Mobile Application Development Approaches: Recommendation for E-commerce Enterprises

Hasan, Md. Mahmudul
Haque, Md. Ariful

2016 Laurea

Laurea University of Applied Sciences

Mobile Application Development Approaches: Recommendation for E-commerce Enterprises

Md. Mahmudul Hasan,
Md. Ariful Haque
Degree Programme in
Business Information Technology
Bachelor’s Thesis
December, 2016

Laurea University of Applied Sciences **Abstract**
Degree Programme in Business Information Technology
Bachelor's Thesis

Md. Mahmudul Hasan, Md. Ariful Haque

Mobile Application Development Approaches: Recommendation for E-commerce Enterprises

Year	2016	Pages	36

Different mobile application development approaches are described with advantages and disadvantages. A predefined methodology is used namely "tool selection matrix" to determine appropriate development approach for a use case company. To measure the company's circumstances from three different perspectives (end user's perspective, developer's perspective and business perspective), interview was arranged with product owner from the company. Interview findings and experts opinions were combined to use as input for "tool selection matrix". Additionally, Data was collected about several developer companies to determine current trends and practices of application development in terms of monetary aspect and usage of technologies. At the end, recommendation was made with regard to appropriate application development approach for the company.

Keywords: Mobile Application Development, Appropriate Mobile Application Development Approach, Native Application Development, Hybrid Application Development, Application Development Approach Comparison, App Development, App Development Approaches.

Table of Contents

1 Introduction: 6
2 Problem Statement 6
3 Literature Review: 7
4 About Synergy Scandic Oy: 8
5 Major Mobile Operating Systems: 9
5.1 Android Operating System: 9
5.1.1 Android Architecture: 9
5.2 IOS Operating System: 12
5.2.1 IOS Architecture: 12
5.3 Windows 10 Mobile: 14
5.3.1 Windows 10 Mobile Architecture: 14
6 Mobile Application Development Approaches: 15
6.1 Native Application Development Process: 15
6.1.1 Android Development: 15
6.1.2 iOS Development: 15
6.1.3 Windows Development: 15
6.2 Web Application Development Process: 16
6.3 Hybrid Application Development Process: 16
7 Advantages and Disadvantages: 19
7.1 Advantages and Disadvantages of Native Approach: 19
7.2 Advantages and Disadvantages of Web Application Approach: 19
7.3 Advantages and Disadvantages of Hybrid Approach: 20
8 Current Practices, Trends and Discussion: 20
9 Recommendation for Senergy Scandic Oy: 22
9.1 The Tool Selection Matrix: 23
9.2 Evaluation Criteria: 24
9.2.1 End User's Perspective: 24
9.2.2 Developer's Perspective: 25
9.2.3 Business Perspective: 25
9.3 Evaluation of Existing Development Approaches: 26
9.4 Interview and Result: 27
9.5 Final Evaluation 29
10 Discussion: 30
11 Conclusion: 31
References 33
Figures 35
Tables 36

1 Introduction:

With the availability of smartphones and mobile devices, currently business organizations can afford unique opportunities for streamlining their internal processes as well as improving, maintaining and extending customer services. Because of the ease in device portability and availability of internet, mobile technologies, as means of marketizing services and attracting customers, often appear to be better than other possible options (IBM 2015). Therefore, many small business organisations, as like as large ones, are considering e-commerce mobile application as a viable technological choice for establishing, developing and fostering customer relationship (Gazdecki 2015).

Each of the major mobile application development approaches has some defining as well as distinctive characteristics. As some of the approaches are continually evolving, their definitions and classifications are subject to considerable controversy. It might be demanding for non-developers and individuals, without previous experience in this discipline, to appreciate the definitions, distinctions and classifications of different approaches and their possible implications. This research attempts to briefly define and classify available mobile application development technologies from technical point of view for business oriented audience group who might not have enough technical background.

Business organizations might encounter difficulty, to some extent, in determining appropriate application development approach in consideration of their specific business set-ups, arrangements and requirements. Because of the constant evolution of development technologies and diversified nature of business circumstances, it appears to be impossible to make or adopt a generic and universal recommendation as well. Therefore, current research advocates for case specific recommendation based upon the circumstances of an organization. In this research, an attempt was made to exemplify a case specific recommendation approach by analyzing business circumstances against expert opinion. Additionally, financial and technological aspects were taken into consideration to examine logical compliance of the recommendation and theoretical findings with actual development practices.

2 Problem Statement

Mobile applications targeted for customers have positive impact on the profitability of small ecommerce businesses (Goodwin, Babin & Cole 2014). As there are substantial variations among mobile application development technologies, it is crucial to choose appropriate one with the consideration of usability, performance, maintenance, reachability, companies monetary policy and so on.

Without a careful decision for mobile application development technology, business might suffer long term financially and promotionally. This paper attempts to narrow down its scope to the independent enterprises who are conducting e-commerce business in Finland (Statistics Finland). The research approach takes Synergy Scandic Oy, an independent enterprise, as representative agent for the entire sample group and undertakes objective analysis to propose a development choice for the company. Appropriate choice for ecommerce mobile application development approach can not only contribute to the revenue by serving end customers, but possibly reduce unnecessary expenses.

Programmers often experience difficulty in making choice of suitable programming language for a new project (Spinellis 2006). Startup companies, who plan to begin career in software development for mobile phones, may go through similar kind of indecisions about choosing the right platform or language. Moreover, non-developer business organizations might face such challenge while planning for new mobile application.

This paper further attempts to explore different mobile application development approaches and to analyze the data collected through field research by the researchers to indicate current mobile application development trend. Detailed knowledge of different development approaches may help non-developer businesses making better choices according to their circumstances. Current trend analysis might encourage new business ideas and plans for both start-up developer companies and individuals.

3 Literature Review:

Over the last few years the cross platform mobile development or hybrid development technologies have been evolved significantly. Therefore, hybrid mobile technologies are enjoying a gradual popularity and are deserving the consideration of being an alternative to native development approach. Andrade, Albuquerque, Frota, Silveira and Silva (2015) conducted a case study on a Brazilian company to examine the user experience variation in native and hybrid applications. The outcome of the case study revealed that, around 87 per cent of the users could not identify the difference between native and hybrid application. The researchers argued that hybrid development approach could be a viable alternative for small companies with a potentiality of providing similar level of service to the end customers. Developing native application for each mobile platforms appears to be taxing for large organizations as well. Before the availability of cross platform application development frameworks, corporation like Google encountered difficulty of hosting their applications in different App Stores of respective mobile platforms (Nuttall 2009).

Some developers might be willing to get the advantage of both approaches while not losing the direct access to native interfaces. In 2011, Charland merely proposed an approach for native developers to streamline the process of multiple platform oriented development of a single application. Later, Acord (2012) experimented and further extended this idea and named it as ‘Unified Design Process’. Rather than classifying as conventional ‘hybrid development’, this proposed approach could better be described as ‘hybrid design’ where the researchers attempted to reveal the underlying similarities between two operating systems (iOS and Android) and to propose a reconciliation for some of the core stylistic differences (ie. implementing MVC pattern to Android application development which is not natural for the operating system). With the revealed similarities, they identified few design features and discovered a methodology that would direct to maintaining a common application development design and only delegate the design to a specific platform when there is a genuine difference. According to the researchers, this approach would not only secure the application better than that of ‘hybrid development’, but would allow the freedom of not being confined to any third party API.

Low performance has notable negative impact on application usage. Everts (2015) pointed out, as finding of an experiment, that even one second delay can deter significant number of visitors from application. In another survey, participants considered usability (41%) and performance (33%) as their top most priorities for a high-quality application (Nitze & Schmietendorf 2015). Furthermore, Rösler, Nitze, and Schmietendorf (2014) argued that performance should not be compromised by monetary convenience. Others implicitly attempted to address this issue by recommending customized solutions based upon particular circumstances. Ottka (2015) argued that cross-platform solutions could be an alternative for native development provided that the features and context of the application necessitate it. Opposing the con-hybrid accounts, some considered hybrid development approaches are mature enough to compete with their native counterparts (Adinugroho, Reina & Gautama 2015) . Nitze and Schmietendorf (2013) recommended enterprises to adopt hybrid development approach for better utilization of their resources.

4 About Synergy Scandic Oy:

According to Statistics Finland, Synergy Scandic Oy belongs to the category of “independent enterprises” as neither its capital nor voting rights are owned by any other enterprise (Statistics Finland). The company was established in October 2014. Currently it has one selling outlet in Pietarsaari and one e-commerce website. Anyone can visit their e-commerce website “http://www.ebazaar.fi/” and order desired product online. The company has its own service for delivering the products to respective customers. Currently the delivery service covers Helsinki metropolitan area, Tampere and Pietarsaari. Additionally, customers can visit the outlet, situated in Pietarsaari, in person and purchase desired products. The business products primarily consist of Asian grocery targeted for the population living in Finland. Currently the company has five employees and it aims to extend the business with new e-commerce ideas and potentialities.

