KTH Computer Science
and Communication

XDK on Mobile

JANNICA THUN

Master thesis at NADA
Supervisor: Jeanette Hellgren Kotaleski
Examiner: Anders Lansner

TRITA xxx yyyy-nn

Abstract

Today, digital media is getting more and more accessible and adapted to the users. In recent years, there has been an increasing interest in developing cross-platform in terms of mobile devices. The concept plays an important role in the development and maintenance of applications by saving time. Accedo Broadband AB has come up with a cross-platform development framework, called the XDK, for HTML-based platforms like Smart TVs targeting VOD applications. It is currently not possible to use for applications running on mobile devices but it is highly desirable in order to accomplish the vision of *build once - deploy on many*.

This master thesis investigates how the XDK can be integrated or co-exist with existing cross-platform frameworks for mobile devices of today. The findings suggest that a hybrid application approach is the most suitable one and in order to demonstrate its feasibility a prototype based on PhoneGap was implemented. The prototype was successful both implementation-wise and performance-wise. Furthermore, it is certain to say that HTML is advancing and the mobile devices are progressing, all together paving the way for even better performances.

Referat

XDK för Mobil

Digital media blir idag mer och mer tillgänglig och anpassad till användarna. Under senare år har det funnits ett ökat intresse för att utveckla plattformsoberoende gällande mobila enheter. Konceptet spelar en viktig roll i utvecklingen samt underhållandet av applikationer genom att spara tid. Accedo Broadband AB har tagit fram ett plattformsoberoende ramverk, vid namn XDK, för HTML-baserade plattformar såsom Smart TVs inriktat på VOD applikationer. Det är för tillfället inte möjligt att använda det för applikationer på mobila enheter men det är mycket önskvärt för att kunna uppnå visionen *build once - deploy on many*.

Detta examensarbete undersöker hur XDK kan integreras eller samexistera med idag befintliga plattformsoberoende ramverk för mobila enheter. De framtagna resultaten tyder på att en hybrid applikationsstrategi är mest lämplig och för att bevisa detta implementerades en prototyp baserad på PhoneGap. Prototypen var lyckad både gällande implementation och prestanda. Dessutom är det säkert att säga att HTML avancerar och de mobila enheterna blir bättre och bättre. Tillsammans banar de väg för ännu bättre prestanda.

Acknowledgements

Kristoffer Vinell, Accedo, for guidance and feedback.

Jeanette Hellgren, KTH, for your guidance and input.

Anders Lansner, KTH, for feedback on the report.

Contents

Chapter 1

Introduction

Digital media is getting more and more accessible and adapted to the users. There are no limits in terms of when to access it or which platform to use. With the different range of digital media devices available today a common concept that is used is cross-platform. A cross-platform framework is a software framework that outputs an application that can be run on several different platforms. This saves time both for developing the application as well as maintaining it. However, it might have its downsides such as worse performance or restricted access to device specific features whereas a more traditional way providing all this is to develop the application separately for each platform. More specifically, developing a native application. The goal is to find a balance.

Accedo Broadband AB is the market leading enabler of TV application solutions, providing applications, tools and services to media companies, consumer electronics and TV operators globally. With a wide range of products and solutions and over 250 customers worldwide, including large brands such as Netflix, Samsung and Spotify, Accedo has provided more than 1000 deployed applications on more than 40 different platforms reaching more than 100 million households [2].

Accedo has come up with a cross-platform development framework for HTML-based platforms like Smart TVs and Set-Top Boxes (STBs) called the Accedo XDK (Cross-platform Development ToolKit). XDK targets large-scale TV applications, or more specifically Video on demand (VOD) applications. It is currently not possible to use it for applications running on mobile devices but it is of course highly desirable in order to accomplish the vision of *build once - deploy on many*.

1.1 Purpose and aim

The purpose of this master thesis is to explore if and how the cross-platform framework XDK can be integrated or co-exist with existing cross-platform frameworks for mobile devices of today. More specifically, the aim is to find out whether the Smart TV-based framework can be used somehow for VOD applications on mobile devices in order to accomplish the vision of *build once - deploy on many*. An analysis of the

different tools, their approach to device abstraction and feature set provided will be made.

1.2 Problem statement

The questions which the thesis will focus on are:

1. How can existing cross-platform development frameworks for mobile devices be integrated into or co-exist with the Accedo XDK 2.x to accomplish the vision of *build once - deploy on many*?
2. How is the user experience of this compared to a native application?

1.3 Scope and limitations

Since this is a master thesis there are some limitations:

- This work will only focus on XDK 2.x.
- The material for this work, such as devices etc, is provided by Accedo.
- The prototype will
  - be implemented on an iPad Mini 2
  - not cover Digital Rights Management (DRM)
  - be a proof-of-concept and will not be production-ready
- The sample for the user study was limited to employees at Accedo.
- There is a restricted set of frameworks to consider when choosing a framework.
- The solution should support three platforms: iOS, Android and Windows Phone.

1.4 Thesis outline

To begin with, the reader is introduced to necessary theory needed to understand this thesis in chapter 2. Concepts such as Video on demand, Smart TV and the Accedo XDK are presented, as well as some of the most popular approaches to choose between when developing mobile applications. After presenting the theory, the thesis moves on to describing the methodology used in chapter 3, followed by its results in chapter 4. Subsequently, the thesis discusses these findings in chapter 5, ending with a conclusion in chapter 6.

