**	1		
Vista	34				
XP	92				
Total sessions:	**2133**		**4**		**5**

TABLE K.2: DESKTOP OPERATION SYSTEM VERSION DISTRIBUTION.

Table K.3 show a summary of each mobile operating system version that were used to access ELA Mobile.

Android versions	Sessions	iOS versions	Sessions	Windows Phone versions	Sessions
2.3.5	9	7.1	30	8.0	2
4.1.2	56	7.1.1	47		
4.2.1	144	7.1.2	69		
4.3	67	8.1.2	14		
4.4.2	124	8.1.3	9		
4.4.4	222	8.3	7		
5.0	425	8.4	24		
5.0.1	370	8.4.1	54		
5.0.2	80	9.0	21		
5.1.1	441	9.0.1	40		
		9.0.2	245		
		9.1	758		
Total sessions:	1938		1318		2

TABLE K.3: MOBILE PLATFORM VERSION DISTRIBUTION.

Appendix K.2 *Devices*

Table K.4 shows a summary of the number of sessions registered for each unique mobile device model that were used to access ELA Mobile.

Android	Sessions
(not set)	175
Asus K010	9
CAT S50	27
Feiteng GT-i9300	2
Mozilla Firefox for Android	144
Mozilla Firefox for Android Tablet	2
Samsung GT-I9100 Galaxy S II	59
Samsung GT-I9295 Galaxy S4 Active	179
Samsung GT-I9300 Galaxy S III	23
Samsung GT-I9500 Galaxy S IV	45
Samsung GT-I9505 Galaxy S IV	35
Samsung GT-I9506 Galaxy S4	33
Samsung GT-N5110 Galaxy Note 8.0	5
Samsung GT-P3100 Galaxy Tab 2 7.0	1
Samsung GT-S7710 Galaxy Xcover 2	5
Samsung I9506 Galaxy S4	33
Samsung SM-G870A Galaxy S5 Active	163
Samsung SM-G900F Galaxy S5	128
Samsung SM-G920F Galaxy S6	18
Samsung SM-G925F Galaxy S6 Edge	41

Samsung SM-N9005 Galaxy Note 3	38
Samsung SM-N910F Samsung Galaxy Note 4	87
Samsung SM-N915FY Galaxy Note Edge	15
Samsung SM-P600 Galaxy Note 10.1 2014	24
Samsung SM-T365 Galaxy Tab Active LTE-A	80
Samsung SM-T535 Galaxy Tab 4 10.1	2
Samsung SM-T800 Galaxy Tab S 10.5	1
Samsung SM-T817 Galaxy Tab S2 9.7	22
Sony C6903 Xperia Z1	5
Sony D5503 Xperia Z1 Compact	63
Sony D5803 Xperia Z3 Compact	84
Sony D6503 Xperia Z2	29
Sony D6603 Xperia Z3	170
Sony E2303 Xperia M4 Aqua	96
Sony E6553 Xperia Z3+	10
Sony E6653 Xperia Z5	43
Sony LT25i Xperia V	42
iOS	Sessions
Apple iPad	92
Apple iPhone	1226
Windows Phone	Sessions
Nokia Lumia 925	2

TABLE K.4: LIST OF DEVICE ID'S AND NUMBER OF SESSIONS.

Appendix K.3 ***Browsers***

Table K.5 lists every unique browser type in the dataset (marked with blue) together with the total number of sessions. Below each browser type the unique versions for each browser are shown together with the total number of sessions for each version.

Browser type/version	Sessions
Android Browser	**88**
4.0	88
Chrome	**2439**
12.0.742.112	2
28.0.1500.94	4
30.0.0.0	53
33.0.0.0	5
34.0.1847.76	467
35.0.1916.141	4
36.0.1985.534	11
38.0.2125.102	40
42.0.2311.111	7

43.0.2357.93	20
44.0.2403.133	30
45.0.2454.101	89
45.0.2454.89	1
45.0.2454.94	244
46.0.2490.71	145
46.0.2490.76	800
46.0.2490.80	255
46.0.2490.86	253
47.0.2526.73	9
Edge	**5**
12.10240	5
Firefox	**316**
40.0	4
41.0	168
42.0	144
Internet Explorer	**1206**
10.0	278
11.0	895
9.0	33
Opera	**25**
32.0.1953.96473	12
33.0.1990.43	1
33.0.2002.97426	7
33.0.2002.97617	2
33.0.2002.98088	3
Safari	1321
7.0	146
7.1	1
7534.48.3	3
8.0	108
9.0	1062
9.0.1	1

TABLE K.5: BROWSER TYPE AND VERSION DISTRIBUTION.

Appendix L Project cost calculations

Appendix L contains cost and years' work calculations for the ELA Mobile project.

Appendix L.1 *Project hours*

Employee name	2012	2013	2014	2015	Total
Daniel Huus	32	94	42	91	**259**
Pål Gammelsæter	0	373	107	314	**794**
Arild Kjølseth	0	86	46	111	**243**
John Erik Johnsen	0	51	73	12	**136**
May Britt Solheim	0	60	144	29	**233**
Total:	**32**	**664**	**412**	**557**	**1665**

TABLE L.1: SUMMARY OF HOURS USED PR. YEAR AND PR. EMPLOYEE ON THE ELA MOBILE PROJECT (FROM 01.01.2012 TO 28.11.2015)

Table L.1 note: The hours only include work hours from the hour lists that resulted in a cost for Adcom Molde. Hours that have been used to write the thesis and conduct the research are not included. The hours have been rounded up to the nearest integer.

Table L.1 note: The hours from May Britt Solheim includes meeting and sales activities for the ELA application as well. Thus, some of these hours could have been excluded. However, since the ELA and ELA Mobile sales activities are closely linked they have been included in the calculation.

According to Statistics Norway[63] (SSB), one year's work for one fulltime employee is 1750 hours, excluding holidays (Statistics Norway 2015).

Appendix L.2 *Calculation values*

- One year's work according to SSB = 1750 hours (Hyear)
- Work hours pr. month (Hday x Days x Weeks) = 162.375 ≈ 162 (Hmonth)
  Note: an average since and each month is individual with regard to number of days and holidays.
- A average number of weeks pr. month = 4.33 weeks (Weeks)
- Work days pr. week = 5 days (Days)
- Hours in a normal work day = 7.5 hours (Hday)

[63] Statistics Norway (refer to: http://www.ssb.no/omssb/om-oss)

- Internal cost for one work hour in Adcom Molde = 500 NOK (Ci)
- Front-end development hours for version one = 307 hours (H1)
- Total hours front-end development = 794 hours (H2)
- Total project hours = 1665 hours (H3)
- Native development time estimate (section 7.4) = 20 months (N)

Note: Internal cost for one work hour, covers daily expenses that includes salary, payroll taxes, equipment costs and common costs. They do not include expenses such as mobile devices used for testing, travel expenses related to customer meetings etc.

Appendix L.3 *ELA Mobile costs*

The following calculations represent an estimate of the cost for the ELA Mobile project based on the hour list in Table L.1**.**

Appendix L.3.1 Total project years' work

Hyear ÷ H3 = 0.951 year's work

Appendix L.3.2 Total project cost

H3 x Ci = 832,500 NOK

Appendix L.3.3 Total project sales earnings

Project sales:

The total income from actual sales of ELA Mobile license is 330,700 NOK.