5 Major Mobile Operating Systems:

As mobile applications of any kind have certain relation and interaction with mobile operating systems, appreciation of underlying architectures might be helpful for later discussion of advantages and disadvantages of different application development approaches.

5.1 Android Operating System:

Android appears to be the best selling mobile operating system and it is developed by Google.inc. According to Gartner (2016), Android achieved 84% of global share in terms of selling in the first quarter of 2016. The primary development of Android was done by Android inc and later in 2005, google purchased the company. At the core of it, there is linux kernel and on top of linux kernel there are other libraries and application frameworks. The first version of the operating system Android 1.0 was released in 2008 and since then, in every six to nine months Google releases an upgraded version of it. Android applications can be downloaded from google play store. According to Statista (2016), google play store has over 2 millions applications available to download.

5.1.1 Android Architecture:

Android operating system can be divided into different layers. Linux kernel sits at the lowest layer, on top of it there are Dalvik virtual machine and other system libraries. Higher layers contain application frameworks and the user application layer. The following image shows a visual representation of android architecture.

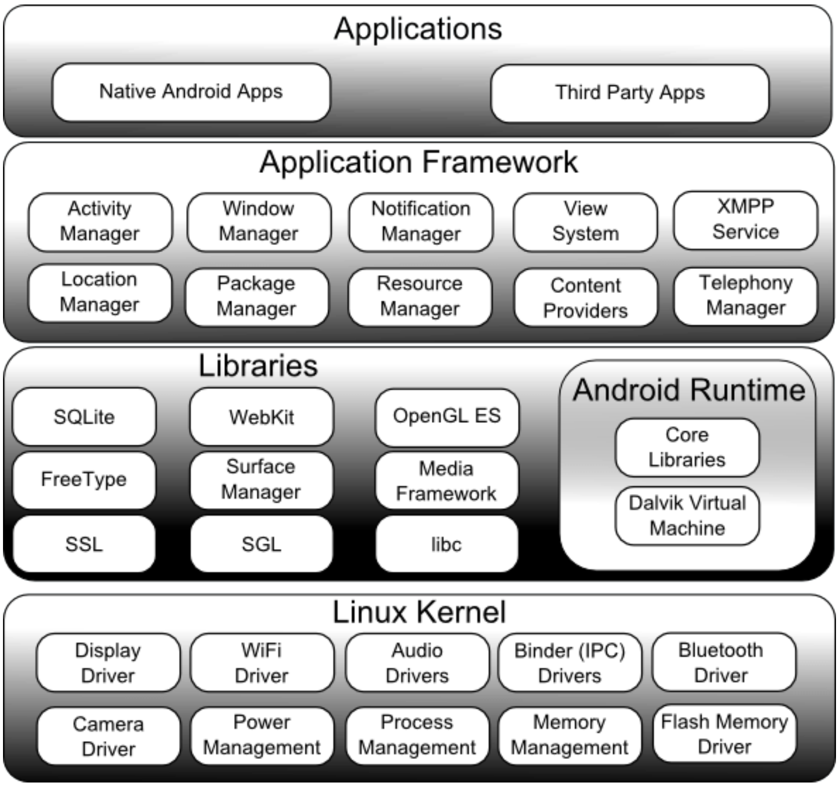


Figure 1: Android Architecture (Smyth 2016, 83)

Android linux is based on kernel 2.6 and it is distinct from desktop or server linux. It uses only kernel with some added features. These new features are called androidism. Shared memory, binder, low memory killer, anonymous shared memory and alarms are some of the important androidisms. Android linux kernel primarily provides drivers for file system access, process management, networking, hardware and so on (Elenkov 2015).

The greater part of android is written in java and these java codes have to run on a java virtual machine like every java application. Androids implementation of java virtual machine is called Dalvik Virtual Machine. Dalvik VM is developed for mobile devices and it can not run normal java bytecodes. Another type of file system called Dalvik Executable (dex) is created to run on it. System java libraries and android applications are nothing more than some collections of dex files. Every time user launches an application it runs as a distinct kernel process inside an instance of virtual machine. As an application is run on virtual machine, it is sandboxed and can not conflict with other applications and this increases the performance. Moreover, getting direct access on device’s hardware becomes impossible and it enhances security of the device (Smyth 2016).

It is important to mention that from android 5.0, Dalvik is replaced by a newer runtime called Android Runtime (ART). Dalvik uses a compilation mechanism called JIT (Just In Time) which interprets every bytecode during the launch of an application. On the other hand, ART uses

AOT (Ahead Of Time) compilation. During the installation of the application all the bytecodes are converted to machine code and installed in a persistent memory. When user launches an application it runs directly from the machine code and do not require further compilation. It results in a quicker execution of the application and decreases the power consumption which increases the battery life of mobile devices (Thakur 2015).

Android core libraries can be divided into different categories such as dalvik VM specific libraries, java interoperability libraries, android libraries and c++ libraries. Dalvik VM specific libraries are used for directly interacting with an instance of Dalvik virtual machine and android application developers generally do not require to use them. Standard java development environment includes a set of core java runtime libraries and these libraries are extensively used by java developers. Android's Java interoperability libraries are a set of those core java runtime libraries that are embraced and transformed for the use of applications that are run inside an instance of Dalvik virtual machine. These libraries are used to performs the tasks of file manipulation, string handling, networking and so on (Smyth 2016).

Android libraries are exclusively used for android application development. These libraries layer provides developers all the required java based libraries to start building an application. They provide database access, 2D-3D graphics drawing, audio video playback, rich user interface building, web browsing capabilities and so on. And these libraries are essentially java wrappers around a set of c/c++ libraries. Android developers generally do not directly interact with c/c++ libraries. They call java wrappers API to perform a specific task and java based API calls c/c++ library and c/c++ library perform the task with linux kernel. But direct access to these libraries also possible through Android Native Development Kit (NDK) (Smyth 2016).

Android Application Framework or Java API Framework is a set of APIs through which entire android OS is exposed to the developers for easily building applications. It includes a view system that provides view and viewgroup objects and using these objects developers can easily build rich user interface for their applications. It also includes key services such as activity manager, notification manager, location manager, content provider etc. (Smyth 2016).

Android applications reside at the top of android architecture. The operating system comes with some core system applications such as email, web browser, messaging, contacts and more. These system applications and other third party applications that are installed by the user reside in this layer (Smyth 2016).

5.2 IOS Operating System:

IOS is developed and solely used by Apple Inc in its mobile devices. It was developed originally in 2007. As plan was devised to manufacture smartphone by Apple Inc, the company decided to customize its existing Mac operating system for the phone. Therefore, it appeared to be convenient for Mac native application developers to build mobile-friendly application. Initially the operating system was named as iPhone OS, but Apple renamed it as iOS with the release of version 4. The iOS kernel is called XNU which was developed by Apple in 2003. XNU is part of Darwin while Darwin is an open-source Unix based operating system also developed by Apple in 2000. By June 2016, the number of available applications in the Apple's App Store has reached to 2 millions (Statista 2016).

5.2.1 IOS Architecture:

The underlying IOS architecture can be classified into four different layers. Core Os layer sits at the bottom of the architecture.The core services are defined at the second layer from the bottom. Media services layer contains the frameworks for audio, video and graphics and it sits at the third layer from the bottom. The top most layer is called Cocoa touch which contains the key technologies for building applications.