Chapter 2

Background

The purpose of this chapter is to provide the necessary theory to understand this thesis. It begins by introducing the reader to the concepts Video on demand and Smart TV followed by explaining how the XDK works. Lastly some of the most popular approaches to choose between when developing mobile applications are presented.

2.1 Video on demand

Video on demand (VOD) is a system that allows users to watch video content when they choose to, instead of having to adapt to a specific broadcast time. Examples of VOD services are Film2home, SVT play, Viaplay, Netflix and HBO [12].

Figure 2.1: Screenshot of Netflix application on an iPad.

A typical VOD application is basically built up of lists, displaying different types of movies or TV shows, along with providing video playback. Another common feature is the electronic programming guide (EPG) that is used to display broadcast programming for current and upcoming programming. Furthermore, in order to

prevent illegal use or copying of the digital content, a class of technologies called Digital Rights Management (DRM) is used.

A screenshot of the Netlix application on an iPad is demonstrated in Figure 2.1. Sliding lists, or more specifically so called carousels, are used to display different movies and TV shows that are available for watching.

2.2 Smart TV

Smart TV, also known as connected TV or hybrid TV, is a term that describes the integration of the Internet with television sets and Set-Top Boxes (STBs). A Smart TV device provides features that differs from the traditional television sets, such as VOD, interactive advertising, voting, games, social networking and other multimedia applications. User-generated content can be saved on a hard drive or in the cloud, allowing users to track and receive notifications about favorite shows, sporting events etc. Furthermore, with some devices the user can use second screen companion devices, such as a smartphone or tablet, spatial gestures input as in Xbox Kinect as well as speech recognition for natural language user interface.

By using the Internet Protocol Television (IPTV) technology, multimedia services can be delivered over the Internet using the Internet Protocol Suite. Apart from television sets and STBs the technology is integrated into devices such as bluray players, game consoles and other devices with a TV as display output. The Smart TV platforms or middleware provides a public SDK (Software Development Kit) allowing third-party developers (such as Accedo) to develop applications as well as application stores for the end users [9, 10].

2.3 Accedo XDK

The media companies and service providers often want to deploy their TV applications on several devices and this is where the Accedo XDK comes in. XDK, a cross-platform development toolkit, is a powerful HTML/JavaScript (JS) development framework targeting browser-based connected TV platforms. It allows deployment of large-scale multi-platform VOD applications, where the application can be maintained through a single code base.

XDK supports a range of TV platforms such as Smart TVs, game consoles, connected media boxes and STBs. Some examples are LG, Samsung, Opera, Philips etc. A device abstraction interface is provided, allowing the application to work on different devices and platforms. This also means that platforms not supported at the initial implementation can be added later, it is future-proof. It is currently not possible to use for applications running on mobile devices.

The XDK architecture is designed to be highly modular. It provides a set of modularized components that are reusable across different applications, such as utilities, focus manager, history manager as well as a set of User Interface (UI) widgets.

XDK conforms to W3C standards on HTML and CSS and is to a large extent written in JavaScript. The applications are developed using the Workstation device abstraction layer, allowing the developer to work on the PC trusting XDK to handle device compatibility [1].

2.4 Mobile development approaches

Here follows some of the most popular approaches to choose between when developing mobile applications [3, 7, 14]. Apart from the first two, native application and web application - mobile website, each of the rest is represented by a cross-platform mobile framework.

2.4.1 Native application

A native mobile application is developed especially for one platform and it can fully access the device features, such as the camera, GPS, contact list, etc. Two well-known examples of this are coding in Objective-C for iOS and Java for Android. The application is installed via the application store (e.g. Google Play or Apple's App Store) and is located at the home screen as an icon.

2.4.2 Web application - mobile website

A web application based on a mobile website is developed using web technologies (HTML, CSS and JS) just like a plain website. However, the one thing that distinguishes this is that it is designed to look like a native application. A plain website can be found by browsing to a URL and the same goes for the web application based on a mobile website. The latter is then "installed" to the home screen on the mobile device by bookmarking the page, consequently resulting in an icon on the home screen similar to a native application.

2.4.3 Web application - built using a UI framework

jQuery Mobile

jQuery and jQuery UI are open-source JavaScript frameworks on which jQuery Mobile is built on top of. jQuery is designed to simplify scripting across browsers whereas jQuery UI provides abstractions for animation and themeable widgets in terms of allowing developers to build interactive web applications [6].

jQuery Mobile, on the other hand, is built specifically for the mobile platform providing a range of touch-optimized widgets. Its main focus is to provide an application with a smooth and mobile-like UI. It is often combined with other frameworks such as PhoneGap due to its inability to, for example, provide access to device specific features and making the application downloadable from the application store.

Listing 2.1 shows an example of a simple list implemented with jQuery Mobile. By importing the jQuery Mobile .js file and adding an attribute 'data-role="listview"'

the list gets the look and feel as if it were a part of a native app.

```
<ul data-role="listview">
    <li><a href="#">Acura</a></li>
    <li><a href="#">Audi</a></li>
    <li><a href="#">BMW</a></li>
    <li><a href="#">Cadillac</a></li>
    <li><a href="#">Ferrari</a></li>
</ul>
```

Listing 2.1: Simple list implemented with jQuery Mobile.

