Multi-Platform Mobile Application Development Analysis

Lisandro Delia, Nicolas Galdamez, Pablo Thomas, Leonardo Corbalan, Patricia Pesado
Institute of Research in Computer Science III-LIDI. School of Computer Science.
National University of La Plata.
La Plata, Argentina
{ldelia, ngaldamez, pthomas, corbalan, ppesado}@lidi.info.unlp.edu.ar

Abstract—**Mobile devices have changed how we conceive software. There is a great range of development alternatives. In this paper, four different multi-platform development approaches (mobile web applications, hybrid, interpreted, and cross-compiled) are analyzed, and their most significant features through a case study are discussed.**

Keywords—mobile devices, multi-platform mobile applications, web mobile applications, hybrid mobile applications, interpreted mobile applications, cross-compilation

I. Introduction

Software development for mobile devices poses new challenges due to the unique features of this activity. The need to cope with various platforms, standards, protocols and network technologies, limited device capacity —although its constant evolution—, and market time demands are but a few of the issues faced. For this reason, software development for mobile devices is considerably different from traditional development [1].

To maximize their market presence, software products should be able to be run in as many devices as possible. One way to achieve this is by means of the native development of the application on each of the existing platforms using each platform's integrated development environment (IDE), language, and tools [2].

Native applications allow accessing all of the features offered by the device (camera, Global Positioning System [GPS], accelerometer, calendar, and so forth), their performance is high, Internet access is not strictly necessary, and they can be run on the background and issue notifications when user attention is required. These applications can be distributed/marketed through the corresponding online stores. However, the price for all these benefits is high: source code cannot be re-used for different platforms, which means that this type of development requires more effort and has higher costs for the development, updating, and distribution of new versions.

Multi-platform development, as opposed to native development, allows code reutilization. Building mobile web applications is a representative example of this approach. However, due to the limitations inherent to having to run the applications within a browser, software engineers have turned to other types of multi-platform applications that yield results that are closer to those obtained with native solutions. In this context, there are several sub-classifications [2] [3] [4], and their inherent features can be analyzed by building an prototype.

This paper goes deeper into the analysis of multiplatform mobile development, begun in [5].Section 2 describes in detail the most significant features that are common to multi-platform mobile applications. Then, the case study is briefly described, and the corresponding development in Phonegap with Jquery Mobile, Sencha Touch, Appcelerator Titanium 3, Xamarin and Delphi XE6 is discussed in the following sections. Finally, the conclusions and future lines of work are presented.

II. Multi-platform Mobile Applications

Multi-platform development is focused in optimizing the cost/benefit ratio by sharing the same code among the various versions developed for the different platforms. Its main advantages are: less development time and cost; near-native services with access to device hardware and availability of strong development environments (Delphi, Visual Studio) or, alternatively, the use of technologies that are well-known to web developers (HTML5, Javascript and CSS) that allow them transferring their knowledge and experience to the mobile paradigm.

Multi-platform applications can be classified into: mobile web applications, hybrid web applications, interpreted web applications, and applications generated by cross-compilation [2].

A. Mobile Web Applications

These applications, designed to be run within a browser, are developed using web technology (HTML, CSS and JavaScript) and present several favorable characteristics: they do not require adaptation to any operating environment, they are platform-independent, and they are easy and quick to launch.

On the other hand, their response times are lower due to client-server interaction, and they are less attractive than native applications because they are not installed in the device. Also, security restrictions imposed by the execution of the code through a browser limit the access that these applications have to all the features offered by the device [6].

B. *Hybrid Applications*

Hybrid applications use web technologies (HTML, Javascript and CSS), but are not run by a browser. Instead, they are run on a web container of the device that has greater access to device-specific features through an Application programming interface (API).

Hybrid applications offer great advantages because they allow code reuse for the various platforms, access to device hardware, and distribution through application stores [5].

Hybrid applications have two disadvantages in relation to native applications: i) user experience is affected by not using native components on the interface, and ii) execution is slower due to the load associated to the web container.

C. *Interpreted Applications*

Interpreted applications are a base project that is mostly translated to native code, with the rest being interpreted at runtime. Their implementation is platform-independent and uses several technologies and languages, such as Java, Ruby, XML, and so forth.