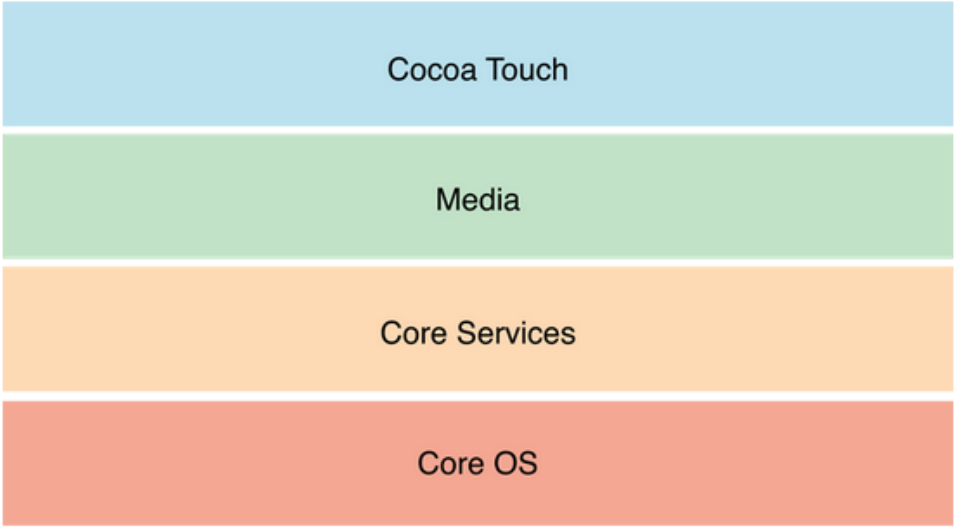


Figure 2: iOS Architecture (Apple developer 2014)

Cocoa Touch:

This layer provides fundamental frameworks for building an IOS application. It provides high level system services such as storyboards, touch based input, multitasking, auto layout, airdrop, push notification, app extensions, handoff and so on. One of the important features in cocoa touch layer is auto layout. It helps the developers to create an user interface easily without writing too many codes. Storyboards provides a way to design the user interface of the application. Multitasking model helps to maximize the battery life of the device. Push notification service is used to provide users with some new information about the application or any form of notification depending on the nature of the application. Notifications can be text based or audible alerts (Apple developer 2014).

Media Layer:

This layer provides all the necessary graphical and audio-video technologies for developing IOS applications. Using these technologies developers can build applications that require high graphical representations. These technologies can be classified into three categories and those are graphics technologies, video technologies and audio technologies. Graphics technologies includes core graphics frameworks, UIkit graphics, core animation, image I/O, photo library, openGLEs and so on. Using these technologies developers can build 2D/3D graphics,animating their content, getting access to users graphics content and more. Audio technologies comprise of core audio, AVfoundation, mediaplayer framework and so on. These technologies help developers to create rich audio content for their applications. Video technologies include AVkit, AVfoundation, coremedia and more. These technologies provide developers ability to create applications that are able to record videos, present and manipulating video content of the devices (Apple developer 2014).

Core Service Layer:

Core service layer provides some high level services as well as core frameworks. High level services includes file sharing support, in-app purchase, icloude storage, data protection and so on. Core frameworks comprise of core location framework, coremedia framework, core data framework, core motion framework, CFnetwork framework, core foundation framework, account frameworks and so on (Apple developer 2014).

Core Os Layer:

Core operating system layer sits at the bottom of the architecture. The features of this layer directly communicate with hardware of the device. Developers generally do not interact with this layer. But all the features that resides upper layers communicate with this layer. In some cases, developers may require direct access to hardware or to address security features, in those cases, they can communicate with the frameworks of this layer. Security frameworks, local authentication frameworks, core bluetooth frameworks are some of the important technologies of this layer (Apple developer 2014).

5.3 Windows 10 Mobile:

Microsoft first launched its mobile operating system called ‘windows phone’ in October 2010. It is closed source and proprietary. The user interface is designed with metro design language which focuses on geometry and typography based design strategy. After releasing several versions of it, Microsoft replaced it with new operating system called ‘Windows 10 Mobile’ in January 2015. The main focus of Windows 10 Mobile is the synchronization of user experiences with different devices such as PCs, tablets, mobile phones and xboxes. Windows 10 Mobile is considered as third largest mobile operating system. According to Gartner, Windows 10 Mobile has 0.7% market share of global mobile operating systems in terms of sale.

5.3.1 Windows 10 Mobile Architecture:

Windows 10 Mobile architecture is divided into different layers.

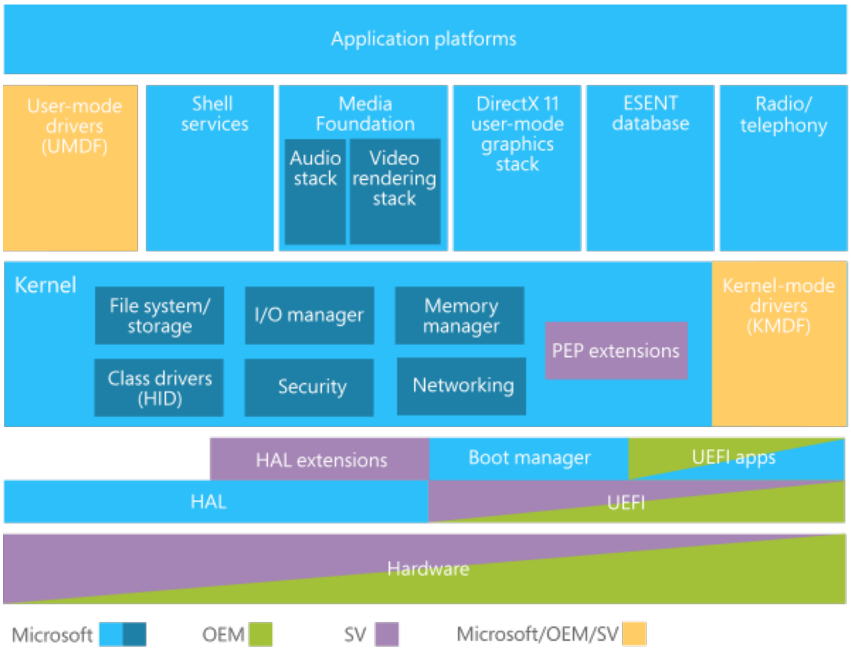


Figure 3: Windows 10 Mobile Architecture (Windows 2015)

The kernel is obtained from windows operating system and it has been modified to work with mobile devices. This layer manages security, networking, storage and other core hardware communications. The layer above is system service and programming framework layer. This layer provides developers all the required frameworks and libraries to build applications. Top most layer contains the system applications that come with operating system and user applications that are downloaded and installed by the user (Windows 2015).

6 Mobile Application Development Approaches:

There are primarily three ways to develop a mobile application and those are Native approach, Web Application approach, and Hybrid approach.

6.1 Native Application Development Process:

Native applications are built using vendor specific programming languages and development toolkits. They are binary executable files that are installed through an app store. Developers write the source codes and compile it to binary forms. They have full access to the hardware functionalities of the device. The following section discusses the native application development process of major companies.

6.1.1 Android Development:

Android applications can be developed using Android Software Development Kit (SDK). Java is the default programming language for android application development. C and C++ programmers also can build android applications using a tool called Native Development Kit (NDK)[1]. All java applications generally runs on Java virtual machine but android applications do not follow this procedure. Up to version 4.4, they run on a separate virtual machine called Dalvik. From version 5.0, all applications run on a new virtual machine called ART (Android Runtime).

Programmers write Java source codes for an application, the source codes are compiled to java bytecodes, then the bytecodes are cross-compiled to a dex file. After that, the apkbuilder is used to package this dex file, resource file and other files into an apk file. This apk file is installed and user runs the application (Kurniawan 2014).

6.1.2 iOS Development:

The required programming languages for iOS development are Swift and Objective-c. It is also possible to develop with C and C++. Once the development is completed, the application must be compiled for iOS. Apple released its initial Software Development Kit (SDK) for iOS in 2008. By the release of Xcode 3.1, it became the default development platform for iOS SDK. Xcode could be used free of charge but test deployment to a device or publishing the application to the App Store requires an annual subscription cost of 99 USD. Once application is published in the App Store, Apple receives thirty per cent of the sales revenue (Apple Developer Program no date).

6.1.3 Windows Development:

Microsoft introduces Universal Windows Platform (UWP) which allows developers to build and run software in different Windows 10 platforms such as mobile, tablet, PC and Xbox. Development tool provided by Microsoft is the Windows 10 SDK which contains Visual Studio Community 2015. Visual studio has emulator for testing. Once the development is finished, respective application packages could be created for the compatible windows devices (ie. mobile, PC or Xbox) (Windows 2016). The virtual marketplace for deploying and selling windows application is called Windows Store. Several programming languages support the development for Universal Windows Platform (UWP). Supported languages could be grouped as following order: (a) HTML, CSS and JavaScript,(b) XAML and C# (c) XAML, DirectX and C++ (d) DirectX and C++. Among the supported languages, group a and b support cross platform development with the help of Cordova and Xamarin (Hissibini 2015).