Invoiced hours:

In addition to the sales figures there are also a few hours that have been invoiced to customers related to support, training and version upgrades. It is estimated that the total income for these hours is 120,000 NOK.

Yearly license fee:

In addition to the sales figures there is also a yearly fee for ELA Mobile that will cover maintenance and server costs etc. This fee is set to 8,112 NOK pr. Customer pr. Year.

2014: 7 customers for the whole year ca. 56,785 NOK

2015: 11 customers for the whole year ca. 89,232 NOK

Project income:

330,700 + 120,000 + 56,785 + 89,232 = 596,717 NOK ≈ 597,000 NOK (income)

Project balance:

In addition to the cost of hours there are some costs related to server maintenance, however the figures are not exact and are not included in the calculation.

597,000 NOK (income) - 832,500 NOK (internal cost) = -235,500 NOK (deficit)

Appendix L.3.4 Front-end development for version one

Number of work months (based on used hours) for version one:

H1 ÷ Hmonth = 1,895 months

Cost of front-end development for version one:

H1 x Ci = 153,500 NOK

Appendix L.3.5 Front-end development for the whole project

Years' work:

H2 ÷ Hmonth = 0,454 years' work

Cost of front-end development:

H2 x Ci = 397,000 NOK

Appendix L.4 *Alternative estimates*

These calculations are estimates for alternative development methods.

Appendix L.4.1 Native development method hours (full time, first version)

Equivalent number of hour's pr. month for full time native development:

N x Hmonth = 3,240 hours

Estimated project cost for full time native development:

3,240 hours x Ci = 1,620,000 NOK

Appendix L.4.2 Native development method (part time, first version)

Average number of hour's pr. month for web development:

H1 ÷ 6 months = 51.16 ≈ 51 hours pr. Month

Equivalent number of hour's for native development:

N x 51 hours pr. Month = 1020 hours

Estimated project cost for native development:

1020 hours x Ci = 510,000 NOK

Additional cost of native development compared to web development:

(H1 x Ci) - (1020 hours x Ci) = 356,500 NOK

Appendix L.4.3 **Native development method (part time, whole project)**

Estimated hour development ratio between native and web development:

1020 hours (from Appendix L.4.2) ÷ H1 = 3.32

Estimated hours used on native front-end development for the whole project:

H2 x 3.32 ratio = 2636 hours

Estimated costs for native front-end development for the whole project:

2636 hours x Ci = 1,318,000 NOK

Additional cost of native development compared to web development for the whole project:

(2636 hours x Ci) - (H2 x Ci) = 921,000 NOK

Years' work for the estimated hours used on native front-end development for the whole project:

2636 hours ÷ Hyear = 1.506 years' work

Additional years' work for native development compared to web development:

1.506 - (H2 ÷ Hyear) = 1.052 years' work

Additional years' work for native development compared to entire project:

1.506 – (H3 ÷ Hyear) = 0.555 years' work

Appendix M ELA Mobile back-end configuration

The CPN model consists of places and transitions with arcs between them creating a graph. A place is a circle/eclipse while a transition is a square/rectangle.

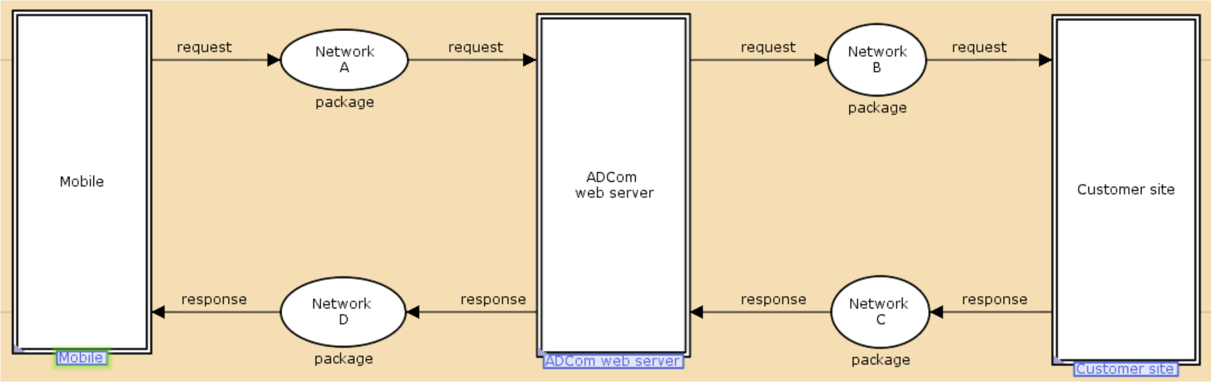


FIGURE M.1: TOP-LEVEL VIEW OF THE CPN MODEL.

Figure M.1 shows that the model consists of four major parts:

1. The mobile device (Figure M.2).
2. Adcom web server (Figure M.4).
3. Customer site (web service and database, Figure M.5).
4. The networks connecting the servers and mobile device (shown in Figure M.1).

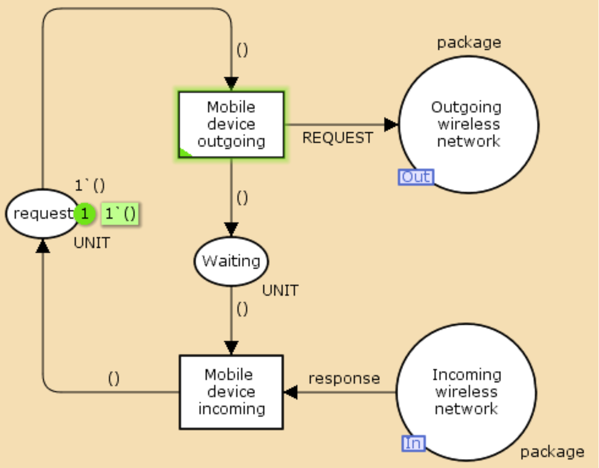


FIGURE M.2: DETAILED VIEW OF THE MOBILE DEVICE.

The first part of the model is a mobile device (Figure M.2) connected to network "A" (like Wi-Fi or 3G). As seen in Figure 8.1, the mobile device access the Adcom web server (hereinafter WS1) located in the demilitarized zone[64] (DMZ) of the Adcom Molde network

[64] Demilitarized zone (refer to: https://en.wikipedia.org/wiki/DMZ_(computing))

through HTTPS[65]. WS1 hosts the ELA Mobile web application and is protected by a firewall. The first request downloads the web application to the mobile device. All traffic between the mobile device and WS1 is encrypted with a TLS[66] certificate from an approved certificate authority[67] (CA) as seen in Figure M.3. The certificate is a paid service to ensure that a connection is automatically trusted between the mobile device and the server. If the certificate would not be signed by a CA, then users would be prompted by a security warnings saying the site cannot be trusted. In addition to being secure it is also more user-friendly for ELA Mobile users. There are no connections to the inner network of Adcom Molde.