2.4.4 Hybrid application

PhoneGap

In PhoneGap the application is written using web technologies and is implemented like a web page. The web page is then executed in a WebView in the native application wrapper and by using a file called phonegap.js the application can use API bindings to access native features such as the camera etc. The main focus is for the application to become native: making it downloadable in the application store as well as providing access to device specific features. Since PhoneGap does not provide that much UI enhancing features it can be combined with a UI framework [11].

2.4.5 Interpreted application

Appcelerator Titanium

By writing the application in an interpreted programming language and using an abstraction layer to access native APIs the output becomes an interpreted hybrid application. One framework that does this is Appcelerator Titanium. The programming language in Titanium is JavaScript and the abstraction layer is the Titanium API that is written in the targeted device's native language. The JavaScript code is never converted to for example Objective-C or Java, instead the Titanium API acts as a bridge to expose direct access to the native APIs; the JavaScript is mapped to platform specific APIs. The code is evaluated at runtime with the help of a JavaScript interpreter [13].

In Listing 2.2 code on how to implement a simple button using Titanium is illustrated. Basically the application is written in JavaScript from which methods in the Titanium API is called which in turn calls its native counterpart at runtime.

```
var button = Titanium.UI.createButton({
   title: 'Hello',
   top: 10,
   width: 100,
   height: 50
});
button.addEventListener('click',function(e)
{
   Titanium.API.info("You clicked the button");
});
```

Listing 2.2: Simple button implemented with Titanium.

2.4.6 Cross-compiled application

Xamarin

As for Xamarin, the application is written in C#. In contrast to interpreted applications the source code is actually converted into code that runs on the target platform. This is done thanks to a cross-compiler that converts the code to native binaries at compile-time [8].

2.4.7 Summary

Here follows a matrix summarizing some of the basic features of the different cross-platform approaches/frameworks that has been covered.

Characteristics	**Mobile website**	**jQuery Mobile**	**PhoneGap**	**Titanium**	**Xamarin**
Downloadable from the application store	No	No	Yes	Yes	Yes
Access to device specific features	No	No	Yes	Yes	Yes
Free?	Yes	Yes	Yes	Yes	No
Support for iOS, Android and Windows Phone	Yes	Yes	Yes	No, not Windows Phone	Yes
Development languages	JS, HTML and CSS	JS, HTML and CSS	JS, HTML and CSS	JS	C#

Table 2.1: Matrix displaying some basic features of the different cross- platform approaches/frameworks that has been covered.

Chapter 3

Methodology

This chapter describes and discusses the methods used in this thesis.

3.1 Literature study

A literature study was made to gather information about the XDK, VOD applications and the different cross-platform mobile frameworks in the market of today. Furthermore, an analysis was made to see how these frameworks could be integrated into or co-exist with the XDK to produce a mobile VOD application.

3.2 Case study

Based on the outcome of the literature study a prototype was implemented as a proof of concept. More specifically, the implementation was made in order to demonstrate its feasibility, its potential of being used.

In a case study a smaller part of a big process, referred to as a case, is being analyzed and finally said to represent the process as a whole. The conclusions will then be seen as indications. The process can be divided into different categories whereas the researcher will try to choose a case that covers each one if possible [4].

The process for this case study was a typical mobile VOD application implemented using the outcome of the literature study. It was split into its main categories and the aim was to choose a case covering each one.

3.3 Empirical research

One criterion for a mobile application to be approved by a user is the quality of the user experience. The current way of developing mobile VOD applications within Accedo is writing them native. Many claim that the user experience is always better in a native application compared to a non-native one. Consequently, a comparative research focusing on the prototype's speed and smoothness compared to a corresponding native implementation was made. Furthermore, perceived performance is

considered hard to measure whereas a quantitative research was chosen to obtain users' attitudes and opinions.

Sample

As a sample for the study, employees based at the Hong Kong office and the Stockholm office were chosen. They were chosen based on a non-probabilistic sampling technique called *judgemental sampling*. The term refers to when the units to be sampled are based on their knowledge. In the case of this particular study the people in the sample were found to be appropriate due to their expertise within VOD applications.

Comparative research

A comparative research was conducted in order to identify the similarities and differences in the user experience of the two implementations. To make a meaningful research of this kind, a transformation of the attributes of the UI had to be made [4]. By making the UI in both implementations look exactly the same, more or less, the actual user experience would be the only feature that would differ and consequently get the attention of the user.

Quantitative research

A quantitative research is when you gather quantitative data (data in a numerical form such as statistics), often using an interview or questionnaire, in order to measure some predefined variables. When people are being questioned for their attitudes, taste and opinions a questionnaire is prefered, and was thus chosen for this research.

Furthermore, in terms of ensuring a reliable and valid research, there are two main concepts worth considering: *reliability* and *validity*. A research has a high reliability if it is reliable and that if it would be repeated it would give the same result. Moreover, it has a high validity if the research is measuring what it is supposed to measure [5]. These concepts were taken into account while creating the questionnaire.

Chapter 4

Results

This chapter is divided into three main sections, each of which presents the results relating to each of the research methods. As for the literature study, the section revolves around the flowchart illustrated in Figure 4.1 where the different stages are referred to as numbers.