One of the main advantages of this type of applications is obtaining native interfaces, while their most remarkable obstacle is total dependence from the development environment. Appcelerator Titanium [7] is one of the most popular development environment at this moment.

D. *Applications Generated by Cross-Compilation*

These applications are compiled natively by creating a specific, high-performance version for each target platform. Some examples of development environments used to generate applications by cross-compilation are Applause [8], Embarcadero Delphi XE6 [22] and Xamarin [26].

The open development environment Applause uses as input a specific domain language based on Xtext framework [9], explicitly designed for data-oriented mobile applications, and it generates source code in Objective C, Java, C#, or Python. The characteristics of Delphi XE6 and Xamarin are discussed in subsequent sections.

III. CASE STUDY: WEBUNLP

WebUNLP is a virtual teaching and learning environment that allows educators to present their educational proposals and create a meeting space to communicate with their students, share study material, and generate a virtual educational experience [10].

Currently, WebUNLP is a web application for desktop and portable computers; it is not adapted to be used on mobile devices. However, as described in [5], certain functions of WebUNLP were extended to this type of devices as part of an case study on types of applications for mobile devices. In this paper, this analysis is further developed after carrying out case studies with multi-platform approaches: mobile web, hybrid, interpreted, and cross-compiled.

The utility selected from WebUNLP to adapt to mobile platforms is the electronic notice board, which is a communication tool used to post news on courses, such as changes in schedule, or reminders for assignment delivery deadlines.

Some of the requirements that must be met by the mobile application are:

- Users must be able to access the application with the same credentials they use to access the web version.
- Users must be able to access the notice board of all the courses they are taking, either as educator or student.
- Users must receive a notification on their devices when there is news published on their notice boards. This requirement is not possible for the web version that is accessible from desktop and/or portable computers.
- Users must have the same use experience with all operating platforms.
- The existing web application must be synchronized with the mobile application to be developed, which means that any changes made through the mobile application must be reflected on the web version and vice versa.

In relation to the graphic interface, a serial navigation design was proposed, following the order shown on the mockup [11] in Fig. 1.

IV. WEBUNLP MOBILE WEB APPLICATION

A web application that can access WebUNLP's notice board was developed; this application is available to any mobile device that has a browser that supports the features used for its development: HTML5, CSS and JavaScript.

Since data transmission/reception speed in a mobile device through WiFi, and 3G in particular, is lower than the speed of a desktop, the web version of WebUNLP's notice board is light and most of the requirements are implemented through Asynchronous JavaScript And XML (Ajax) [12] to avoid, in case of changes, the need to reload the entire page.

Due to the limitations that affect applications running on a browser, the feature for receiving a notification on the device when news are posted to the notice board cannot be implemented.

V. WEBUNLP HYBRID APPLICATION

A. *Hybrid Application developed with PhoneGap and Jquery Mobile*

The free, open source framework PhoneGap [13] was used, which allows developing mobile applications that use technologies that are common to all devices: HTML5, CSS and JavaScript. Also, the JavaScript framework called Jquery Mobile [14] was used to achieve interfaces with consistent aspect and behavior on the various mobile platforms.