6.2 Web Application Development Process:

The term "Web Application" is quite ambiguous as it can often be confused with "Web page or Website". Instead, the term "Mobile Web Site", described by Appel (2014), appears to be more precise. However, in the context of mobile application development, this term is often considered as an alternative to native application. Web Applications are designed to be rendered to regular mobile browsers. Therefore, they are built with the consideration to fit into different screen sizes of mobile devices. As they are not designed to be installed in any specific device, developers may have more freedom in development than that of other approaches (ie. native of hybrid). But, maintaining cross browser compatibility could be a concern. Native features like file access and notification are not available to current mobile browsers. Therefore, complicated functionalities, containing these native operations, could not be possibly achieved with Web Application. As the Web Applications depends upon browser only, they can be deployed in a remote server and accessed instantly by typing Uniform Resource Identifier (URL), rather than deploying in a vendor specific application distribution platform (ie. App Store by Apple Inc.). The languages required to build Web Application are HTML 5, JavaScript and CSS.

6.3 Hybrid Application Development Process:

Hybrid applications typically refers to those technologies which are intended to be written once and run into any native platform as like as a native application. Generally these kinds of application are required to be installed in the native platforms. Hybrid Applications are of different kinds and they have different development methods.

Hybrid Application Development and Different Kinds:

As hybrid development approaches are going through constant development and developers are continuously attempting to find more convenient development method, it is challenging to define a solid classification. El-Kassas, Abdullah, Youserf and Wahba (2015) attempted to make an extensive documentation on the taxonomy of different hybrid development approaches. On the other hand, Willocx, Vossaert and Naessens (2016) attempted to generalize all the hybrid development approaches according to their technological characteristics. This thesis would take later approach for describing different kinds of hybrid development approaches. So, hybrid technologies are primarily divided into two kinds:

a) Hybrid Approaches Not Based upon Web Technologies
b) Hybrid Approaches Based upon Web Technologies

a) Hybrid Approaches Not Based upon Web Technologies:

This category of hybrid application development approaches has different variations. Primarily they could be classified as Runtimes and source code translators. Runtimes are layers which lie between the application and respective platforms. Runtime technologies could be further divided into two kinds based upon compilation nature. One kind of runtime technologies does the compilation beforehand and compiles source code to binary code. And this compiled binary code is executed at the runtime by the virtual machine provided by respective platforms. Another kind of runtime technologies runs the source code directly at the runtime without any pre-compilation (Willocx et al. 2016). Example of this technology is Titanium.

Source code translators (most often cross-compiler technologies are referred to as source code translators) uses specific framework to translate bytecode of source platform to source code of the targeted platform. Then this translated source code is compiled by the development tool (ie. Visual Studio of Windows) of the respective platforms (El-Kassas et al. 2015). One example technology is Neomad.

b) Hybrid Approaches Based upon Web Technologies:

This kind of hybrid technologies are built primarily using web technologies like HTML, CSS and JavaScript. HTML is a markup language which is used for building the basic structure of an web based application. JavaScript is a scripting language which, working along with HTML, adds dynamic functionalities to the application. Cascading Style Sheets (CSS) is used for layout and design. These ecosystem of web technologies run on a tabless browser called ‘Webview’ (Webview is discussed elaborately in the subsequent sections). But these web technologies alone can not build a rich mobile application. Mobile applications often require access to the hardware capabilities of the device. To get access the device’s native capabilities, this kind of Hybrid applications provide a wrapper which works like a bridge between native functionalities of the device and the application written using web technologies. The best exemplary technology of this kind is Cordova Framework.

Cordova Framework:

Cordova framework was originally created by Nitobi Corporation in 2009 and Adobe purchased Nitobi in 2011. Later, Adobe released it as an open source software.

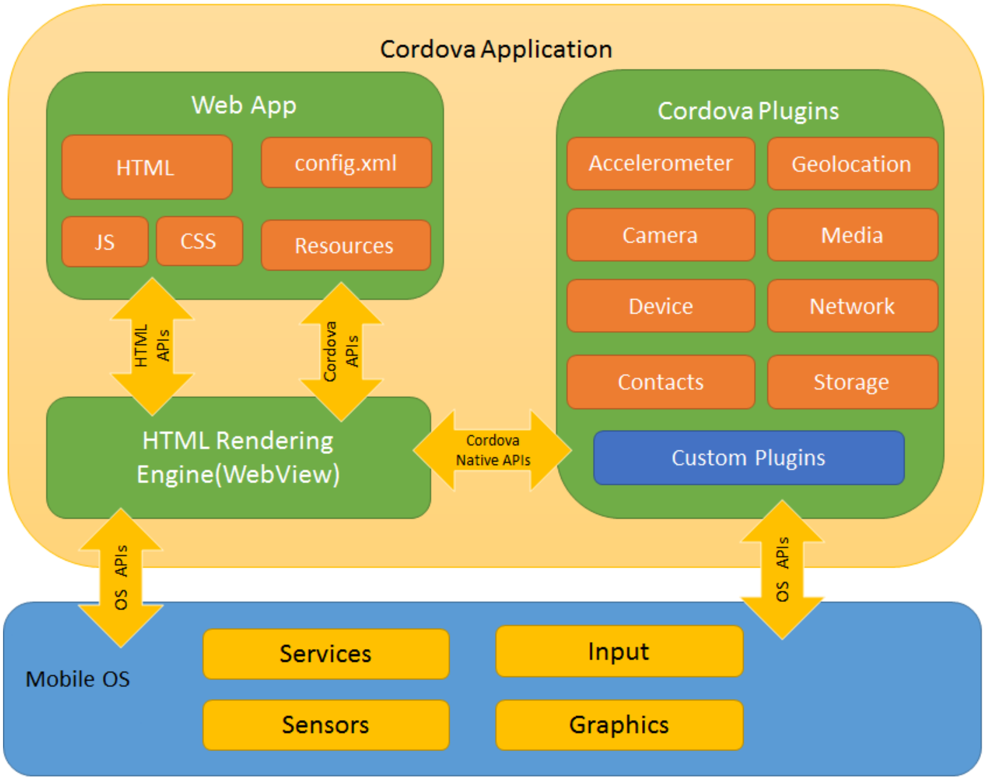


Figure 4: Cordova Framework Architecture (Cordova documentation no date)

Every mobile operating system provides APIs (Application Programming Interface) to build application. These APIs are only accessible to the native compatible programming languages (ie. Java for Android). Cordova framework accesses these respective native APIs and wraps those native APIs for providing common JavaScript APIs regardless of different operating systems and thereby creates bridge between application and native environments of different operating systems. This way hybrid applications can get access to the native device capabilities such as camera, geolocation, accelerometer and so on. To get access each capabilities cordova provides plugins and each plugin is responsible for providing a specific native device functionality. These plugins can be downloaded through npm package management system. Although there are many plugins in nmp but companies can build their own plugins if required (Cordova documentation no date).

WebView:

WebView provides the outlook of the application. Applications are built using HTML, CSS and JavaScript. But native UI frameworks can not run these technologies. Each mobile operating system has an internal tab less browser which is called WebView. Web Developer Petkovski (no date) describes cordova “to be an application container with a webView, which covers the entire screen of the device”. Hybrid applications uses webView to run HTML, CSS and JavaScript. This way, hybrid applications can get the native look and feel.

7 Advantages and Disadvantages:

Every development approach has its own pros and cons. Companies and developers often faces difficulty to choose the right development approach as there is no ideal choice. Selecting a right approach is primarily dependent on companies specifications, budgets, time and the nature of the application itself. The following section discusses advantages and disadvantages of each development approach.

7.1 Advantages and Disadvantages of Native Approach:

As native development approach can make full use of all API calls, it can get access to all functionalities of a device that are offered by the specific vendor. This alone is a significant advantage over other development approaches. As a result, native applications are highly graphical, fast and fluid. Moreover, it has strong community support, plenty of books, online tutorials, exercises and so on.

But native applications are vendor specific. A complete separate application has to be built for each mobile operating system. This is a significant disadvantage of native approach. As a result, to build one application for different platforms, companies always need a different set of programmers for each operating system. This requires a significant amount of resource, time and effort which small companies often can not afford. Moreover, companies frequently want their application to look and feel the same way across multiple platforms. Fulfilling this requirement demands a great deal of effort from the project manager and the developers as well. In addition to, mobile OS companies tend to release new operating system in a short interval compared to the computer operating system, therefore, native applications have to be updated in each new platform release. This requires additional expenditure.

7.2 Advantages and Disadvantages of Web Application Approach:

Web applications are multi-platform supported, easy to develop and well recognized for its stability. As they are developed using an ecosystem of web technologies and can be run through a web browser and every mobile operating system has some kind of web browser, they run everywhere. Any web developer can build mobile web applications. Therefore, developing a mobile web application is easier and cost-effective compared to other approaches. Moreover, web technologies do not tend to change frequently, hence, web applications are more durable. With the release of every new operating system, old web applications do not require the modifications to run as they run through a web browser.