4.1 Literature study

In terms of finding a solution to a mobile VOD application implemented using an integration of XDK and cross-platform mobile frameworks, information was gathered and analyzed. This was followed by breaking the VOD application down into its different cornerstones (1). On this basis an evaluation of what could be used from the XDK was made (2) that consequently generated different criteria for the cross-platform mobile frameworks (3). Finally, suitable frameworks was searched for (4).

The stages are as follows:

1. First of all, a typical mobile VOD application was broken down into its fundamental cornerstones: the UI, the ability to fetch data as well as to provide functionality for authentication (DRM and video playback).

2. Based on the cornerstones, an analysis of the XDK was made. As was pointed out in section 2.4, the key aspect of the XDK is that it is targeting Smart TVs. As a result of this the XDK UI was found to be based too much on interaction with a remote control and would be of no use on a mobile device. Furthermore, as for the user experience in mobile applications written with web technologies, it is desirable for it to behave as if it were native as much as possible, a smooth experience is strived for.

   The data fetching handled by the back-end in the XDK could be used though. Consequently, for this to be applied on a mobile device one criterion for the cross-platform mobile framework is the ability to write in JavaScript.

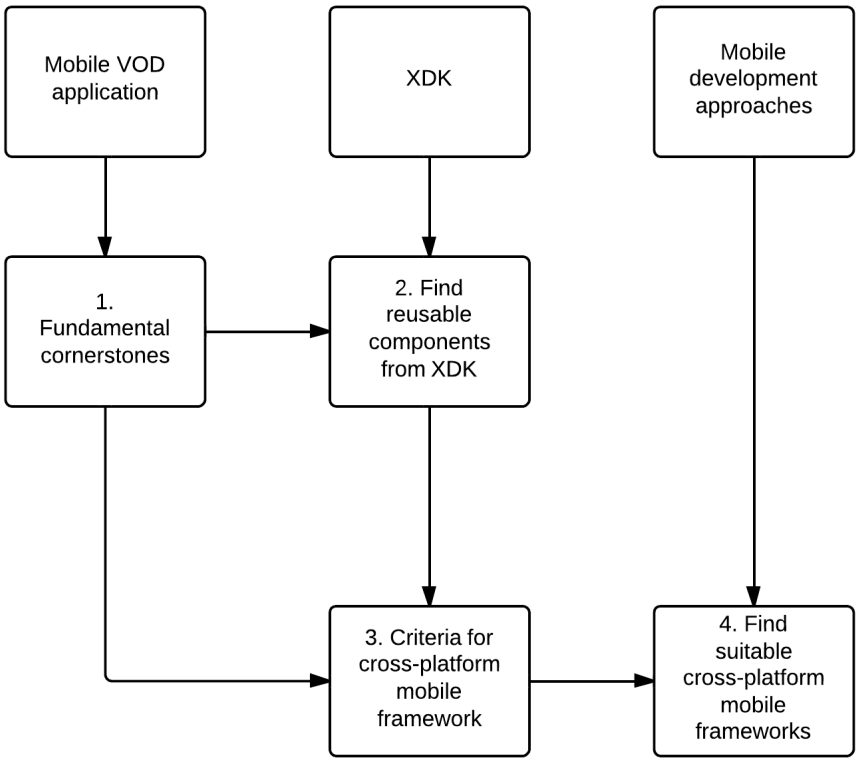


Figure 4.1: A flowchart illustrating the different stages of the literature study.

DRM cannot be handled with web technologies and has to be handled within native code. Consequently the application needs to be native in order to access device specific features as well as making it downloadable from the application store. Even though the DRM code in the XDK cannot be used, existing code supporting DRM for native applications by Accedo might be available depending on the DRM client.

3. The criteria for the cross-platform mobile frameworks are listed in the flow chart in Figure 4.2.

4. The different criteria for the cross-platform framework solution illustrated in Figure 4.2 can be used to narrow our alternatives. As was pointed out in section 2.4 there are several approaches to choose between when choosing which cross-platform framework to use. To easily exclude the nonessential ones the exclusion method was applied.

   Firstly, developing especially for one platform, a native application, is the approach used today within Accedo, but this is not a cross-platform one.