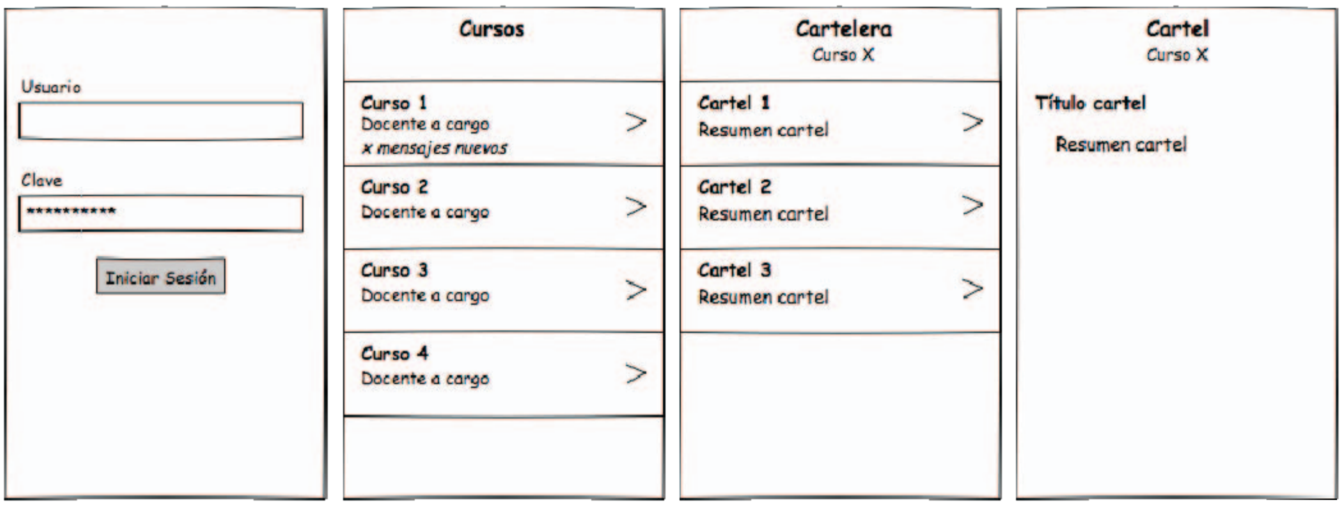


Fig. 1. Prototype mockup for WebUNLP notice board

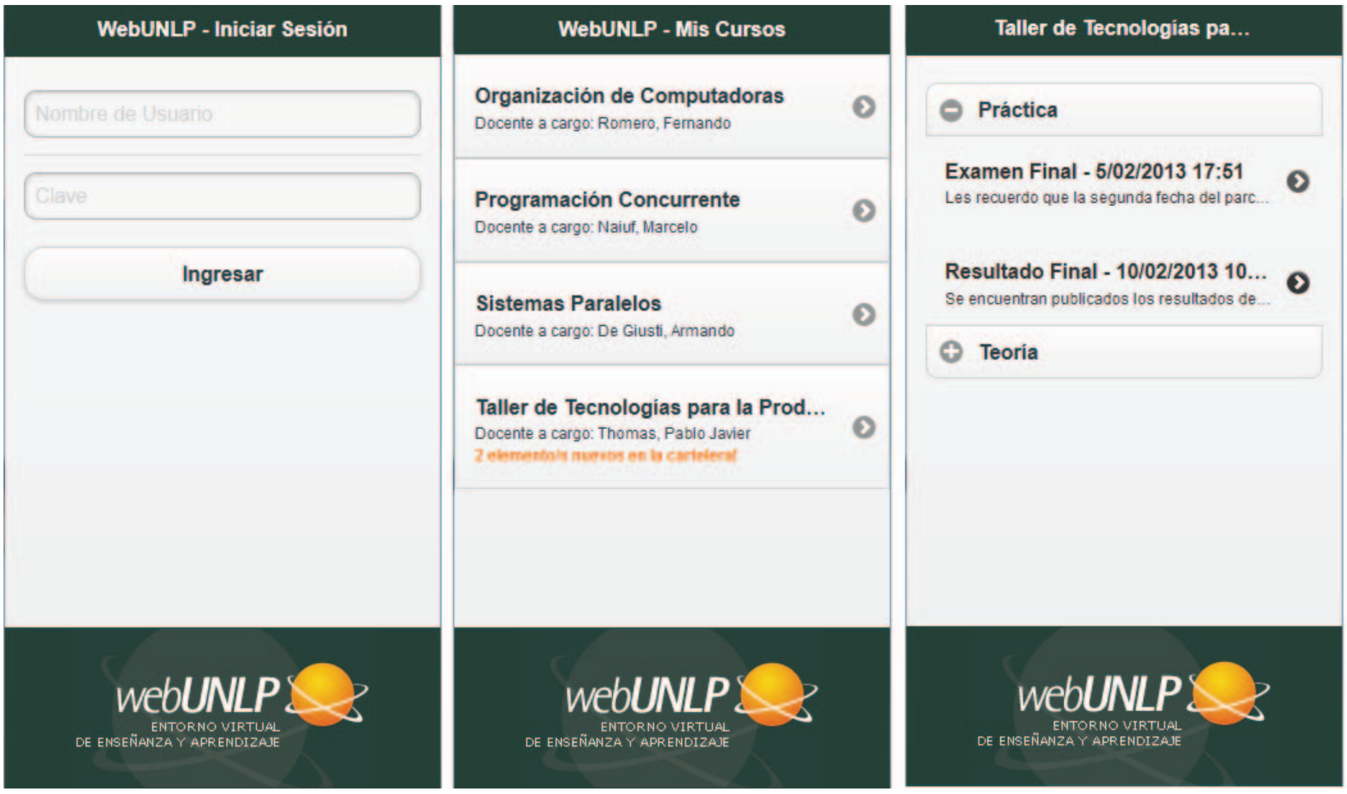
One of the requirements for the WebUNLP mobile application is the feature of receiving notifications on the device when there is news posted on the notice board. To meet this requirement, the official Phonegap's plug-in, known as PushPlugin [15], was used, which implements the feature for receiving messages on the device. This plug-in is implemented for the following platforms: Android, iOS, Windows Phone 8, and Amazon Fire OS.

