First deliverable

State of the art on Android development and choice of a solution

Antonin Lenfant Miguel - Navarro Reinoso

11/10/2012

Contents

I - State of the art of Android development solutions 2
I.1 Native application 2
Android SDK 2
I.2) Cross-platform application 3
I.2.1. Mobile website 3
I.2.2. Hybrid apps 4
I.2.3. Interpreted apps 5
I.2.4. Cross-compilers 6
II - Choice of a solution 7
II.1) Considering the application 7
II.2) Considering the team 8
III - Sources 9

I - State of the art of Android development solutions

We will develop here the different possibilities for developing an Android application. For each one, we will present some of the most popular tools that can be used, and detail its advantages and disadvantages.

I.1 Native application

A native application is the "official" way to develop for a mobile platform, using tools and APIs directly provided by its creator. Since they are specialized, the application will not be able to run on other platforms.

Android SDK

Using the official Android SDK provided by Google, it's possible to create applications using the Java programming language.

The SDK is coupled to the Android Development Tools (ADT) which is composed of a plugin for the Eclipse IDE (Integrated Development Environment) adding functionality such as syntax highlighting and code completion specific for Android (including the XML resource files used for instance to build interfaces), a graphical interface builder, and debugging on an emulator or an actual device.

If a functionality of the application needs to run as fast as possible, it is possible to develop in C or C++ and use it in the application with the Native Development Kit (NDK).

Pros and cons

Advantages

- Best possible performance, which means the app can either do more computing or run on less powerful hardware.
- Ability to make use of all the hardware (Camera, GPS, Accelerometer, Microphone, etc.) and OS APIs.
- Can be distributed via the Play Store for free or a certain price, which gives more exposure to the application since users are used to search the markets first. Furthermore, app marketplaces provide ratings, comments, rankings...
- Interacting with the hardware provides significant app potential.

Drawbacks

- Forced to use the programming language chosen by the targeted operating system's creators.
- Android apps are submitted to the Play Store with little oversight.
- Impossible to easily port the application to another platform.
- Necessity to learn specific skills which may not be used elsewhere.

I.2) Cross-platform application

A cross-platform application will be developed once with one code base (except maybe for platform-specific elements like the interface) and it will be possible to port it to various platforms and execute it on them without additional work.

I.2.1. Mobile website

Using HTML5 and its new possibilities (local storage, media playing, graphics drawing...), it's possible to create a website acting almost like an application, and which works even without Internet access with HTML5's Offline Mode.

With Android and most other smartphone operating systems, you can then put a shortcut to the website on your desktop, allowing easy access to it almost like a "real" application.

Pros and cons

Advantages

- HTML5 is capable of significantly reducing development time, since its premise is the flexibility and simplicity in use.
- Using HTML5 for mobile applications facilitates the maintenance and support of applications, since it's the same code for all platforms.
- HTML5 is supported by virtually all browsers present in mobile terminals and will be by other devices: TVs, cars...
- Mobile websites development has a much easier learning curve and can be done by persons already experienced in web development.

Drawbacks

- Currently the browsers of mobile terminals follow different rhythms when implementing entire HTML5 specification (for instance, http://mobilehtml5.org/ shows the various differences between platforms), they also render it differently so it may not look the same on different platforms.
- Segmentation in Android and other platforms must also be taken into account when developing mobile websites because the browser's abilities depends on the version of the operating system.
- The HTML5 standard is still a draft being defined and is expected to be fully defined for 2022.
- Because the application will not have the same user interface than native apps, it will look a bit "foreign" on the device among other apps. The browser's interface will use a large part of the screen.
- HTML5 and Javascript performance really isn't up to par with native applications, especially if you need to have animations.

- You can't access some native APIs such as notifications, sensors (cameras, geolocation, accelerometer...), contacts, file system, etc.
- Those applications can't be submitted to Google Play (ex-Android Market) or its equivalent on other platforms.
- It's almost impossible to do automatic unit testing. There are a few software, but still in their infancy and quite expensive.

I.2.2. Hybrid apps

They are web applications developed in HTML5 Javascript and CSS, packaged as a native application and run in a customized application displaying the web pages without any interface around them (unlike a browser), which makes it look more like an application than a website.

The Frameworks used for development allow access to some native APIs, which can be used to extend an application's functionality by using the device's sensors and OS features.

Adobe PhoneGap

Leader of mobile cross-platform development and based on the open-source Apache Cordova project, PhoneGap allows to create mobile applications using web technologies (HTML5, Javascript and CSS). It's the most used hybrid applications framework and as such it is well documented and supported.

The application can be exported to a good amount of mobile platforms : Android, iOS, BlackBerry, WebOS, Windows Phone 7, Symbian and Bada. Of course, the supported APIs depend on the target operating system.

Mosync

Just like Phonegap, you develop a web application that will be packaged. The big difference is that if a functionality doesn't exist in the Mosync Framework, you can add it using C or C++.

It supports the most popular smartphone operating systems plus Moblin and MeeGo.

Pros and Cons

Since they basically use the same languages, the pros and cons are the same as mobile websites with a few changes.

Advantages

- The application can be submitted and distributed by Google Play and its equivalents.
- Since there isn't an additional interface on top of the application, the developper has access to more screen real estate, which allows for a better usability.
- Access to some native APIs exposed to Javascript by the Framework that would not be available using only HTML5, which makes the application more integrated and permits more functionality.

Drawbacks

- Just like with HTML5, the exposed native APIs depend on the target operating system.
- There are still differences with different devices (available APIs and their implementations, HTML/CSS rendering engine) which may require a partial rewrite of the application in order to port it.