However, web applications have limited access to application programming interfaces. Therefore, many device functionalities are not available to web applications. But W3 standards constantly working on these issues and device functionalities such as geolocation, camera, audio APIs are now available to web browsers (Standards for Web Applications: current state and roadmap 2015). Nevertheless, web applications lack the look and feel of native applications and they resemble look and feel of a web site. This might be an important consideration for some specific applications and web apps may not be the good choice in those cases.

7.3 Advantages and Disadvantages of Hybrid Approach:

Hybrid applications take the advantages of both approaches. As cross-platform applications are web applications inside a webview, it runs in all platforms. At the same time it does not resemble the look and feel of a website instead it mocks the look and feel of a native application. Users generally can not differentiate whether it is hybrid or native application. It reduces the cost, time and effort that are required to hire programmers from different fields for each mobile operating system. Any developer with web development knowledge can build hybrid application. Therefore, hybrid development approach takes relatively less time and it is cost-effective. Since, web development knowledge is common among developers, it is easier to hire web developers than programmers with highly native development skill.

For the last few years, hybrid applications had been criticized for its performance. It was not as fast as native applications. But with advancement of mobile hardware technologies, cross-platform applications now can run relatively faster.

8 Current Practices, Trends and Discussion:

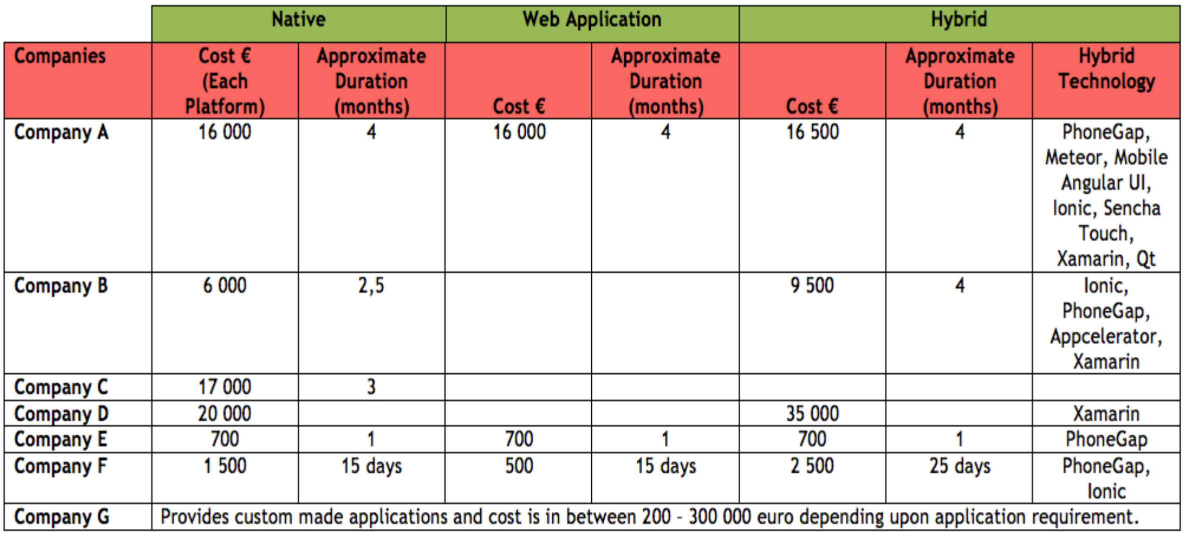
The knowledge of different application development technologies along with their advantages and disadvantages could be further extended by the observation of actual development practices and trends. It might produce some insight as to how enterprises (especially non-developer) could possibly relate theoretical knowledge with the reality. For instance, enterprises could have the opportunity to experience different pricing plans from developer companies and make an assessment of how much of the pricing actually vary between different development approaches. Likewise, how promising and convenient are the non-native development options, from monetary point of view, as alternatives to their native counterparts.

This analysis could indicate some underlying trend in similar approaches adopted by different developer companies .

The researchers attempted to collect pricing data about several mobile application developer companies through e-mail. The query, sent by the researchers, contains the following requirements for an e-commerce mobile application:

- Home Page,
- Page for showing single product with details,
- Shopping Cart,
- Payment System (Card Payment and Account transfer),
- Admin Panel (for uploading, editing and deleting product information),
- Search Option

Data was collected about six application developer companies. These companies are from Finland, Estonia, India and Australia. Data was collected regarding approximate time and cost required by a developer company for developing the requested application. The following table represents the collected data. In case of native development, the table represents the approximate development cost for each native platform (either iOS or Android or Windows). For hybrid development, the table additionally shows the technologies preferred by the companies for building the required application.



	Native		Web Application		Hybrid		
Companies	Cost € (Each Platform)	Approximate Duration (months)	Cost €	Approximate Duration (months)	Cost €	Approximate Duration (months)	Hybrid Technology
Company A	16 000	4	16 000	4	16 500	4	PhoneGap, Meteor, Mobile Angular UI, Ionic, Sencha Touch, Xamarin, Qt
Company B	6 000	2,5			9 500	4	Ionic, PhoneGap, Appcelerator, Xamarin
Company C	17 000	3					
Company D	20 000				35 000		Xamarin
Company E	700	1	700	1	700	1	PhoneGap
Company F	1 500	15 days	500	15 days	2 500	25 days	PhoneGap, Ionic
Company G	Provides custom made applications and cost is in between 200 - 300 000 euro depending upon application requirement.						

Table 1: Collected Data about Developer Companies

There are certain assumptions and limitations of this data collection. As data was collected about companies from four different countries, the researchers converted respective units of currency into one unit of currency for comparison purpose. As the companies about which the data was collected are relatively few in number, the data might not adequately represent the entire sample group. Nonetheless, as a small part of sample group, the collected data could indicate some valuable trend and the implied indications from the collected data could be further supported by theoretical findings and other research outcomes. The researchers intended to collect data about the companies which develop all three kinds of application (Native, Hybrid and Web Application). As Web Application and hybrid development often require same kind of technologies (primarily web based technologies), many developer companies do not explicitly mention about Web Application development expertise in their websites or advertisements. Therefore, the researchers made assumption that the companies who develop Hybrid applications might also develop Web Applications. As an exception to this assumption, Company D uses only a non web based hybrid framework Xamarine and therefore, lack of familiarity could be a reason for not using Web Application approach.

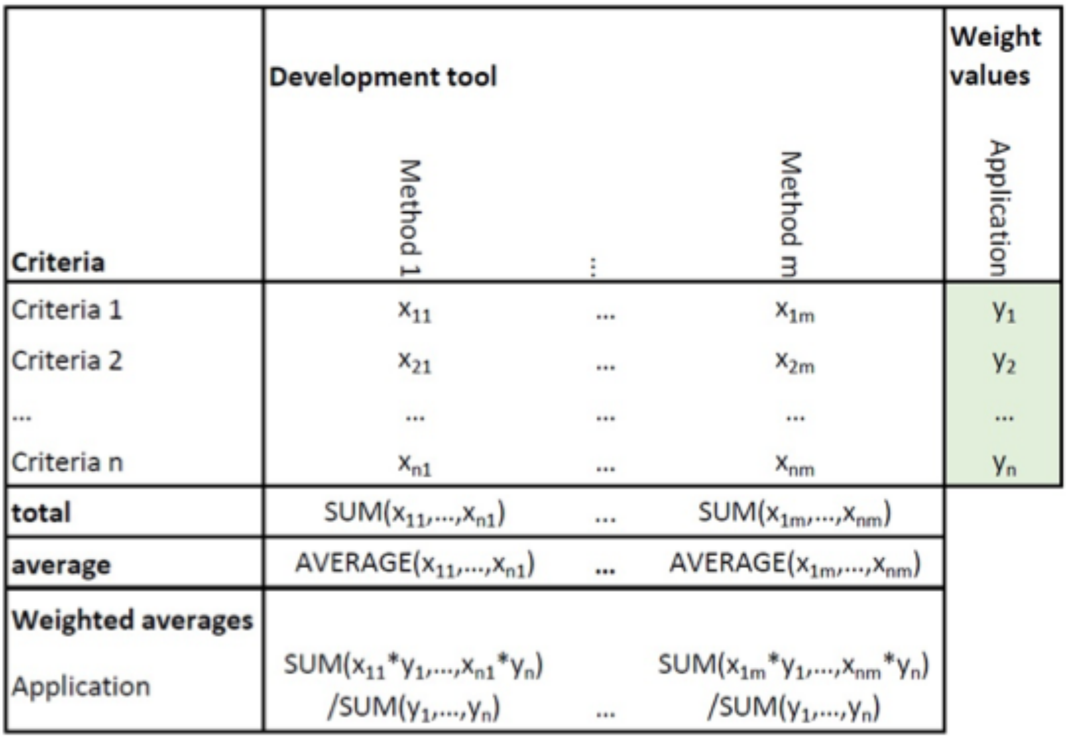
In accordance with previously mentioned benefits, hybrid and web app approaches appear to be relatively cost effective as well as less time and energy consuming. Application development for each native platform costs almost equal to that of non-native development. Therefore, a company promising to build native application for 3 platforms, would likely to bear the expense three times higher than the amount needed for non-native development. As native development approach has no satisfactory alternative in certain aspects such as UI consistency and accessing native functionality, it is recommended to assess the business necessity carefully before making a decision.