Fig. 2 shows the interfaces of the application developed.

B. Hybrid Application Developed with Sencha Touch

Sencha Touch [16] is a free Model-View-Controller (MVC) JavaScript framework built on Ext JS's class system and specially designed for developing mobile web applications for touch devices [17].

The same as PhoneGap, Sencha Touch allows developers to create mobile applications from HTML5, CSS and Javascript web development.



Organización de Computadoras
Docente a cargo: Romero, Fernando

Programación Concurrente
Docente a cargo: Naiuf, Marcelo

Sistemas Paralelos
Docente a cargo: De Giusti, Armando

Taller de Tecnologías para la Prod...
Docente a cargo: Thomas, Pablo Javier
2 elemento/s nuevos en la cartelera!

Examen Final - 5/02/2013 17:51
Les recuerdo que la segunda fecha del parc...

Resultado Final - 10/02/2013 10...
Se encuentran publicados los resultados de...

Fig. 2. Application developed with PhoneGap and Jquery Mobile

The development for WebUNLP that used this framework benefited from the use of MVC's design pattern. This allowed for more flexible and readable code [17][18].

Sencha offers Sencha Command, a command-line multi-platform tool that provides automated tasks for the entire lifecycle of the application [19]. In the case of WebUNLP, it was used both for project creation as well as for application packing.

Fig. 3 shows the interfaces of the application developed.

VI. WebUNLP Interpreted Application with Appcelerator Titanium 3

To develop a mobile application based on the interpreted approach, the free, open source development environment Appcelerator Titanium 3 [7] was used.

This development environment uses the Alloy framework, designed for the agile development of mobile applications. Alloy is based on MVC architecture and supports the use of popular technologies such as Backbone.js [20] and Underscore.js [21].

Application controllers and models are remarkably simple and readable. To build the views, programming can be done through Javascript and Titanium's API, or by means of an XML specification with Titanium Style Sheets (TSS). The latter simplifies the process for creating the views, although a good interface visual editor to power it is still missing.

To meet device notification requirements, Titanium provides its PushNotifications module for Android and iOS platforms.

Fig. 5a shows the development experiment carried out with Titanium

VII. WebUNLP Application Generated by Cross Compilation

A. *Development with Xamarin/Visual Studio*

Xamarin is a proprietary, non-free development platform that allows writing and compiling fully-native applications for iOS, Android and Mac sharing the same base code entirely written in C# language. Even though it has its own IDE, called Xamarin Studio, it can be integrated with Microsoft Visual Studio to generate applications for Windows as well, including Windows RT for tablets and Windows Phone for cell phones.

Xamarin proposes a multi-platform development environment where the entire business logics code is shared. However, interfaces must be programmed separately for each of the target platforms (see Fig. 4). Thus, code reutilization, according to statistical studies carried out by the company Xamarin, is close to 85%.