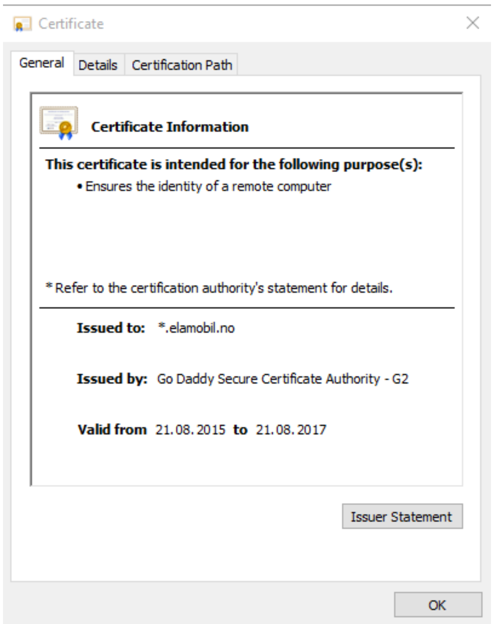


FIGURE M.3: TLS CERTIFICATE OF ELA MOBILE.

When the mobile device makes a subsequent request to WS1 it goes in to a waiting state while it is waiting for a response from the web server.

[65] HTTPS (refer to: https://en.wikipedia.org/wiki/HTTPS)

[66] Transport layer security (refer to: https://en.wikipedia.org/wiki/Transport_Layer_Security)

[67] Certificate authority (refer to: https://en.wikipedia.org/wiki/Certificate_authority)

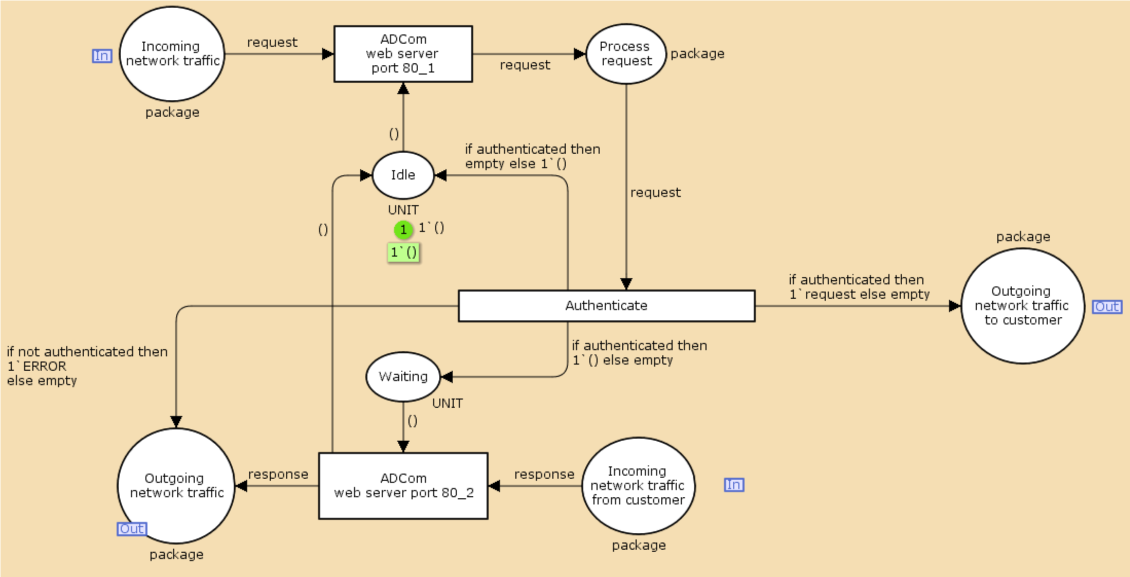


FIGURE M.4: DETAILED VIEW OF THE ADCOM WEB SERVER.

When WS1 receives the request, it checks the authentication of the user. If the user is not authenticated the server sends back an error message (ask the user to log in). If the user is authenticated a request is sent to the customer's web server (hereinafter WS2) seen in Figure M.5, that is running a web service application, the back-end of ELA Mobile. The traffic between WS1 and WS2 is also encrypted, however these certificates are not signed by a CA since it would require a new certificate for each customer and would result in higher costs as well as more maintenance. The certificates are automatically approved by the servers and is only used for encryption purposes.

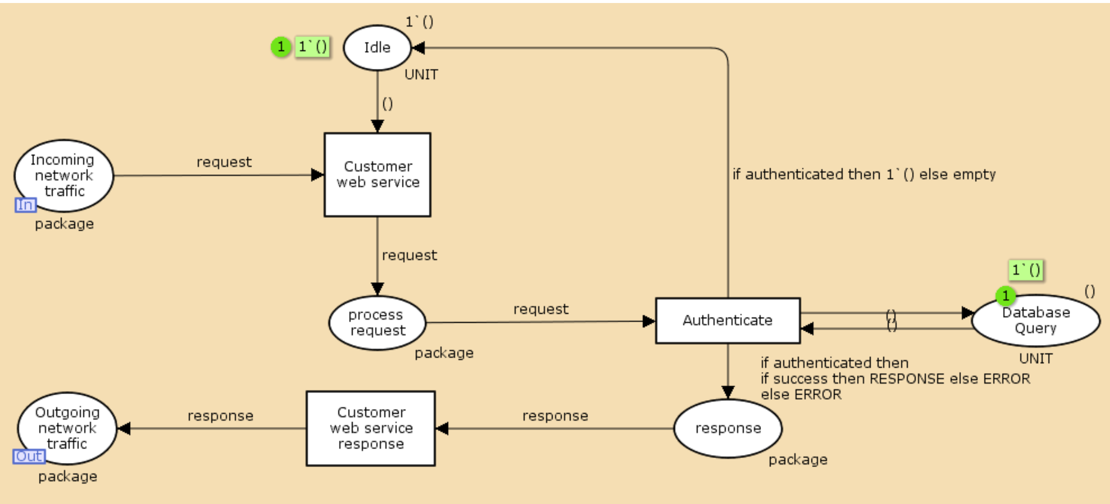


FIGURE M.5: DETAILED VIEW OF THE CUSTOMER WEB SERVICE.

While WS1 is waiting for a response, it goes in to a waiting state until it receives a response from the WS2. The URL for WS2 is stored in the customer configuration for the current domain (not shown in the model). WS2 also checks that the user credentials are correct and that the user has the correct level of authorization for the received request. The users are identified by a unique token that is only available for a period of time. If the request is authenticated by WS2 executes the requested query the customers ELA database and sends a response with the result of the query. If the user is not authenticated the server sends an error response. WS2 also have a limit to how many tries a request can fail authentication. For the sake of simplicity, it is one time in this model and the idle function implements the mechanism. In a working environment, this number would be higher and for instance include a timeout that reset the number of tries after a given interval. The implemented web service solution does not have this feature yet. More information about the development of the back-end web service can be found in section 8.2.5.

The response from WS2 is received on WS1 and passed on to the mobile device. Then the WS1 goes from the waiting state, back to its initial idle state, and is ready to receive new requests. When receiving the response on the mobile device the device processes the response, goes back to the idle state, and will then be able to send more requests.

When creating CPN models it is also possible to calculate State Space reports. These reports give details about all possible states that the model can have and has information about the possible transitions from the different states in the model. This gives helpful insight in the feasibility of the model. CPN also allows for automatic code generation, but this is not yet available for mobile devices. However, for the scope of this thesis this topics will not be further elaborated.

Appendix N ELA Mobile examples

The following code represent the login page of ELA Mobile.