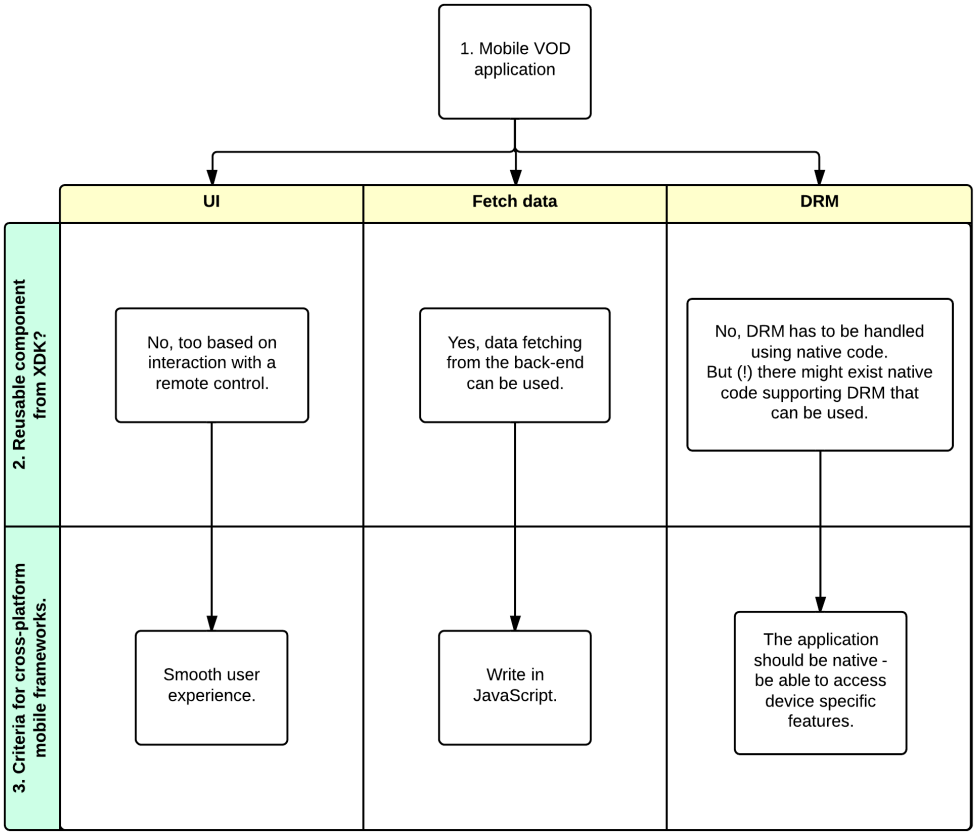


Figure 4.2: A flow chart illustrating the calculation of the different criteria for the cross-platform mobile frameworks.

Furthermore the second and third approach, the web applications, are also ignored since there is a demand for a native application as output.

As far as the remaining approaches is concerned, there exist several frameworks within each. Due to time constraints the wide range of frameworks were reduced down to the ones presented within each approach in section 2.4. In the case of the interpreted application approach, Titanium was mentioned. Despite the fact that Titanium uses JavaScript, the code would have to be modified in order to be integrated correctly with the framework specific APIs. This is time consuming and does not go along with the concept *build once - deploy on many*, thus the approach had to be left out as well. Furthermore the last approach is also disclaimed since the applications are written in C#. Lastly, this leaves the hybrid approach; wrapp a web application as native.

4.2 Case study

As was mentioned earlier the process of this case study, or more specifically what is to be evaluated, is represented by a typical mobile VOD application implemented using the outcome of the literature study. As for the case, the aim was to cover each of the cornerstones explained in the previous section. However, due to time constraints the last one, the DRM, had to be left out. With this said, something that could represent the process and that also works as a stress-test is the EPG. It covers the remaining two cornerstones: the UI, since it is a common UI structure within VOD applications, also handling touch gestures; the data fetching, since it requests and handles a lot of data.

In order to reuse code, an existing EPG implemented using XDK 2.x was searched for. As the upcoming empirical research requires a corresponding native implementation this was also one criterion. Since XDK 2.x was relatively new, it was hard to find an EPG implemented using this version. Finally, a recently made demo project was found that had a corresponding EPG developed using Objective-C for iPad.

Architecture

Based on the literature study a framework producing a hybrid application should be used, one that wraps a web application as native. One framework that supports the three different criteria within the hybrid application approach is PhoneGap. It is undoubtedly the most popular and used framework among the hybrid applications. It is free, open-source, straightforward and easy to set up. Consequently, it was chosen for the prototype.

In Figure 4.3 a basic overview of how a mobile VOD application based on the XDK implemented with PhoneGap would look like is illustrated. The VOD application would be implemented using web technologies and thereafter be executed within a WebView in the native application wrapper. The XDK back-end could be reused for fetching the data and a UI framework could be used to provide a smooth UI. Since DRM has to be handled in native code it could be accessed through a PhoneGap plugin. A PhoneGap plugin bridges a bit of the functionality between the WebView and the native platform where a call from JavaScript can be made to a method in the native code, supposedly implemented to handle the DRM.

Prototype

As a UI framework, the UI plugin IScroll was chosen. It is a multi-platform JavaScript scroller that provides multidimensional scrolling, an expected feature when it comes to EPGs on mobile devices.

In terms of fetching data, instead of using the XDK back-end, some basic JSON requests were used (this is basically what the XDK does as well). JSON is an open standard format that is primarily used to transmit data between a server and web

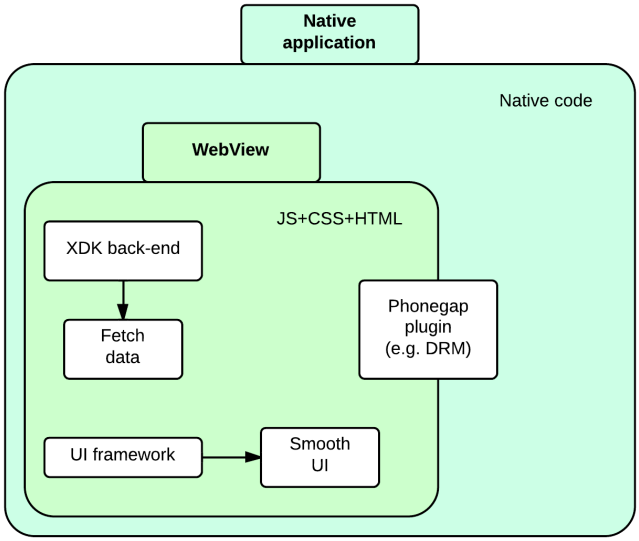


Figure 4.3: A mobile VOD application based on XDK implemented with PhoneGap.

application. The reason behind this was the lack of XDK 2.x material, limited time to create new such material as well as it only concerned a basic and simple prototype. The prototype structure is illustrated in Figure 4.4.