For the WebUNLP development, Xamarin integrated with Microsoft Visual Studio was used. Following the most efficient strategy to work in this environment, a unique solution containing three different projects was created. One of these was used to generate the Android application, another one was used for Windows Phone 8, and the third project was

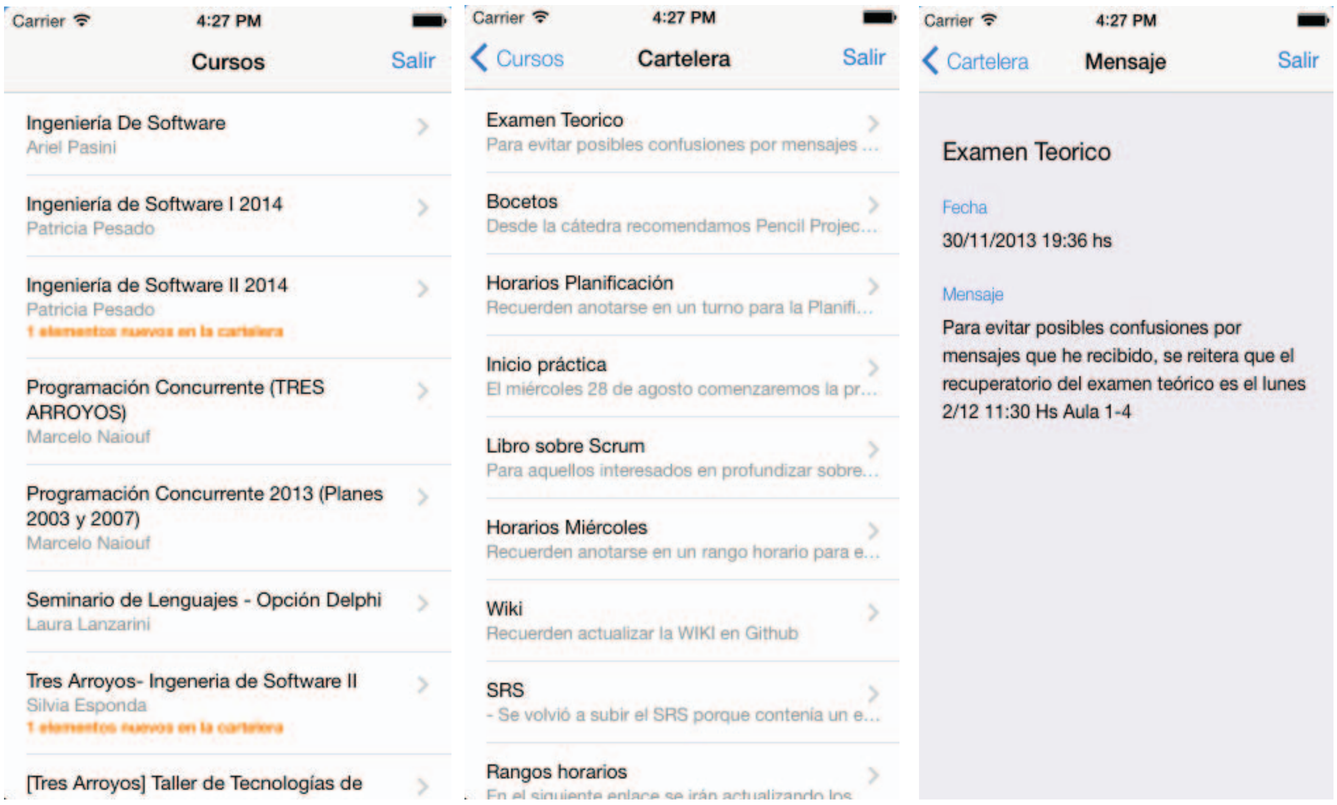

Carrier
4:27 PM
Cursos
Salir
Ingeniería De Software
Ariel Pasini
Ingeniería de Software I 2014
Patricia Pesado
Ingeniería de Software II 2014
Patricia Pesado
1 elementos nuevos en la cartelera
Programación Concurrente (TRES ARROYOS)
Marcelo Naiouf
Programación Concurrente 2013 (Planes 2003 y 2007)
Marcelo Naiouf
Seminario de Lenguajes - Opción Delphi
Laura Lanzarini
Tres Arroyos- Ingeneria de Software II
Silvia Esponda
1 elementos nuevos en la cartelera
[Tres Arroyos] Taller de Tecnologías de
Carrier
4:27 PM
Cursos
Cartelera
Salir
Examen Teorico
Para evitar posibles confusiones por mensajes ...
Bocetos
Desde la cátedra recomendamos Pencil Projec...
Horarios Planificación
Recuerden anotarse en un turno para la Planifi...
Inicio práctica
El miércoles 28 de agosto comenzaremos la pr...
Libro sobre Scrum
Para aquellos interesados en profundizar sobre...
Horarios Miércoles
Recuerden anotarse en un rango horario para e...
Wiki
Recuerden actualizar la WIKI en Github
SRS
- Se volvió a subir el SRS porque contenia un e...
Rangos horarios
Carrier
4:27 PM
Cartelera
Mensaje
Salir
Examen Teorico
Fecha
30/11/2013 19:36 hs
Mensaje
Para evitar posibles confusiones por mensajes que he recibido, se reitera que el recuperatorio del examen teórico es el lunes 2/12 11:30 Hs Aula 1-4