Appendix N.1 *Login page - code*

```
<?php

    /*
     * Login view
     *
     */

    /* @var $this AppController */

    // Setter variabler om de ikke er satt i argumentlisten
    $variables = array('username','password','errormsg');
    foreach ($variables as $variable) {
        if (!isset($$variable)) $$variable = '';
    }

    $loginUrl = $this->createUrl('app/login');

?>

<div data-role="page" data-theme="b">
    <div data-role="header">
        <h1>Pålogging ELA Mobil</h1>
    </div>
    <div class="ui-content" role="main">

        <?php

            // Hvis det finnes feilmeldinger, så vis feilmeldingsboksen
            if ($errormsg!='') {
                echo '<div class="errormsg">';
                echo $errormsg;
                echo '</div>';
            }

        ?>

        <form method="post" action="<?php echo $loginUrl; ?>">
            <div class="ui-field-contain">
                <label for="username">Brukernavn:</label>
                <input type="text" name="username" value="<?php echo $username; ?>" id="username" placeholder="Brukernavn"/>
            </div>

            <div class="ui-field-contain">
                <label for="password">Passord:</label>
                <input type="password" name="password" id="password" />
            </div>

            <div class="ui-field-contain">
                <label for="login_submit" /> </label>
                <input type="submit" class="" name="login_submit" id="login_submit" value="Logg inn"/>
            </div>
        </form>

        <div class="version">v.<?php echo Yii::app()->params['version']; ?></div>
    </div>
</div>
```

First, PHP and the Yii framework is used to set the variables needed to login, including the web service URL the login page should query for authentication. A new jQuery Mobile page is created by using HTML and the attribute data-role=”page”[68] on a div-tag. The data-theme[69] attribute is also used to tell jQuery that the new page should use CSS theme “b”. Usually jQuery Mobile comes with two themes, a light theme called “a” and a dark theme called “b”. ELA Mobile uses a custom theme that was created with ThemeRoller[70], a theme generator for jQuery Mobile.

After creating the new page, a header is created by using the data-role=”header”. Inside the header the name of the page is displayed. Then the content of the page is created by using the role attribute. First, PHP checks if there are any errors and displays them below the header, for instance after a failed login.

Then a regular HTML form is created for input together with a button to submit the form and the various elements of the form is connecting to the proper jQuery attribute. Finally, the application version is displayed in the bottom right corner using a custom CSS class.

Appendix N.2 *Login page – result*

The result of the code from Appendix N.1 is shown in Figure N.1.

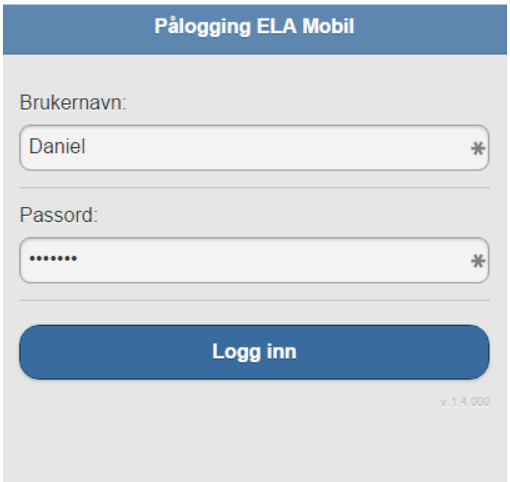


FIGURE N.1: LOGIN SCREEN OF ELA MOBILE.

[68] Data-role (refer to: http://www.w3schools.com/jquerymobile/jquerymobile_pages.asp)

[69] Data-theme (refer to: http://www.w3schools.com/jquerymobile/jquerymobile_themes.asp)

[70] ThemeRoller (refer to: https://themeroller.jquerymobile.com/)

Appendix N.3 ***AJAX transitions***

Code to disable the Ajax transitions in jQuery Mobile:

```
<script type="text/javascript">
                            $(document).on("mobileinit", function() {
                        $.mobile.ajaxEnabled = false;
    })
</script>
```

Appendix N.4 ***Media Capture API example***

The document archive page uses the HTML5 Media capture API. This page is accessed on the work order form and lists all files connected to the work order.

API call:

```
<input type="file" accept="image/*;capture=camera" name="img" id="img" />
```

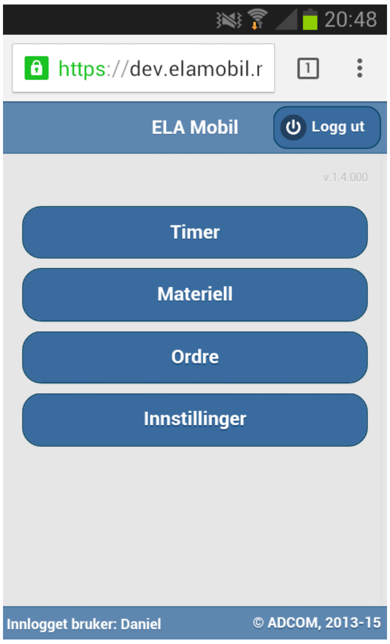
The complete form:

```
<form action="" method="post" enctype="multipart/form-data" id="upload-form">
    <h3>Valgt bilde</h3>
    <input type="file" accept="image/*;capture=camera" name="img" id="img" />
    Endre filnavn: <input type="text" id="filename" />
    <input type="hidden" id="extension" />
    <input type="submit" id="upload" value="Last opp" data-icon="check" />
</form>
```

In addition to the code above, there is also added some JavaScript in ELA Mobile to give the user feedback about events in the capture and upload processes and code that retrieves the list of files that already exist in the document archive. The description below shows the steps users must follow to access and use the Media Capture API in ELA Mobile.

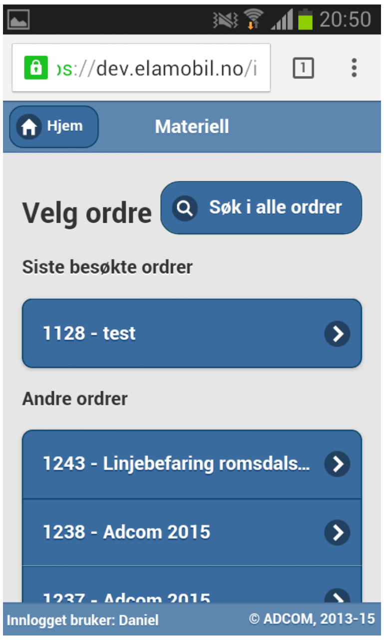
Step 1

The user has logged in and selects "Ordre" from the menu.



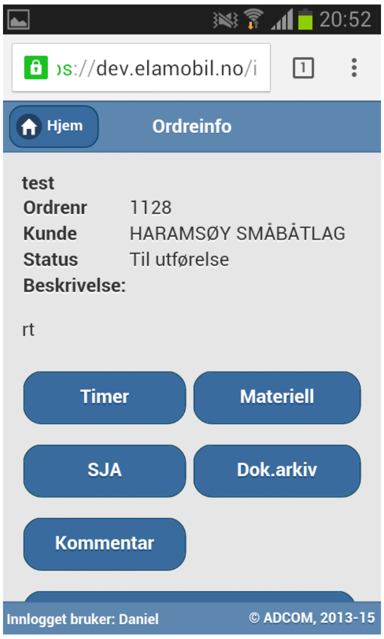
Step 2

A list of work orders will be shown, with the five most resent work orders on the top.