The table 1 demonstrates interesting trend about hybrid development. According to the collected data, PhoneGap is the most used technology adopted by four companies. Followed by that, Xamarin and Ionic appear to be used by 3 companies. It is recommended for the companies, interested in having mobile application for longer period of time, to choose relatively popular technology in order to avoid the possible problem of obsolescence. Moreover, as popular technologies are more likely to have stronger community support, maintenance of the application could be easier.

9 Recommendation for Senergy Scandic Oy:

This research lays its foundation based upon a previous research conducted by Ottka (2015). The research purpose was to explore the existing mobile application development technologies and to propose a viable development method for building customer oriented mobile application for a company use case (ABB Ltd.) . In quest for the best suited development approach, Ottka developed a methodology called “tool selection matrix” based upon Decision Matrix method (Belton & Pictet 2012). The researcher applied this methodology to 3 mobile applications (2 of them ware already built and one was planned to be built) of ABB and this approach seems to produce practical and logical outcomes. This thesis intends to follow the same methodological approach to propose the best suited application development strategy in context of Synergy Scandic Oy.

9.1 The Tool Selection Matrix:



Criteria	Development tool Method 1	⋮	Method m	Weight values Application
Criteria 1	x_{11}	...	x_{1m}	y_1
Criteria 2	x_{21}	...	x_{2m}	y_2
...	...	...	...	...
Criteria n	x_{n1}	...	x_{nm}	y_n
total	SUM(x_{11},...,x_{n1})	...	SUM(x_{1m},...,x_{nm})	
average	AVERAGE(x_{11},...,x_{n1})	...	AVERAGE(x_{1m},...,x_{nm})	
Weighted averages Application	SUM(x_{11}*y_1,...,x_{n1}*y_n) /SUM(y_1,...,y_n)	...	SUM(x_{1m}*y_1,...,x_{nm}*y_n) /SUM(y_1,...,y_n)	

Table 2: The Tool Selection Matrix Formula (Ottka 2015, 8)

According to the above mentioned model, the criteria are the different application development features which will be taken into consideration. Development tool section lists different development options as individual method or approach of development. In the actual evaluation, each method's efficacy is supposed to be graded against the corresponding criterion. This section will assess how effective an existing application development tool is in consideration of a particular feature or functionality. Weight values section is supposed to present actual application's score against each development feature or criterion. This section defines, how important the respective criterion is for the planned application. Notably, scoring scale for assessing Development tool and Weight values might differ in the actual evaluation. Weighted average section reveals the final result for the assessment. Each method's performance score and respective application's expectation score are used for calculating the weighted average. And the final score determines that which method is preferable for a particular application.

9.2 Evaluation Criteria:

Evaluation Criteria were chosen by Ottka primarily from other research papers. Some modification were made according to the necessity of ABB project. Those chosen criteria seem to be equally applicable for the current project. Evaluation criteria are divided into three parts: end users' perspective, development perspective and business perspective.

9.2.1 End User's Perspective:

In case of Synergy Scandic Oy, the end users are their existing customers and potential customers. The criteria are presented below as described by Ottka.

UI consistency with the target platform The end-user generally expects an application to behave a certain way on each platform. This includes aspects such as the style of the UI components and the location and behavior of buttons. From the application owner's side the application should be uniform between the platforms so that it is recognizable regardless of platform, and users can easily switch devices and still have the application behave the same way. This criterion examines how well the application matches the native look and feel and how much work is required for doing so.
Interaction and responsiveness Represents the overall speed and performance of the final applications. This includes for instance how quickly the application responds to inputs, moves between views and how well it can utilize the touch-based interaction model of mobile devices. Some cross-platform interfaces cause latency, because the commands must cross an additional abstraction layer. Also, some tools cannot utilize multi-touch functions without additional tools or libraries.
Supported platforms This criterion examines the range of platforms that are available for the method and how feasible it is to support multiple platforms with the method.
Energy consumption High memory inefficiency and mobile data usage increase the energy consumption of the application. While the application's actual context and implementation have the biggest impact on its energy consumption, the type of the application also affects it.

Table 3: Criteria based on the application's end-user's perspective (Ottka 2015, 26)

9.2.2 Developer’s Perspective:

According to Ottka, “This category examines how the choice of method affects the development speed and availability of proficient developers.” As discussed earlier in this paper, native approach requires platform specific development while hybrid approaches allow code sharing. But, UI design phase needs to be done separately even in hybrid development approach.

Shared code Reusing the same code for multiple platforms significantly reduces development time and makes targeting other platforms easier. This criterion examines how much of the code base can be common between the different platforms.
Access to native functionality Cross-platform development tools allow developers to use custom APIs for accessing the device’s native functionality platform-independently. Therefore, the array of available native functionalities depends on the tool and how quickly it is updated when new features for the platform are released.
Competence availability This criteria examines the availability of the skillset required for using the tool. Choosing a tool that uses uncommon programming languages or requires extensive framework-specific knowledge limits the number of available qualified developers. Training developers adds additional costs and slows down the development process.
Ease of design The design and prototyping process differs between the different methods. Quick prototyping improves the development process and allows the developers to easily share ideas and plans for the application.

Table 4: Criteria based on developer's perspective (Ottka 2015, 28)

9.2.3 Business Perspective:

This section deals with the development issues which are mostly related to business affairs.

Cost criterion examines the total cost of developing the application for multiple platforms using the method. This includes also subscriptions and additional software needed.

Long-term feasibility

A mobile application in industrial use is likely to have much longer life-cycle than mobile applications have on average. Developing the application with a specific tool means that it needs to be supported and updated using the same tool, as porting it to some other tool is generally not possible or requires a heavy amount of manual work. This criterions examines the long-term feasibility of the tool based on how actively it is updated, how well it supports newest version of mobile platforms, and whether it has an active community and commercials supporters.

Publishing and distribution

This criterion evaluates how easy it is to distribute the application to end-users and how well the update process works.

Security

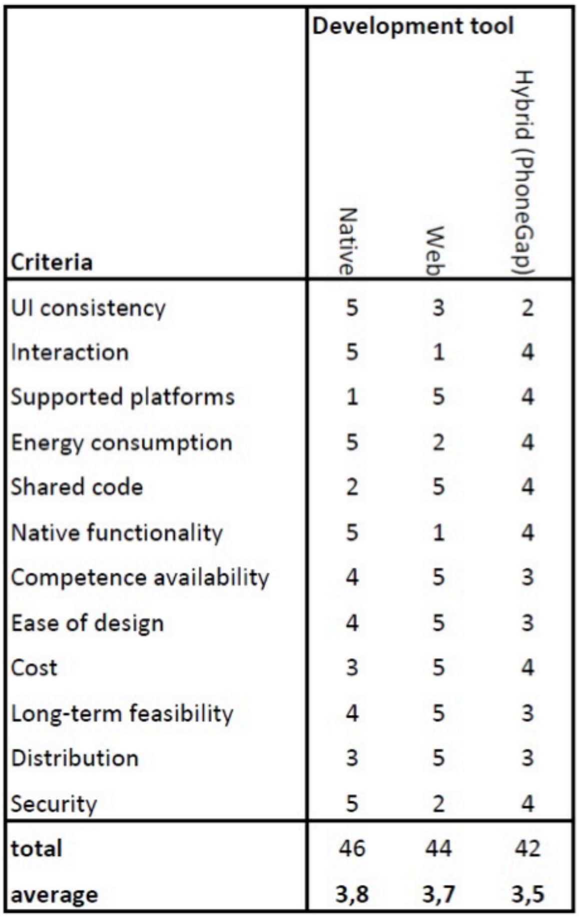
The choice of application type affects the security options available for it, such as encrypted data storage, secure authentication and access to other applications. Industrial applications are likely to play a critical role in large-scale systems and have access to sensitive information so good security is imperative.

Table 5: Criteria based upon business perspective (Ottka 2015, 29)

9.3 Evaluation of Existing Development Approaches:

Three different types of application development approaches were evaluated in the previous research (Ottka, 2015). Evaluated approaches are native development, web application development and hybrid development. Currently there are many hybrid development tools available. One prominent hybrid development tool, phonegap, was selected as a representative of other hybrid approaches.