Fig. 3. Application Developed with Sencha Touch

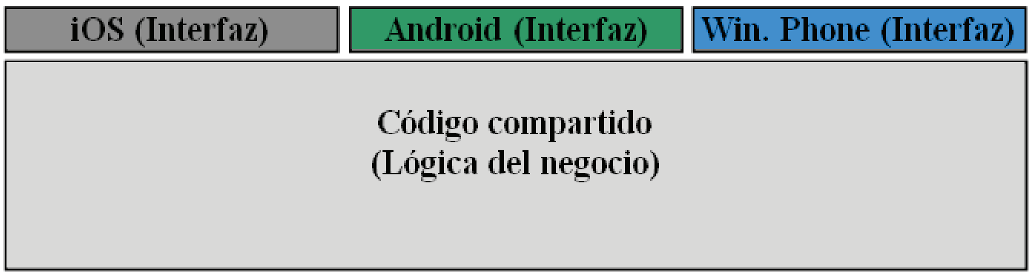


Fig. 4. Xamarin's unique development approach

used for implementing a portable class library (PCL) where the entire shared code. The three projects were developed in a joint and concurrent manner.

Even though one of the most significant advantages of Xamarin is code reutilization, it was strongly affected by the type of application being developed. The existing relation between interface complexity and business logics has a direct impact on the possibility of maintaining most of the code shared in the PCL. In the case of the development for WebUNLP, code reutilization was close to 50%. However, the advantages of using the same language, environment, and toolkit for the development of applications for different mobile platforms are not negligible.

Fig. 5b shows the interface developed for Windows Phone 8.

B. *Development with Embarcadero Delphi XE6*

Embarcadero Delphi XE6 [22] is a proprietary, non-free development platform that allows developers quickly create applications through a comfortable and intuitive visual environment. The application thus developed can be compiled for multiple platforms, including Windows, Mac, Android, and iOS.

Delphi XE6, the same as Xamarin, have the advantage of producing multiplatform applications that are fully native, with full access to device sensors and features (camera, notifications, GPS, accelerometer, etc.).