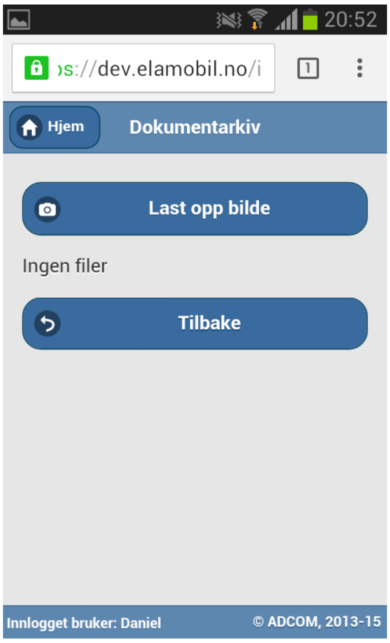
Step 3

The work order details are displayed together with the buttons for each action. The user select the «Dok.arkiv» button.



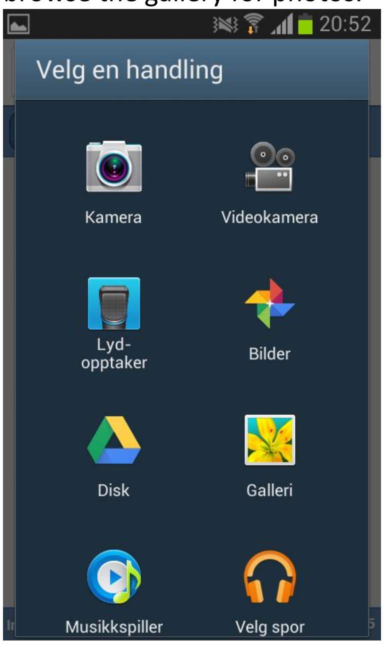
Step 4

The user is presented with a list of all documents available on the work order. No files where available for the work order in this example.



Step 5

The API have been activated. The user is prompted with a pop-up from the Android system to select an action. For instance, the user can take a new photo or browse the gallery for photos.

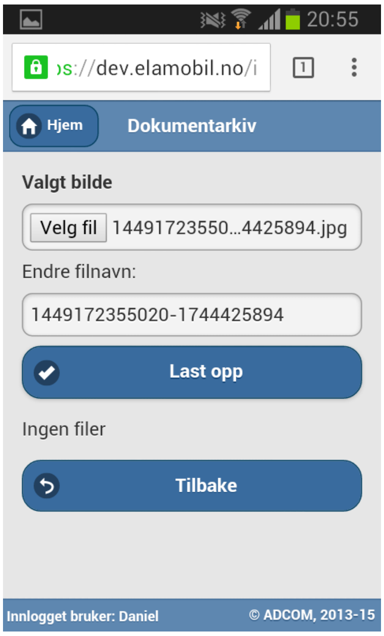


Step 6

The user takes a photo with the camera and press save.

Step 7

After taking the photo and pressing save, ELA mobil will be shown again with the name of the selected image. Here it is also possible to give the file a new name before uploading it to the server.



Step 8

The user press the “Last opp” button to upload the selected photo. A spinning process indicator is shown during the uploading to tell the user that the application is working.

Step 9

When the photo is uploaded, the file archive is refreshed showing the current files. It is now possible to repeat step four to nine again.

Appendix O ELA Mobile back-end examples

Appendix O.1 *Back-end service interface example*

The following example code shows two of the service interfaces that are available in the ELA web service:

```
namespace Webservice
{
    [ServiceContract]
    public interface IELAPPService
    {
        #region STATUS

        [OperationContract]
        [WebInvoke(Method = "GET", UriTemplate = "/servicecheck",
        RequestFormat = WebMessageFormat.Json, ResponseFormat = WebMessageFormat.Json)]
        String serviceCheck();

        #endregion

        #region GET

        [OperationContract]
        [WebInvoke(Method = "GET", UriTemplate = "/get/lonnsart",
        RequestFormat = WebMessageFormat.Json, ResponseFormat = WebMessageFormat.Json)]
        ELALibrary.Domain.ELA.Lonnsart getLonnsart();

        #endregion

    }
}
```

Appendix O.2 *Entity framework class*

In the example below, the Entity framework has generated a class in C# using a T4 template. The class represents the entity class shown in Figure 8.2:

```
//------------------------------------------------------------------------------
-----
// <auto-generated>
//     This code was generated from a template.
//
//     Changes to this file may cause incorrect behavior and will be lost
if
//     the code is regenerated.
// </auto-generated>
//------------------------------------------------------------------------------
-----

using System;
using System.Collections;
using System.Collections.Generic;
using System.Collections.ObjectModel;
using System.Collections.Specialized;
using System.Runtime.Serialization;

namespace ELALibrary.Domain.ELA
{
    [DataContract(IsReference = true)]
    public partial class Lonnsart
    {
        #region Primitive Properties
        [DataMember]
        public virtual int ID
        {
            get;
            set;
        }
        [DataMember]
        public virtual string LonnsArt
        {
            get;
            set;
        }
        [DataMember]
        public virtual string Beskrivelse
        {
            get;
            set;
        }
        [DataMember]
        public virtual Nullable<int> rowStatus
        {
            get;
            set;
        }

        #endregion
    }
}
```

Appendix 0.3 ***JSON examples***

Appendix 0.3.1 **Example 1**

The following JSON is returned by the operation contract “getLonnsart” shown in Appendix O.1. The returned object is generated using data from the database and returned in the structure of the class shown in Appendix O.2. The JSON below is an array with two entries, each with four value pairs.


```
[
 {
  "ID": 1,
  "LonnsArt": "10",
  "Beskrivelse": "TimeIF",
  "rowStatus": null
 },
 {
  "ID": 2,
  "LonnsArt": "50",
  "Beskrivelse": "Overtid 50%",
  "rowStatus": null,
 }
]
```


Appendix O.3.2 **Example 2**

This example from W3Schools (2014) contains an array called “employees”. Inside the array, enclosed with brackets, we find three objects that all have a first name and a last name.


```
{
"employees": [
    { "firstName":"John" , "lastName":"Doe" },
    { "firstName":"Anna" , "lastName":"Smith" },
    { "firstName":"Peter" , "lastName":"Jones" }
]
}
```

References

Access. 2014. "NetFront Browser." http://gl.access-company.com/products/browser/browser/.

Adobe. 2011. "Adobe Announces Agreement to Acquire Nitobi, Creator of PhoneGap." Accessed 04.06.2015. http://www.adobe.com/aboutadobe/pressroom/pressreleases/201110/AdobeAcquiresNitobi.html.

Adobe. 2015a. "About Adobe AIR." Accessed 13.06.2015. http://help.adobe.com/en_US/air/build/WS5b3ccc516d4fbf351e63e3d118666ade46-7fc4.html.

Adobe. 2015b. "Adobe AIR FAQ." Accessed 07.06.2015. http://www.adobe.com/no/products/air/faq.html.

Adobe. 2015c. "Adobe Open Screen Project FAQ." Accessed 13.06.2015. https://www.adobe.com/openscreenproject/faq/.

Anderson, John. 2013. "Appcelerator Titanium: Up and running." In, ed Mike Hendrickson Andy Oram: O'Reilly Media, Inc.

Appcelerator. 2012a. Appcelerator/IDC Q3 2012 Mobile Developer Report.

Appcelerator. 2012b. Native vs. HTML5 Mobile App Development. 11.

Appcelerator. 2015a. "Appcelerator Open Source."

Appcelerator. 2015b. "Mobile App development platform." http://www.appcelerator.com/.