The tools were evaluated in a scale of 1 through 5 where 1 represents “very poor” quality and 5 represents “very good” status. The tools are evaluated based upon the criteria described in the previous section. Decision was made by interviewing mobile application development experts and with the help of the evaluation carried out by other researches.



	Development tool		
Criteria	Native	Web	Hybrid (PhoneGap)
UI consistency	5	3	2
Interaction	5	1	4
Supported platforms	1	5	4
Energy consumption	5	2	4
Shared code	2	5	4
Native functionality	5	1	4
Competence availability	4	5	3
Ease of design	4	5	3
Cost	3	5	4
Long-term feasibility	4	5	3
Distribution	3	5	3
Security	5	2	4
total	46	44	42
average	**3,8**	**3,7**	**3,5**

Table 6: Development method evaluation results based on literature and interviiews with experts (Ottka 2015, 38)

As the survey revealed, end users might get most pleasant experience using native application. Web applications are best in terms of platform support as this feature can cover larger number of users. Hybrid approach maintains moderate level of consistency in providing end user satisfaction. In development point of view, web application approach delivers convincing performance as source code might be shared easily and number of qualified web app developers seem to be more significant than that of other development approaches. Interestingly, web apps performance is the lowest in accessing native functionality. Web apps appear to be most beneficial in business perspective. Naturally, native approach ensures the application security most as there is no intermediary layer between application and native APIs.

9.4 Interview and Result:

Synergy Scandic Oy had the requirement of a basic ecommerce application. The application was supposed to have the ability of product exhibition, product management by administrator and transaction once customers make purchasing decision. The application requirements were:

- Home Page,
- Page for showing single product with details,
- Shopping Cart,
- Payment System (Card Payment and Account transfer),
- Admin Panel (for uploading, editing and deleting product information),
- Search Option

The researchers interviewed the target application’s product owner from Synergy Scandic Oy for knowing the priorities of the company on the previously described evaluation criteria. Against each criterion, the interviewee was asked that how important that criterion is for the company on a scale of 1 to 3 (not very important, important and very important) and what is the reason for the evaluation. To dispel any misconception and misunderstanding, the researchers arranged a short talk with the interviewee before the interview. The outcome of the interview is presented in the table below.

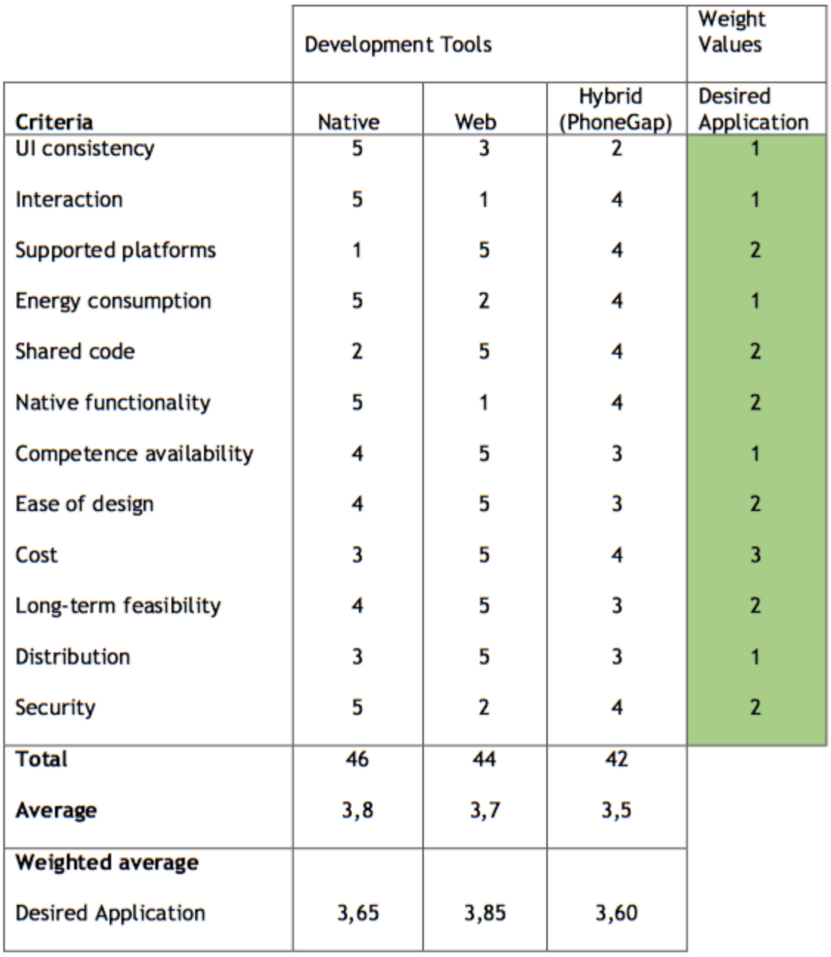
Reason	**Score**
UI consistency with the target platform If the application is not very consistent with the native platform but still serves the purpose, that would be enough for an independent enterprise like Synergy Scandic Oy.	1
Interaction and responsiveness As it will be a small application, that much data processing is not necessary.	1
Supported platforms Most of the company's customers seem to use android. That is why, this is the most priority. Although supporting iOS will be plus point.	2
Energy consumption Not a concern. Because, customers will use the app once in a while.	1
Shared code As it is a small sized independent enterprise, it is preferable that the management would have less worry in future regarding any possible modification.	2
Access to native functionality Using native functionality like camera could make the administration of the application (eg. uploading product picture directly from mobile) easier.	2

Competence availability This is not primary concern for the company	1
Ease of design It can reduce the app building cost and time.	2
Cost As a small sized independent enterprise, it is one of the highest priorities of the company to reduce the cost.	3
Long-term feasibility As an independent enterprise, the management intends to proceed with the resources that would last longer.	2
Publishing and distribution As the customer group is limited, the customers would know about the mobile application from the company's e-commerce website.	1
Security It is expected that the customers would provide the company with their data second time if it is needed due to any security related problem.	2

Table 7: Response from the Interview with Product Owner

9.5 Final Evaluation

The following table demonstrates the weighted average for the desired application at the bottom as well as product owners review collected through interview at the right. The web application achieved the top score of 3.85 followed by native and hybrid approaches respectively.



	Development Tools			Weight Values
Criteria	Native	Web	Hybrid (PhoneGap)	Desired Application
UI consistency	5	3	2	1
Interaction	5	1	4	1
Supported platforms	1	5	4	2
Energy consumption	5	2	4	1
Shared code	2	5	4	2
Native functionality	5	1	4	2
Competence availability	4	5	3	1
Ease of design	4	5	3	2
Cost	3	5	4	3
Long-term feasibility	4	5	3	2
Distribution	3	5	3	1
Security	5	2	4	2
Total	46	44	42	
Average	3,8	3,7	3,5	
Weighted average				
Desired Application	3,65	3,85	3,60	

Table 8: Final Evaluation

10 Discussion:

According to the selection matrix, Web Application is the best approach for the discussed application of Synergy Scandic Oy. Notably, Web Application scored much higher than the other two approaches while native and hybrid approaches achieved ratings very close to each other. Since UI consistency and native look and feel is not a priority, Native approach is not an appropriate choice. Business organizations, in general, aim to marketize their products to largest possible customer groups. Native choice could be a constraint in this regard as only limited number of users may have the privilege of using respective platform-oriented application. Native approach can ensure better application security, but it is not a major priority for Synergy Scandic Oy. Maintenance of finished native application for different platforms could be challenging for the companies who have a prioritized cost-saving policy.

As Hybrid approaches can conveniently support different platforms, any of the hybrid frameworks could be a proposal. But, a significant advantage of hybrid approach, access to the native functionalities, is not the priority for the discussed application. Moreover, determining one particular hybrid framework could be restricting in terms of skillset availability.

Web approach appears to be a cost-saving solution as it performs irrespective of any specific platform. Consequently, it can lack customized design for different platforms. But this deficiency is not significant in this context. Web applications are not fully capable of accessing all native functionalities, but currently HTML5 supports accessibility to native camera features. Therefore, the company requirement of uploading product pictures could possibly be satisfied with Web approach. Even though the company decided not to deploy the mobile application to any of the application stores from different vendors, the nature of web application makes it much easier to access the application through web without any installation. Installed application would not have any benefit over web application in this context as both types of application would need internet to communicate with server and there is no necessity to store any data locally.

As the company considers cost-effectiveness as their highest priority (demonstrated in the table 7), the data collected from different developer companies (demonstrated in the table 1) might produce some insight into this discussion. The table 1 demonstrates Web Application approach to be the most cost-effective option according to the data collected from developer companies. Therefore, final evaluation from the Tool Selection Matrix confirms the previous finding and vice versa.