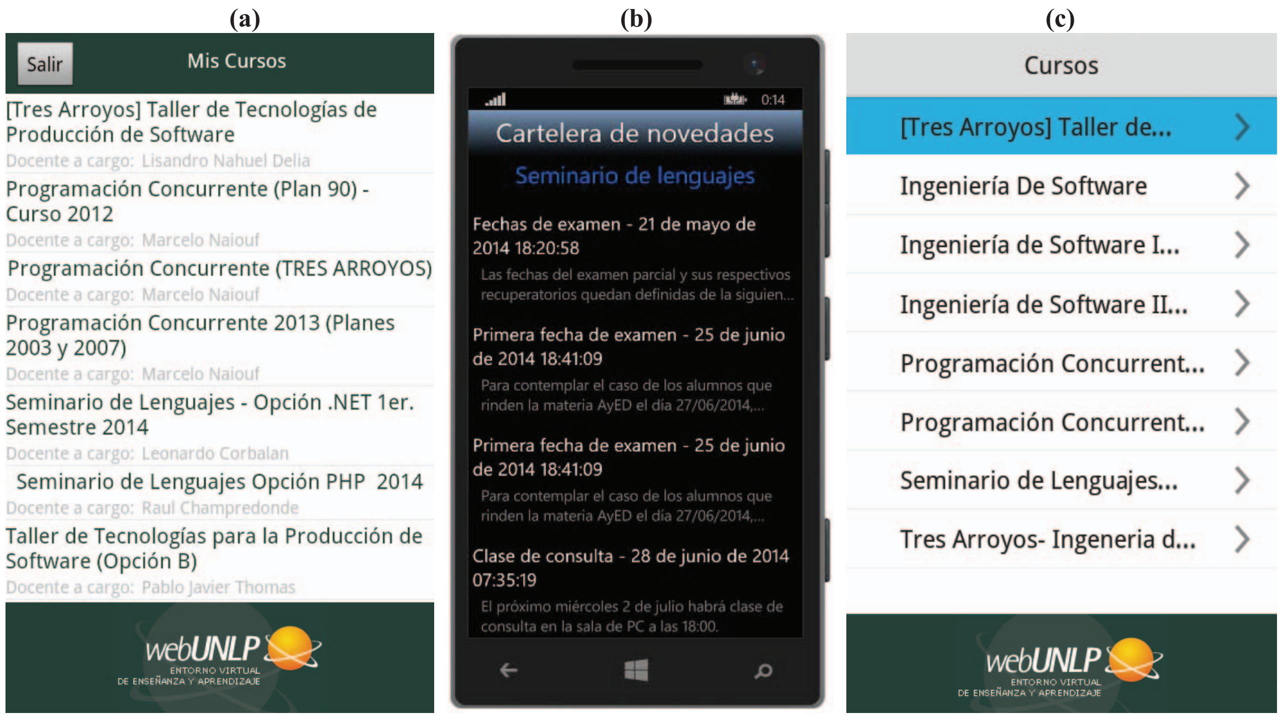
In particular, for the WebUNLP development, both the components provided by the tool to access RESTful services as well as its visual development environment were highly useful.

The feature for receiving notifications with WebUNLP news on the device, the TPushEvent component connected to the Kinvey service [23] was used.

Fig. 5c shows the interface of the application developed.

VIII. Conclusions

Given the increase in demand for specific software for


(a)

Salir
Mis Cursos

[Tres Arroyos] Taller de Tecnologías de Producción de Software
Docente a cargo: Lisandro Nahuel Delia

Programación Concurrente (Plan 90) - Curso 2012
Docente a cargo: Marcelo Naiouf

Programación Concurrente (TRES ARROYOS)
Docente a cargo: Marcelo Naiouf

Programación Concurrente 2013 (Planes 2003 y 2007)
Docente a cargo: Marcelo Naiouf

Seminario de Lenguajes - Opción .NET 1er. Semestre 2014
Docente a cargo: Leonardo Corbalan

Seminario de Lenguajes Opción PHP 2014
Docente a cargo: Raul Champredonde

Taller de Tecnologías para la Producción de Software (Opción B)
Docente a cargo: Pablo Javier Thomas

webUNLP
ENTORNO VIRTUAL DE ENSEÑANZA Y APRENDIZAJE

(b)

0:14

Cartelera de novedades
Seminario de lenguajes

Fechas de examen - 21 de mayo de 2014 18:20:58
Las fechas del examen parcial y sus respectivos recuperatorios quedan definidas de la siguien...

Primera fecha de examen - 25 de junio de 2014 18:41:09
Para contemplar el caso de los alumnos que rinden la materia AyED el día 27/06/2014,...

Primera fecha de examen - 25 de junio de 2014 18:41:09
Para contemplar el caso de los alumnos que rinden la materia AyED el día 27/06/2014,...

Clase de consulta - 28 de junio de 2014 07:35:19
El próximo miércoles 2 de julio habrá clase de consulta en la sala de PC a las 18:00.

(c)

Cursos

[Tres Arroyos] Taller de...
Ingeniería De Software
Ingeniería de Software I...
Ingeniería de Software II...
Programación Concurrent...
Programación Concurrent...
Seminario de Lenguajes...
Tres Arroyos- Ingeneria d...

webUNLP
ENTORNO VIRTUAL DE ENSEÑANZA Y APRENDIZAJE

Fig. 5. Application developed with: a) Titanium; b) Xamarin/Visual Studio, and c) Delphi XE6

mobile devices and the growing number of platforms, new tools and technologies have appeared for mobile development. Since companies need to cover as large a portion of the market as possible, multi-platform application implementation is an appealing option aimed at reducing times and costs.

Considering the difficulties inherent to the development of native applications for the various mobile platforms, an analysis in the development of multi-platform mobile applications was carried out.