Moreover, something worth mentioning is that some code was reused from the existing browser EPG version. This concerned mainly convenience functions such as calculation of the programs width as well as the different parameters for fetching data.

4.3 Empirical research

The user study evaluated the prototype's user experience compared to the corresponding native ones. More specifically, a focus was put on the scrolling in the two different apps where the participants were asked to answer questions about how smooth it was.

Platform

The native version of the EPG was only targeted for iOS 7.x and iPads whereas the testing device for the user study was chosen among the iPads in the office with that version installed: iPad 4, iPad mini 2 and iPad Air. Due to time constraint only one device was selected. iPad mini 2 and iPad Air were released at the same time and has nearly the same hardware as each other and iPad 4 was released right before them. Since this is only a small start to a possibly much greater project the device

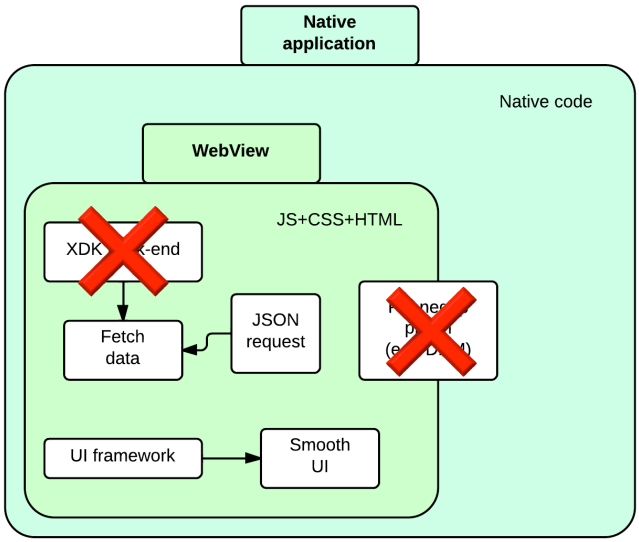
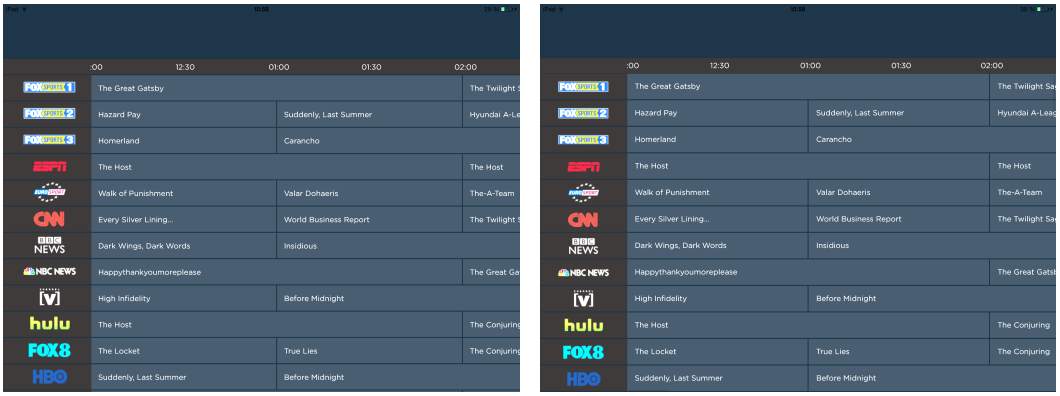


Figure 4.4: Prototype structure.

that are most likely to stay in the market in the near future was chosen, consequently one of the latest ones. Finally iPad mini 2 was chosen based on convenience.

Implementation

For the comparative research the different implementations were designed to look exactly the same. Screenshots of the final result can be found in Figure 4.6.



a) App 1 - Prototype b) App 2 - Native version

Figure 4.6: Screenshots of the two different EPG versions.

The native EPG was originally a part of a complete VOD application but had been modified to only consist of the EPG component where all the extra features

and images had been removed. What was left in terms of user interaction was the scrolling of the data. Even the icons on the home screen had the same appearance. The EPGs consisted of 90 channels vertically and 24 hours horizontally. These numbers were appropriate since the user was now able to scroll the EPG diagonally without any premature stops.

Questionnaire

To ensure the questionnaire's reliability two main things were achieved: the study was well prepared with clear and thorough questions and the procedures were documented in detail. With regard to the first case a pilot test was made on two colleagues, resulting in some clarifications in some of the questions. As for the second case a Google Form was used to gather the information. The respondents were invited by email and their answers were collected in an online spreadsheet. A tour around the offices with the iPad containing the two different applications was made, asking each employee to take a few minutes to try them out and answer the questions. The form included some guidelines, a glossary of used terms and a few questions and can be found in Appendix B. Furthermore, in order to make sure all mandatory questions were filled in they were set with a flag "required" making the form not submittable without filling those in.