I.2.3. Interpreted apps

Unlike the previous cross-platform development Frameworks, interpreted apps are not developed similarly to a website, but instead using a script language which is then interpreted when the application is running on a device.

Appcelerator Titanium

Allows you to develop native applications using Javascript code which will then be interpreted and executed by a virtual machine. You can use the platform's native controls instead or having them simulated or replaced using HTML and Javascript.

It supports Android, iOS and Blackberry.

Rhomobile Rhodes

Rhomobile is a development suite under the MIT license created by Motorola. It allows to develop multiplatform mobile apps using a MVC pattern inspired from Ruby on Rails, the views being in HTML and the controllers in Ruby. The application is then packaged and run on the device using the RubyVM interpreter.

Rhodes supports development for Android, iOS, BlackBerry, Windows Mobile and Windows Phone 7.

Pros and Cons

Again, they are the same as hybrid applications but with a few differences.

Advantages

- Better performance than hybrid applications.
- Better integration with the operating system since it's sometimes possible to use native controls when creating a graphical user interface.
- Unit testing is possible.

Drawbacks

- Sometimes proprietary or with licences that could be restrictive.
- Often less different OS you can export your application to (since it requires more work to develop an interpreter).
- Still some differences between the different mobile OS that sometimes requires to change parts of the code (for example with Titanium).

I.2.4. Cross-compilers

A step further from interpreted apps, cross-compilers compile the application's source code to native code, allowing it to run as fast as possible while still having a choice for the programming language.

Mono for Android/Monotouch

Xamarin, the creators of Mono, a software allowing cross-platform execution of programs created with the Microsoft .Net Framework, created Monotouch (for iOS) then Mono for Android in order to enable developers to create mobile applications using the C# programming language and Visual Studio.

This solution is popular with companies mainly developing for the Microsoft ecosystem since it uses the same tools and languages they already know. However, it is proprietary and quite expensive.

Adobe Flash Builder

Flash Builder is a development suite created by Adobe to address the lack of Flash support on smartphones. Using it, a developer can have a source code in Actionscript (Adobe Flash's scripting language) and compile it to native code for both iOS and Android.

Pros and Cons

The advantages are basically the same as native apps, with some additional drawbacks.

Advantages

- Native-like performance.
- No need to learn a new programming language.
- Unit testing is supported.

Drawbacks

- Proprietary and expensive.
- Few documentation and free support.

II - Choice of a solution

II.1) Considering the application

First, it's necessary to consider the application's specifics in order to choose a development solution.

The application will be used on tablets provided by an NGO (Terre des Hommes) and used in Burkina Faso, so it is probable that the tablets will not be high-end devices and that they might use a weak processor, which we need to take into account. As we saw, the best performance is attainable by developing either native or cross-compiled applications. Since there is not budget for development tools and cross-compilers are expensive, it's better to create a native application if we want to take this into account. There is also no plan to use devices based on another platform than Android, so multi-platform development, which is the biggest drawback of developing native applications, is not the most important factor here.

Furthermore, the application will be used for medical treatments, so it is absolutely primordial to ensure that it does a good diagnostic and follows exactly the Integrated Management of Childhood Illness (IMCI) protocol because errors could be dangerous or even fatal. Therefore, testing will be one of the focuses of this project : it should be possible to check in depth with various scenarios if the given treatment is the right one. As we saw in the previous part, native development on Android makes it easy to do unit testing, in part because it uses the Java language and there are already many tools dedicated to this purpose (such as JUnit for instance).

The application is going to be used by people who most probably never used a technological device in their lives, even less a computer or smartphone. It is then necessary to take this in consideration when designing the user interface, which has to be intuitive, simple and easy to use. In order to do this, it is necessary to have flexible enough tools that allow a good amount of customization. It is also necessary to focus, at least a bit, on animations because they are believed to help novice users better understand the interface when they are used correctly.

Again, for both those objectives, developing a native application is one of the best choices.

We wondered if it might be possible to re-use at least part of original application, which might have been possible if we had chosen to create a mobile website or a web application by copying the Javascript code. However, it was developed in Ruby on Rails and consequently most of its internal logic should be in Ruby, and not in Javascript which will be used only for presentation. Therefore, it seems unlikely that we would be able to reuse the existing code.

II.2) Considering the team

We also have to consider the knowledge and preferences of the members of the development team.

Below are detailed the knowledge acquired by the two members of the working group during their years of programming education.

Antonin Lenfant:

Software Engineering (ISEP Paris)

- Web Development
  - HTML
  - PHP
  - Javascript
  - CSS
- Application Development
  - C, C++, C#
  - Java
  - Python

Miguel Navarro Reinoso

Telecommunications Engineering (Engineers Higher Technical School of Seville)

- Web Development
  - HTML
  - PHP
  - XML
- Application Development
  - C, C++
  - Java

While Antonin has experience in web programming, Miguel only has never used these languages in depth and only knows notions since his training is targeted mainly at application development, which implied using the Java or C programming languages.

In consequence, and also considering the reasons given in the previous section, we think it would be appropriate to develop a native application for this project.

III – Sources

http://fr.slideshare.net/peterfriese/cross-platform-mobile-development-11239246

http://es.slideshare.net/itsas_ehu/f4hc-2011-moviles

http://mashable.com/2010/08/11/cross-platform-mobile-development-tools/

http://fr.slideshare.net/kcresus/phonegap-vs-appcelerator

http://www.phonegap.com/

http://www.mosync.com/

http://www.appcelerator.com/

http://www.motorola.com/Business/US-EN/Business+Product+and+Services/Software+and+Applications/RhoMobile+Suite

http://xamarin.com/monotouch

http://www.adobe.com/fr/products/flash-builder.html