WebUNLP, a virtual teaching and learning environment used by various grade and post-grade courses of the National University of La Plata, was used as case study. Development was replicated with four types of applications: web, hybrid, interpreted, and cross-compiled.

The main advantages of the mobile web application are its development simplicity and that it is easily distributed (all that is needed is a web browser). On the downside, it does not allow receiving notifications on the device when there is news posted on the notice board.

On the other hand, the hybrid application developed with Phonegap was able to combine the development simplicity of the web approach with the use of all the features offered by the device. Its performance is better than that of a mobile web application, but worse than that of a native application.

In the case of interpreted applications, Titanium was studied, and one of its main advantages is the generation of native code for the interface, which helps achieve a high level of performance. On the downside, there is no visual tool for interface design.

Finally, in the case of applications generated by cross-compilation, two alternatives were explored: Xamarin/Visual Studio and Delphi XE6. In both cases, fully native applications were obtained, although with different levels of code reutilization: 100% with Delphi XE6 and 50% with Xamarin/Visual Studio. This difference is caused by the need of coding interfaces individually for each platform when using Xamarin/Visual Studio.

It therefore follows that all the development alternatives analyzed in this paper have both advantages and disadvantages. If priority is given to good performance and user experience towards native applications, interpreted and crossed compilation approaches are the most appropriate. On the other hand, if the goal consists of minimizing the development effort, then hybrid and mobile web approach are the best options. This analysis suggests that the first approach should be adopted if access to higher capacities from mobile device is required.

IX. Future Work

As a further step in the study of multi-platform application development, an analysis will be carried out in the future to determine the scope of other tools available in the market, such as AppMethod [24], Applause [25], among others.

To carry out this study, quantitative and qualitative values will be experimentally measured, for example, amount of reused code, performance, battery consumption, look and feel, user adoption, etc., that will allow accurately characterizing each approach/tool analyzed.

Additionally, the mobile application for WebUNLP will be expanded by adding functionalities that are already available in the web platform, such as messaging and forum, as well as the addition of a new chat service.

References

[1] Hayes, I. S. Just Enough Wireless Computing. Prentice Hall Professional Technical Reference . 2002. ISBN:0130994618

[2] Spyros Xanthopoulos, Stelios Xinogalos, A Comparative Analysis of Cross-platform Development Approaches for Mobile Applications, BCI' 2013, Greece

[3] Yonathan Aklilu Redda, Cross platform Mobile Applications Development, Norwegian University of Science and Technology, Norwegian University of Science and Technology, Norwegian University of Science and Technology, Master in Information Systems, June 2012.

[4] Dalmasso I., Datta S.K., Bonnet C. Nikaein N., Survey, comparison and evaluation of cross platform mobile application development tools, Wireless Communications and Mobile Computing Conference (IWCMC), 2013 9th International.

[5] Delia L., Galdamez N. Thomas P., Pesado P., Un Análisis Experimental de Tipo de Aplicaciones para Dispositivos Móviles, XVIII Congreso Argentino de Ciencias de la Computación, CACIC 2013.

[6] Tracy, K.W., Mobile Application Development Experiences on Apple's iOS and Android OS, Potentials, IEEE, 2012

[7] http://www.appcelerator.com/

[8] https://github.com/applause/applause

[9] http://www.eclipse.org/Xtext/

[10] http://webunlp.unlp.edu.ar

[11] http://es.wikipedia.org/wiki/Mockup

[12] https://developer.mozilla.org/en/docs/AJAX

[13] http://phonegap.com/

[14] http://jquerymobile.com/

[15] https://github.com/phonegap-build/PushPlugin

[16] http://www.sencha.com/

[17] Kosmaczewski, A. (2013). Sencha Touch 2 Up and Running. O'Reilly.

[18] Clark, J. (2013). Creating Mobile Apps with Sencha Touch 2. Packt Publishing.

[19] Lee Bonstra. (2014). Hands-On Sencha Touch 2.

[20] http://backbonejs.org/

[21] http://underscorejs.org/

[22] https://www.embarcadero.com/es/products/delphi

[23] http://www.kinvey.com/

[24] http://www.appmethod.com/

[25] https://github.com/applause/applause

[26] http://xamarin.com/