Appcelerator. 2015c. "Mobile Web Platform Overview." http://docs.appcelerator.com/platform/latest/#!/guide/Mobile_Web_Platform_Overview.

Apperian. 2012. Solving Android Multiple Personality Disorder.

Apple. 2012a. "Identifying iPhone models." http://support.apple.com/kb/HT3939.

Apple. 2012b. "iOS 5.1 Software Update." http://support.apple.com/kb/DL1504.

Apple. 2012c. "iPhone comparison." http://www.apple.com/iphone/compare-iphones/.

Apple. 2013. " iOS Human Interface Guidelines." https://developer.apple.com/library/ios/#documentation/UserExperience/Conceptual/MobileHIG/AppDesign/AppDesign.html#//apple_ref/doc/uid/TP40006556-CH19-SW1.

Apple. 2014. "Distribute." Accessed 01.02.2014. https://developer.apple.com/programs/ios/distribute.html.

Apple. 2015. "Start developing iOS Apps Today: Setup." Apple Accessed 08.05.2015. https://developer.apple.com/library/ios/referencelibrary/GettingStarted/RoadMapiOS/index.html.

Basili, Victor R., Gianluigi Caldiera, and H. Dieter Rombach. 1994. "The Goal Question Metric Paradigm." *Encyclopaedia of Software Engineering* 2:528-532.

Baskerville, Richard L, and Trevor Wood-Harper. 1998. "Diversity in information systems action research methods." *European Journal of Information Systems* 7:17.

Bender, Adam. 2013. "BYOD vs CYOD: Bring or choose your own device?". Computerworld, Last Modified 21.03.2013. http://www.computerworld.com.au/article/456899/byod_vs_cyod_bring_choose_your_own_device_/.

Bort, Julie. 2013. Google: There Are 900 Million Android Devices Activated. *Business Insider*. Accessed 15.05.2013.

Brossier, Véronique. 2011. "Developing Android Applications with Adobe AIR." In, ed Mary Treseler: O'Reilly Media, Inc. (accessed 07.06.2015).

Burrows, Ashlee Vance and Peter. 2011. "Do Not Anger the Alpha Android." *Blomberg Businessweek*, 30.04.2011.

Camden, Raymon, and Andy Matthews. 2012. "jQuery Mobile Web Development Essentials." In: Packt Publishing Ltd.

Charland, Andre, and Brian Leroux. 2011. "Mobile Application Development: Web vs. Native." *Communications of the ACM* 54 (5):49-53.

Cutroni, Justin. 2014. "Hits, Sessions & Users: Understanding Digital Analytics Data." Accessed 03.12.2015. http://cutroni.com/blog/2014/02/05/understanding-digital-analytics-data/.

Easterbrook, Steve, Janice Singer, Margaret-Anne Storey, and Daniela Damian. 2008. Selecting Empirical Methods for Software Engineering Research

Elektronikkbransjen. 2007. Domsavgjørelse, mobil levetid.

Elksten, M. 2014. "Learn REST: A tutorial." Accessed 24.01.2014. http://rest.elkstein.org/.

Engadget. 2010. Hardware fragmentation on Android phones (menu buttons).

Fette, Ian, and Alexey Melnikov. 2011. "RFC 6455 The WebSocket Protocol ". Internet Engineering Task Force (IEFT), Last Modified Desember 2011. http://tools.ietf.org/html/rfc6455.

Gartner, Inc. 2010. "Mobile sales statistics by operating system (2008/2009)." Last Modified February 23. http://www.gartner.com/it/page.jsp?id=1306513.

Gartner, Inc. 2011a. "Mobile sales statistics by operating system (1Q 2011)." Last Modified May 19. http://www.gartner.com/it/page.jsp?id=1689814.
Gartner, Inc. 2011b. "Mobile sales statistics by operating system (2Q 2011)." Last Modified August 11. http://www.gartner.com/it/page.jsp?id=1764714.
Gartner, Inc. 2011c. "Mobile sales statistics by operating system (3Q 2011)." Last Modified November 15. http://www.gartner.com/it/page.jsp?id=1848514.
Gartner, Inc. 2011d. "Mobile sales statistics by operating system (2009/2010)." Last Modified February 9. http://www.gartner.com/it/page.jsp?id=1543014
Gartner, Inc. 2012a. "Mobile sales statistics by operating system (1Q 2012)." Last Modified May 16. http://www.gartner.com/it/page.jsp?id=2017015.
Gartner, Inc. 2012b. "Mobile sales statistics by operating system (2Q 2012)." Last Modified August 14. http://www.gartner.com/it/page.jsp?id=2120015.
Gartner, Inc. 2012c. "Mobile sales statistics by operating system (3Q 2012)." Last Modified November 14, 2012. http://www.gartner.com/newsroom/id/2237315.
Gartner, Inc. 2012d. "Mobile sales statistics by operating system (4Q 2011)." Last Modified February 15. http://www.gartner.com/it/page.jsp?id=1924314
Gartner, Inc. 2013a. "Mobile sales statistics by operating system (1Q 2013)." Last Modified May 14, 2013. http://www.gartner.com/newsroom/id/2482816.
Gartner, Inc. 2013b. "Mobile sales statistics by operating system (2Q 2013)." Last Modified August 14, 2013. http://www.gartner.com/newsroom/id/2573415.
Global Intelligence Alliance. 2010. Native or Web application? How Best to Deliver Content and Services to Your Audiences over the Mobile Phone. Accessed April 2010.
Google. 2012a. "Android platform versions." Google Accessed 02.04.2012. http://developer.android.com/resources/dashboard/platform-versions.html
Google. 2012b. "Developer guide: API Level." Google. http://developer.android.com/guide/topics/manifest/uses-sdk-element.html#ApiLevels.
Google. 2012c. "Developer guide: Supporting multiple screens." Google. http://developer.android.com/guide/practices/screens_support.html.
Google. 2013. Android Distribution. Google.
Google. 2014. "Open distribution." Google Accessed 01.02.2014. http://developer.android.com/distribute/open.html.
Google. 2015a. "Android Developer Tools." Google Accessed 08.05.2015. https://developer.android.com/tools/help/adt.html.
Google. 2015b. "Download Android Studio and SDK tools." Google Accessed 08.05.2015. https://developer.android.com/sdk/index.html.
Gorschek, Tony, Claes Wohlin, Per Garre, and Stig Larsson. 2006. "A Model for Technology Transfer in Practice." *IEEE Software* 23 (6):88-95. doi: http://dx.doi.org/10.1109/MS.2006.147.
GSM Arena. 2015a. "Apple iPhone 4." Accessed 21.11.2015. http://www.gsmarena.com/apple_iphone_4-3275.php#.
GSM Arena. 2015b. "Apple iPhone 4s." Accessed 21.11.2015. http://www.gsmarena.com/apple_iphone_4s-4212.php.
GSM Arena. 2015c. "Apple iPhone 5." Accessed 21.11.2015. http://www.gsmarena.com/apple_iphone_5-4910.php.
GSM Arena. 2015d. "HTC Desire HD." Accessed 21.11.2015. http://www.gsmarena.com/htc_desire_hd-3468.php.
GSM Arena. 2015e. "HTC Legend." Accessed 21.11.2015. http://www.gsmarena.com/htc_legend-3141.php.
GSM Arena. 2015f. "HTC One S." Accessed 21.11.2015. http://www.gsmarena.com/htc_one_s-4574.php.
GSM Arena. 2015g. "Nokia 3720." Accessed 21.11.2015. http://www.gsmarena.com/nokia_3720_classic-2867.php.
GSM Arena. 2015h. "Nokia C5." Accessed 21.11.2015. http://www.gsmarena.com/nokia_c5-3122.php.
GSM Arena. 2015i. "Nokia Lumia 720." Accessed 21.11.2015. http://www.gsmarena.com/nokia_lumia_720-5321.php.
GSM Arena. 2015j. "Nokia N8." Accessed 21.11.2015. http://www.gsmarena.com/nokia_n8-3252.php.
GSM Arena. 2015k. "Samsung Galaxy S2." Accessed 21.11.2015. http://www.gsmarena.com/samsung_i9100g_galaxy_s_ii-4327.php.
GSM Arena. 2015l. "Samsung Galaxy S3." Accessed 21.11.2015. http://www.gsmarena.com/samsung_i9300_galaxy_s_iii-4238.php.