11 Conclusion:

This research endorses customized recommendation strategy to select a suitable development approach by considering business circumstances. Although this kind of case specific recommendation approach can be chosen for organization of any size, this research effort was only devoted to small sized e-commerce organizations. This study considers expert opinion based evaluation as a standard assessment for development technologies. Along with this standard, the product owner from the company was interviewed to determine the technical circumstances of the company. Then the interview outcome and the standard from experts were combined using a predesigned methodology called "Tool Selection Matrix" to adopt a final recommendation for the company.

As business organizations of small size often have cost effective policy and the Independent Enterprise, presented as a case study for third research, is not an exception, a further research was conducted to explore the pricing for a specific e-commerce application from different developer companies. Although, discussion was initiated about the positive and negative aspects of different existing development approaches, later analyses and insights gained from pricing data collection supported theoretical knowledge.

According to “tool selection matrix”, among the three discussed development approaches, web application is recommended for the company. In spite of the apparently rational outcome, the “tool selection matrix” might have some limitations. Therefore, its efficacy and effectiveness of implication as well as possible limitations could be subject to further research.

References

1. Adinugroho, T. Y., Reina. & Gautama, J. B. 2015. Review of Multi-Platform Mobile Application Development Using WebView: Learning Management System on Mobile Platform. Procedia Computer Science, 59, 291 – 297. Accessed 4 December 2016. http://www.sciencedirect.com/science/article/pii/S1877050915020979
2. Andrade, P R M., Albuquerque, A B., Frota, O F., Silveira R V., Silva, F A. 2015. Cross Platform App A Comperative Study. International Journal of Computer Science & Information Technology (IJCSIT), 7(1). Accessed 30 November 2016. https://pdfs.semanticscholar.org/c28d/223fe09cb97b34dd2f07335315edd874f6cb.pdf
3. Appel, R. 2014. Modern Apps: mobile web sites vs. Native Apps vs. Hybrid Apps. Posted November. Accessed 29 November 2016 https://msdn.microsoft.com/en-gb/magazine/dn818502.aspx
4. Apple Developer Program. No date. Accessed 28 November 2016. https://developer.apple.com/programs/
5. Apple developer. 2014. About the iOS Technologies. Accessed 6 December 2016. https://developer.apple.com/library/content/documentation/Miscellaneous/Conceptual/iPhoneOSTechOverview/Introduction/Introduction.html
6. Belton, V. & Pictet, J. 2012. A Framework for Group Decision Using a MCDA Model: Sharing, Aggregating or Comparing Individual Information? Journal of Decision Systems, 6(3), 283-303.
7. Cordova documentation. No date. Accessed 6 December 2016. https://cordova.apache.org/docs/en/latest/guide/overview/
8. Elenkov, N. 2015. Android Security Internals: An In-Depth Guide to Android's Security Architecture. San Francisco: No Starch Press.
9. El-Kassas, W. S., Abdullah, B. A., Youserf, A. H. & Wahba, A. M. 2015. Taxonomy of Cross-platform Mobile Application Development Approaches. Ain Shams Engineering Journal. Accessed 4 December 2016. http://www.sciencedirect.com/science/article/pii/S2090447915001276
10. Everts, T. 2015. 23 stats you should know about mobile web performance. Posted 21 August. Accessed 29 November 2016. https://www.soasta.com/blog/23-stats-mobile-web-performance-monitoring/
11. Gartner. 2016. Gartner Says Worldwide Smartphone Sales Grew 3.9 Percent in First Quarter of 2016. Accessed 28 November 2016. http://www.gartner.com/newsroom/id/3323017
12. Gazdecki, A. 2015. 22 SMB Industries Cashing in on Mobile Marketing [INFOGRAPHIC]. Posted 11 May. Accessed 30 November 2016. https://www.biznessapps.com/blog/22-smb-industries-cashing-in-on-mobile-marketing-infographic/
13. Goodwin, E., Babin, L & Cole, H. 2014. The Impact of Mobile Apps on Small Business Revenues. Journal of Business and Economics, 5(4), 437-442. Accessed 30 November 2016. http://www.academicstar.us/UploadFile/Picture/2014-6/201461451251651.pdf
14. Hissibini, C. 2015. How to choose the right programming languages for my Universal Windows app project?. Posted 23 February. Accessed 29 November 2016. https://blogs.msdn.microsoft.com/cdndevs/2015/02/23/how-to-choose-the-right-programming-languages-for-my-universal-windows-app-project/
15. IBM. 2015. Fifth Annual UK Online Retail Christmas Readiness Report. Accessed 30 November 2016. http://offers.worldpayglobal.com/rs/850-JOA-856/images/Fifth_Annual_UK_Online_Retail_Christmas_Readiness_Report-2.pdf
16. Kurniawan, B. 2014. *Android application development: A beginner's Tutorial*. 1st edition. Waterloo, AU: Brainy Software. Book from finna. Accessed 4 December 2014. https://laurea.finna.fi/
17. Nitze, A & Schmietendorf, A. 2015. Universal Performance Assessment of Cross-Platform Mobile Applications. Accessed 4 December 2016. https://www.researchgate.net/publication/296700868_Universal_Performance_Assessment_of_Cross-Platform_Mobile_Applications
18. Nuttall, C. 2009. App stores are not the future, says Google. Financial Times, 17 July. Article from Financial Times online. Accessed 29 November. https://www.ft.com/content/b0461cb3-c294-3118-bbe1-1b8a0ef67b5c

19. Ottka, S. 2015. Comparison of mobile application development tools for multi-platform industrial applications. Accessed 4 December 2014. https://aaltodoc.aalto.fi/bitstream/handle/123456789/16008/master_Ottka_Sanna_2015.pdf?sequence=1
20. Petkovski, F. No date. *Apache Cordova Tutorial: Developing mobile applications with Cordova*. Accessed 28 November 2016. https://www.toptal.com/mobile/developing-mobile-applications-with-apache-cordova
21. Rösler, F., Nitze, A. & Schmietendorf, A. 2014. Towards a Mobile Application Performance Benchmark. Accessed 4 December 2016. https://www.researchgate.net/publication/264129149_Towards_a_Mobile_Application_Performance_Benchmark
22. Smyth, N. 2016. Android Studio 2.2 Development Essentials - Android 7 Edition. CreateSpace Independent Publishing Platform.
23. Spinellis, D. 2006. Choosing a Programming Language. the IEEE Computer Society. Accessed 30 November 2016. http://www.academia.edu/2722516/Choosing_a_programming_language
24. Standards for Web Applications on Mobile: current state and roadmap. 2015. Accessed 4 December 2016. https://www.w3.org/Mobile/mobile-web-app-state/
25. Statista. 2016. Number of apps available in leading app stores as of June 2016. Accessed 28 November 2016. http://www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores/
26. Statistics Finland. SME. Accessed 30 November 2016. http://www.stat.fi/meta/kas/pk_yritys_en.html
27. Thakur, A. 2015. Difference between Dalvik and ART runtimes in Android. Posted 6 February. Accessed 4 December 2016. http://opensourceforgeeks.blogspot.fi/2015/02/difference-between-dalvik-and-art.html
28. Willocx, M. Vossaert, J. & Naessens, V. 2016. Comparing Performance Parameters of Mobile App Development Strategies.
29. Windows. 2015. Windows Phone architecture overview. Accessed 6 December 2016. http://sysdev.microsoft.com/en-us/Hardware/oem/docs/Getting_Started/Windows_Phone_architecture_overview
30. Windows. 2016. Getting started with Windows 10 development. Accessed 28 November 2016. https://www.youtube.com/watch?v=0z73oDdO03E

Figures

Figure 1: Android Architecture (Smyth 2016, 83) 10
Figure 2: iOS Architecture (Apple developer 2014) 12
Figure 3: Windows 10 Mobile Architecture (Windows 2015) 14
Figure 4: Cordova Framework Architecture (Cordova documentation no date) 18

Tables

Table 1: Collected Data about Developer Companies 21
Table 2: The Tool Selection Matrix Formula (Ottka 2015, 8) 23
Table 3: Criteria based on the application's end-user's perspective (Ottka 2015, 26) 24
Table 4: Criteria based on developer's perspective (Ottka 2015, 28) 25
Table 5: Criteria based upon business perspective (Ottka 2015, 29) 26
Table 6: Development method evaluation results based on literature and interviiews with experts (Ottka 2015, 38) 27
Table 7: Response from the Interview with Product Owner 29
Table 8: Final Evaluation 30