Secondly, to ensure validity of the questionnaire it is important that the problem statement is thoroughly decided and analyzed. The purpose of the questionnaire was to answer the second question presented; "How is the user experience of this compared to a native application?". It was therefore split into two different variables: the prototypes user experience and the native implementations user experience. To measure this, several indicators (in terms of questions in the questionnaire) were created for each variable and if these would point in the same direction, the research would have a high validity.

Questionnaire - results

Altogether 74 employees from the Hong Kong office and the Stockholm office participated and the results can be found in Appendix C. In Table 4.1 and Table 4.2 below the mean scores of the questions related to the performance can be found. Since these questions used a predefined set of values, all close to each other, the mean score was chosen instead of median score for representation. Median is usually preferred over mean if the set would have an outlier (an extreme value that differs greatly from the other ones).

In the questionnaire the different implementations were referred to as App 1 and App 2. The reason behind not revealing which was the prototype and which was the native implementation to the participants, was to prevent predefined attitudes of the native versus non-native debate to affect the results.

In Table 4.1 the participants seemed to agree that the prototype is as good as the native implementation. Although more people would prefer the native imple-

Question	**App 1** *mean score*	**App 2** *mean score*
What do you think about the scrolling behaviour? How smooth is it?	4.5	4.4
If you would have downloaded this app from the app store, you would be satisfied with the smoothness of the scrolling.	4.3	4.4

Table 4.1: Mean scores of some questions related to the performance. App 1 refers to the prototype whereas App 2 refers to the native version. The likert scale ranged from one to five where five was the most positive statement.

Question	**App 1**	**App 2**	**They are equally good/bad**
Which app is the best one do you think when it comes to the smoothness of the scrolling?	28%	38%	34%

Table 4.2: Responses measured in percent in terms of a question related to the performance.

mentation over the prototype in Table 4.2 it is a significant percent that thinks that they are equally good.

The questions in Table 4.1 were different ways of measuring the user experience of the scrolling. The mean values in the table shows that the results for each app is very similar. The different indicators are pointing in the same direction, resulting in a statement that the research has a high validity.

Chapter 5

Discussion and future work

This part of the thesis discusses the findings presented in the previous chapter.

Discussion

There are several approaches when choosing a cross-platform mobile framework, and the one that was found to be most suitable for the problem of this thesis work was the hybrid application approach; to wrapp a web application as native. In theory, this would satisfy the three different criteria a mobile VOD application, based on the XDK, should have: a UI framework to provide a smooth user experience, the XDK back-end for fetching data and a bridge to the native APIs in order to handle the DRM.

In order to demonstrate its feasibility a case study was performed. PhoneGap was chosen as a framework providing a theoretical solution fulfilling each of the different criteria. For the implementation though, the XDK 2.x was replaced by some basic JSON requests for fetching data due to a limit of material. Furthermore due to time constraints the DRM part had to be left out.

The case that was chosen to represent a typical VOD application was the EPG and the UI part was handled by the UI plugin IScroll. The prototype was successful both implementation-wise and performance-wise. The PhoneGap solution was easy to set up and the JSON requests to fetch the data from the demo servers were successful. Furthermore, the participants in the user study agreed that the prototype was as good as the native implementation, more or less.

The proof of concept was proved. Of course it is only a simple prototype and when performing a case study the conclusions will only be seen as indications. There are a lot of factors left to be researched and evaluated. For example, the DRM and the video playback was left out, as well as the actual integration with the XDK back-end. In theory, it is supposed to work, but when a fully scaled VOD application is being tested on different platforms, mobile operating systems and versions with different amounts of data, it will most certainly cause new problems.

Furthermore, the performance will of course depend on the frameworks being

used. As for the prototype, PhoneGap and IScroll were chosen. Maybe IScroll will not perform as good on an Android tablet as on an iPad and maybe the WebView in PhoneGap will prevent the application from using the best resources that are available on the device.

Likewise, the perceived performance depends on the user. The people chosen as a sample for the study were narrowed down to employees and they can not be said to represent the whole community of people that uses mobile VOD applications. However, as a large part of the sample work with developing and testing mobile VOD applications every day they can be seen as more harsh in their judgement, meaning that if they approve the average user will as well.

Future

As stated earlier, many claims that the performance is always better in a native application compared to a non-native one. Even so, there is no doubt that HTML will play a critical and dominant role in the future. In addition, the mobile devices are paving the way for even better performances. Furthermore, in this thesis work it is indicated that a non-native approach is as good as a native one regarding performance, more or less.

Along with HTML advancing, a recently released framework, famo.us, claims that their applications will perform as good as native ones with 60 frames per second using just JavaScript. Some people have stated that famo.us will "reinvent the web". Consequently, the non-native approach for developing mobile applications is a solid alternative to the more traditional way of developing in native code.

As for DRM, a crucial feature in a VOD application, several major organizations are planning to integrate DRM within HTML. Rumours indicate that the World Wide Web Consortium (W3C) is supposed to include DRM in the next release of HTML. Furthermore, a collaboration between Google, Microsoft and Netflix has been formed with the intention to create an API that will enable encrypted media in HTML.

Chapter 6

Conclusion

This thesis has shown that it is possible to integrate cross-platform development frameworks for mobile devices with the XDK. The findings suggest that a hybrid application approach is the most suitable one. As far as the user experience is concerned, the case study indicates that a solution including PhoneGap combined with IScroll was as good as a corresponding native implementation.