Hadlock, Kris. 2012. "jQuery Mobile: Develop and design." In, ed Peachpit Press. http://ptgmedia.pearsoncmg.com/images/9780321820419/samplepages/032182041X.pdf.
Hess, Ken. 2014. "How to Evaluate Mobile Management Solutions." Accessed 01.11.2015. http://www.tomsitpro.com/articles/evaluating-mobile-management-solutions,2-708-2.html.
Holzer, Adrian, and Jan Ondrus. 2010. "Mobile application market: A developer's perspective." *Telematics and Informatics* (28):10.
Httpful. 2015. "Httpful: The REST Friendly PHP HTTP Client Library." Accessed 26.11.2014. http://phphttpclient.com/.
Jonathan L. Krein, Alexander C. MacLean, Charles D. Knutson, Daniel P. Delorey, and Dennis L. Eggett. 2010. "Impact of Programming Language Fragmentation on Developer Productivity: A Sourceforge Empirical Study." *International Journal of Open Source Software and Processes (IJOSSP)* 2 (2):41-61. doi: 10.4018/jossp.2010040104.
JSON. 2014. "Introducing JSON." Accessed 17.01.2014. http://www.json.org/.
Kai Petersen, Cigdem Gencel, Negin Asghari, Dejan Baca, and Stefanie Betz. 2014. "Action research as a model for industry-academia collaboration in the software engineering context." International Workshop on Long-term Industrial Collaboration on Software Engineering, Västerås.
Kothari, C.R. 2004. "Research Methodology Methods and Techniques." In: New Age International (P) Ltd. http://www.suza.ac.tz/saris/download/132376585119680689-Research-MethodologyMethods-and-Techniques-by-CR-Kothari.pdf%202.pdf (accessed 18.06.2015).
Lane, Jon. 2012. "How does the Internet work." Accessed 28.02.2012. http://www.w3.org/wiki/How_does_the_Internet_work#Server-side_languages.
Lardinois, Frederic. 2014. "Appcelerator Makes Its Platform More Flexible With Native SDK Support, API Builder And Updated Analytics." Last Modified 17.07.2014. http://techcrunch.com/2014/07/17/appcelerator-makes-its-platform-more-flexible-with-native-sdk-support-api-builder-and-updated-analytics/.
Leenheer, Niels. 2012. "HTML5 browser support test." Accessed 24.02.2012. http://www.html5test.com.
Lerman, Julia. 2010. "Programming Entity Framework." In, ed Mike Hendrickson and Laurel Ruma: O'Reilly Media, Inc. http://sd.blackball.lv/library/Programming_Entity_Framework_-_Building_Data_Centric_Apps_with_the_ADO.NET_Entity_Framework_2nd_Edition.pdf.
Liao, Sean. 2014. *Migrating to Android for iOS developers*: Apress.
Lionbridge. 2012. Mobile Web Apps vs. Mobile Native Apps: How to Make the Right Choice. 13. Accessed 01.11.2013.
Löwy, Juval. 2010. "Programming WCF Services." In, ed Mike Hendrickson and Laurel Ruma: O'Reilly Media, Inc. . https://ashishmit99.files.wordpress.com/2013/01/oreilly-programming-wcf-services-3rd-edition-aug-2010.pdf.
MEDIENORGE. 2013. "Andel som har smarttelefoner." http://www.medienorge.uib.no/?cat=statistikk&medium=it&aspekt=tilgang%20og%20bruk&queryID=379.
Microsoft. 2013. "What Is Windows Communication Foundation." Accessed 22.05. http://msdn.microsoft.com/en-us/library/ms731082.aspx.
Microsoft. 2015. "Windows Dev Center." Accessed 09.05.2015. https://dev.windows.com/en-US/.
Mobilevenue. 2010. All iPhone models compared for speed.
Mominis. 2011. "Solving Mobile Fragmentation with Platform Agnostic Ecosystems." http://www.mominis.com/wp-content/uploads/2011/02/Solving-Mobile-Fragmentation-with-Platform-Agnostic-Ecosystems.pdf.
Mozilla. 2012. "About Javascript." https://developer.mozilla.org/en/About_JavaScript.
Nelson, Robert. 2013. "Eric Schmidt: Android seeing 1.5 million activations daily." Last Modified 16.04.2013 Accessed 30.10.2013. http://androidcommunity.com/eric-schmidt-android-seeing-1-5-million-activations-daily-20130416/.
Nokia. 2011. "Nokia and Accenture finalize Symbian software development and support services outsourcing agreement." Accessed 17.10.2012. http://press.nokia.com/2011/06/22/nokia-and-accenture-finalize-symbian-software-development-and-support-services-outsourcing-agreement/.
Nokia. 2012. "Welcome to the third ecosystem." http://conversations.nokia.com/2011/02/11/welcome-to-the-third-ecosystem/.
Opera. 2012. "Opera mobile." Accessed 28.02.2012. http://www.opera.com/mobile/specs/.
PC Magazine. 2012. "Encyclopedia." PC Magazine Accessed 29.02.2012. http://www.pcmag.com/encyclopedia_term/0,2542,t=Smartphone&i=51537,00.asp.

PC Magazine. 2015. "Encyclopedia." PC Magazine Accessed 24.10.2015. http://www.pcmag.com/encyclopedia/term/52520/tablet-computer.
PC World, Norge. 2012. "Stadig nye rekorder for Android." http://www.idg.no/pcworld/article239599.ece.
PhoneGap. 2015a. "FAQs." Accessed 18.12.2015. http://phonegap.com/about/faq/.
PhoneGap. 2015b. "Supported Features." Accessed 04.06.2015.
pxldot. 2012. "iOS ebb and flow." Last Modified 04.03.2012. http://pxldot.com/18754186750.
Rajapakse, Damith C. 2008a. "Fragmentation of mobile applications." http://www.comp.nus.edu.sg/~damithch/df/device-fragmentation.htm.
Rajapakse, Damith C. 2008b. "Techniques for De-fragmenting Mobile Applications: A Taxonomy." Intl. Conf. on Software Engineering and Knowledge Engineering Conference (SEKE'08), San Francisco, USA.
Research In Motion, BlackBerry. 2012a. "Platform Choice." https://developer.blackberry.com/platform/.
Research In Motion, BlackBerry. 2012b. "Why BlackBerry?". https://developer.blackberry.com/whyblackberry/.
Runeson, Per, and Martin Höst. 2008. "Guidelines for conducting and reporting case study research in software engineering " *Empirical Software Engineering* (14):131-164.
Samsung. 2012. Samsung Galaxy Note II.
Santos, Paulo Sergio Medeiros dos, and Guilherme Horta Travassos. 2011. "Action Research Can Swing the Balance in Experimental Software Engineering." *Advances in Computers*:77. doi: 10.1016/B978-0-12-385510-7.00005-9.
Sarah Allen, Vidal Graupera, Lee Lundrigan. 2010. "Pro Smartphone Cross-Platform Development: iPhone, BlackBerry, Windows Mobile and Android Development and Distribution." In: Apress.
Saunders, Aaron. 2015. *Building CrossPlatform Apps Using Titanium™, Alloy, and Appcelerator® Cloud Services*: John Wiley & Sons Ltd.
Seaman, Carolyn B. 1999. "Qualitative Methods in Empirical Studies of Software Engineering." *IEEE TRANSACTIONS ON SOFTWARE ENGINEERING* 25 (4):17.
Sekaran, Uma. 2003. *RESEARCH METHODS FOR BUSINESS*. Fourt ed: John Wiley & Sons, Inc. .
Shotts, Kerri. 2014a. "PhoneGap 3.x Mobile Application Development HOTSHOT." In: Packt Publishing Ltd. (accessed 05.06.2015).
Shotts, Kerri. 2014b. "PhoneGap for Enterprise: Master the art of building secure enterprise mobile applications using PhoneGap." In: Packt Publishing Ltd. (accessed 04.06.2015).
Shull, Forrest, Janice Singer, and Dag I.K. Sjøberg. 2008. *Guide to Advanced Empirical Software Engineering*: Springer-Verlag London.
Sjøberg, Dag I. K., Tore Dybå, and Magne Jørgensen. 2007. "The Future of Empirical Methods in Software Engineering Research ".
Smith, David. 2012. "iOS 5.1 Upgrade stats." Last Modified 10.04.2012. http://david-smith.org/blog/2012/03/10/ios-5-dot-1-upgrade-stats/.
Statista. 2013. Number of daily activations of Android devices from August 2010 to March 2013. Statista.com: Statista.
Statistics Norway. 2015. "Uførte årsverk." Accessed 28.11.2015. http://www.ssb.no/a/metadata/conceptvariable/vardok/2744/nb.
StatsCounter. 2014a. Top 9 Mobile Browsers from Dec 2008 to Feb 2009 (bar).
StatsCounter. 2014b. Top 9 Mobile Browsers from Dec 2008 to Jan 2014 (bar).
StatsCounter. 2014c. Top 9 Mobile Browsers from Dec 2008 to Jan 2014 (graph).
StatsCounter. 2014d. Top 9 Mobile Browsers from Oct 2013 to Jan 2014 (bar).
Steffens, Jan, and Erik Andersen. 2012. 65 prosent var smarttelefoner.
Susman, Gerald I., and Roger D. Evered. 1978. "An Assessment of the Scientific Merits of Action Research." *Administrative Science Quarterly* 23 (4):582-603. doi: 10.2307/2392581.
Sutton, Jenifer Hyman. 2014. "BYOD, CYOD, COPE: What Does It All Mean?". Business 2 Community, Last Modified 05.10.2014. http://www.business2community.com/mobile-apps/byod-cyod-cope-mean-01025828.
Telenor. 2013. "Dekningskart." http://www.telenor.no/privat/dekning/dekning_data.jsp.
Trice, Andrew. 2012. "PhoneGap explained visually." PhoneGap Accessed 04.06.2015. http://phonegap.com/2012/05/02/phonegap-explained-visually/.
UCWeb. 2014. "UC Browser." http://www.ucweb.com/.
Valmot, Odd Richard. 2015. "NETCOM 4G UTBYGGING." Teknisk ukeblad, Last Modified 12.05.2015 Accessed 10.12.2015. http://www.tu.no/incoming/2015/05/12/for-oppgraderte-netcom-20-basestasjoner-i-uka.-sa-la-de-om-tempoet.
VisionMobile. 2010. Mobile Developer Economics 2010 and Beyond.

VisionMobile. 2011a. Clash of Ecosystems.
VisionMobile. 2011b. Developer Economics 2011: How developers and brands are making money in the mobile app economy.
VisionMobile. 2012a. Cross-platform developer tools 2012. 14.
VisionMobile. 2012b. Developer Economics 2012.
VisionMobile. 2013a. Developer Economics 2013: Developer tools - the foundations of the app economy.
VisionMobile. 2013b. Developer Economics Q3 2013: State of the developer nation.
VisionMobile. 2013c. How can HTML5 compete with Native? Accessed 03.01.2014.
VisionMobile. 2015. State of the Developer Nation Q1 2015. Accessed 06.03.2015.
W3C. 2011. "W3C Confirms May 2011 for HTML5 Last Call, Targets 2014 for HTML5 Standard." http://www.w3.org/2011/02/htmlwg-pr.html.
W3C. 2012a. "HTML & CSS." Accessed 28.02.2012. http://www.w3.org/standards/webdesign/htmlcss.
W3C. 2012b. "Standards for Web Applications on Mobile: February 2012 current state and roadmap." http://www.w3.org/2012/02/mobile-web-app-state/.
W3C. 2014. "Application foundations for the open web plaform." Accessed 10.05.2015. http://www.w3.org/blog/2014/10/application-foundations-for-the-open-web-platform/.
W3Schools. 2012. "AJAX Introduction." Accessed 28.02.2012. http://www.w3schools.com/ajax/ajax_intro.asp.
W3Schools. 2014. "JSON Tutorial." Accessed 01.02.2014. http://www.w3schools.com/json/.
Wagner, Richard. 2009. "Adobe® AIR™ For Dummies®." In: Wiley Publishing, Inc. (accessed 13.06.2015).
Wargo, John M. 2012. "PhoneGap Essentials: building cross-platform mobile apps." In: Pearson Education, Inc. (accessed 04.06.2015).
Wargo, John M. 2014. "Apache Cordova 3 Programming." In: Pearson Education, Inc.
Whinnery, Kevin. 2012. "Comparing Titanium and PhoneGap ". Appcelerator, Last Modified 12.05.2012 Accessed 05.06.2015. http://www.appcelerator.com/blog/2012/05/comparing-titanium-and-phonegap/.
Whitwam, Ryan. 2011. "Why OEMs need months to deliver Android updates to your phone." ExtremeTech, Last Modified 08.12.2011. http://www.extremetech.com/computing/108263-why-oems-need-months-to-deliver-android-updates-to-your-phone.
Winesett, Jeffery. 2010. "Agile Web Application Development with Yii 1.1 and PHP5." In *Fast-track your web application development by harnessing the power of the Yii PHP Framework*: Packt Publishing Ltd. (accessed 26.11.2015).
Wolfsen, Mike. 2013. *Android Developer Tools*. First edition ed: O'Reilly.
Yeung, Edith. 2012. 5 Reasons HTML5 Apps are Hard to Monetize.
Yii Software LLC. 2014. "The Definitive Guide to Yii 2.0." Accessed 18.12.2015. http://www.yiiframework.com/doc-2.0/guide-index.html.
Zeman, Eric. 2012. "What iOS 6 Says About Mobile Fragmentation." *Informationweek - Online*.