Despite the fact that the proof of concept was proved, there are a lot of factors left to be researched and evaluated. However, it is certain to say that HTML is advancing and the mobile devices are progressing, altogether paving the way for even better performances.

Bibliography

[1] Accedo Broadband AB. *Accedo XDK 2.x, Solution Description*. 2013.

[2] Accedo Broadband AB. *What we do*. URL: `http://www.accedo.tv` (visited on 05/26/2014).

[3] Rahul Raj C.P and Seshu Babu Tolety. *A study on approaches to build cross-platform mobile applications and criteria to select appropriate approach*. Bangalore, India: Siemens Technology and Services Pvt Ltd, 2012.

[4] R. Ejvegård. *Vetenskaplig metod*. 4th ed. Lund: Studentlitteratur AB, 2009.

[5] A. Eliasson. *Kvantitativ metod*. 3rd ed. Lund: Studentlitteratur AB, 2006.

[6] The jQuery Foundation. *The jQuery Foundation*. URL: `https://jquery.org` (visited on 05/26/2014).

[7] Peter Friese. *Cross-platform mobile development*. URL: `http://cross-platform.mobi/documents/01-overview/01-introduction.html` (visited on 05/26/2014).

[8] Xamarin Inc. *What is Xamarin?* URL: `https://xamarin.com/tour` (visited on 05/26/2014).

[9] *Smart TV*. URL: `http://en.wikipedia.org/wiki/Smart_TV` (visited on 05/26/2014).

[10] Sergios Soursos and Nikos Doulamis. *Connected TV and Beyond*. Intracom S.A. Telecom Solutions, National Technical University of Athens, 2012.

[11] Andrew Trice. *PhoneGap Explained Visually*. URL: `http://phonegap.com/2012/05/02/phonegap-explained-visually/` (visited on 05/26/2014).

[12] *Video on demand*. URL: `http://en.wikipedia.org/wiki/Video_on_demand` (visited on 05/26/2014).

[13] Kevin Whinnery. *Comparing Titanium and PhoneGap*. URL: `http://www.appcelerator.com/blog/2012/05/comparing-titanium-and-phonegap/` (visited on 05/26/2014).

[14] Spyros Xanthopoulos and Stelios Xinogalos. *A Comparative Analysis of Cross-platform Development Approaches for Mobile Applications*. Aristotle University of Thessaloniki: Directorate of Technical Services and Computerization , University of Macedonia: Department of Applied Informatics, 2013.

Appendix A

Acronyms

HTML	Hyper Text Markup Language
EPG	Electronic Program Guide
VOD	Video on demand
JS	JavaScript
XDK	Accedo's Cross-Platform Development Kit
CSS	Cascading Style Sheets
JSON	JavaScript Object Notation
DRM	Digital Rights Management
STB	Set-Top Box
UI	User Interface
API	Application Programming Interface

Appendix B

Questionnaire

This questionnaire is designed to gather information about your attitudes concerning the scrolling of the channel programs in the two different implementations of the EPG.

The results of this survey will be a part of a Master Thesis in Computer Science at the Royal Institute of Technology in Stockholm (KTH).

The survey is voluntary and your answers will be anonymous.

PLEASE do not submit your answers more than one time.

Glossary

EPG	Electronic Program Guide
Scroll	The sliding of text/images across a display that moves the user's view.
Lagging	Choppy, discontinous, rough, uneven, bumpy.
Smooth	The oppposite of lagging: continuous, calm, not rough.

* Required

1. How often do you use a tablet? *

   Mark only one oval.

   - ○ Never
   - ○ Less than once a month
   - ○ Once a month
   - ○ Once a week
   - ○ Everyday or almost everyday

2. What do you think about the scrolling behaviour? How smooth is it? *

   Mark only one oval per row.

	1	2	3	4	5
App1	○	○	○	○	○
App2	○	○	○	○	○

3. If you would have downloaded this app from the app store, you would be satisfied with the smoothness of the scrolling. *

   Mark only one oval per row.

	Strongly disagree	Disagree	Neutral	Agree	Strongly agree
App1	○	○	○	○	○
App2	○	○	○	○	○

4. Which app is the best one do you think when it comes to the smoothness of the scrolling? *

   Mark only one oval.

   - ○ App 1
   - ○ App 2
   - ○ They are equally good/bad

5. Is there something you would like to add that hasn't been covered in the previous questions?

Appendix C

Results

Question	Response	Respondents
How often do you use a tablet?	Never	8
	Less than once a month	10
	Once a month	4
	Once a week	9
	Everyday or almost everyday	43
What do you think about the scrolling behaviour? How smooth is it? (App 1)	1	0
	2	3
	3	5
	4	18
	5	48
What do you think about the scrolling behaviour? How smooth is it? (App 2)	1	0
	2	2
	3	7
	4	21
	5	44
If you would have downloaded this app from the app store, you would be satisfied with the smoothness of the scrolling. (App 1)	Strongly disagree	0
	Disagree	3
	Neutral	8
	Agree	29
	Strongly agree	34
If you would have downloaded this app from the app store, you would be satisfied with the smoothness of the scrolling. (App 2)	Strongly disagree	0
	Disagree	3
	Neutral	2
	Agree	30
	Strongly agree	39
Which app is the best one do you think when it comes to the smoothness of the scrolling?	App 1	21
	App 2	28
	They are equally good/bad	25